View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Columbia University Academic Commons

Empirical Evaluation of Approaches to Testing Applications without Test Oracles

Christian Murphy, Gail Kaiser
Dept. of Computer Science
Columbia University
New York NY 10027
{cmurphy, kaiser}@cs.columbia.edu

ABSTRACT

Software testing of applications in fields like scientific computing, simulation, machine learning, etc. is particularly challenging because many applications in these domains have no reliable "test oracle" to indicate whether the program's output is correct when given arbitrary input. A common approach to testing such applications has been to use a "pseudo-oracle", in which multiple independently-developed implementations of an algorithm process an input and the results are compared: if the results are not the same, then at least one of the implementations contains a defect. Other approaches include the use of program invariants, formal specification languages, trace and log file analysis, and metamorphic testing.

In this paper, we present the results of two empirical studies in which we compare the effectiveness of some of these approaches, including metamorphic testing and runtime assertion checking. These results demonstrate that metamorphic testing is generally more effective at revealing defects in applications without test oracles in various application domains, including non-deterministic programs. We also analyze the results in terms of the software development process, and discuss suggestions for both practitioners and researchers who need to test software without the help of a test oracle.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification; D.2.5 [**Software Engineering**]: Testing and Debugging

General Terms

Reliability, Verification

Keywords

Software Testing, Oracle Problem, Metamorphic Testing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...$5.00.

1. INTRODUCTION

Applications in the fields of scientific computing, simulation, optimization, machine learning, etc. are sometimes said to be "non-testable programs" [37] because there is no reliable "test oracle" to indicate what the correct output should be for arbitrary input. In some cases, it may be impossible to know the program's correct output *a priori*; in other cases, the creation of an oracle may simply be too hard. Despite the importance and prevalence of such applications, the absence of a test oracle clearly presents a challenge when it comes to detecting subtle errors, faults, defects or anomalies in software in these domains.

For these types of applications, there may in fact be a small subset of the input domain for which the correct output *can* be known, and a "niche oracle" may exist in such cases [37]. In the general case, though, a popular approach to testing such applications is to use a "pseudo-oracle" [13], in which multiple independently-developed implementations of an algorithm process an input and the results are compared: if the results are not the same, then at least one of the implementations contains a defect.

In their 2001 survey paper [6], Baresi and Young described other techniques for addressing the lack of a test oracle (sometimes referred to as "the oracle problem"), including the use of runtime assertion checking, extrinsic interface contracts, formal specification languages, and trace checking and log file analysis. Recently, others have applied metamorphic testing [9] to non-testable programs. This approach is based on the notion that, although it is impossible to know in general what the expected output should be for any given input, it may be possible to predict how changes to the input should cause changes to the output, and thus identify a relation between a *set* of inputs and the *set* of their respective outputs.

In previous work [24], we applied metamorphic testing to machine learning applications and presented the results of initial studies to demonstrate its effectiveness at detecting defects. Here, we present the results of new empirical studies in which we investigate a wider variety of application domains and compare the effectiveness of different approaches to testing non-testable programs: using a niche oracle, metamorphic testing, and runtime assertion checking.

Our contributions include: (1) an empirical comparison of techniques for testing software without test oracles; (2) evidence that some of these techniques can also be used for non-deterministic applications; and (3) suggestions for both practitioners and researchers who need to develop and test software without the help of a test oracle.

2. TECHNIQUES INVESTIGATED

2.1 Niche Oracle

It is impossible to know whether the output of these non-testable programs is correct for *arbitrary* input, because there is no general test oracle to cover all cases. However, in some trivial cases, it may be possible to know what the correct output should be, based on analysis of the algorithm and understanding how it should perform under certain conditions. We define a "*niche oracle*" as one that will indicate correctness or incorrectness of the output in an otherwise non-testable program, but only for a limited set of test cases.

Determining the boundaries of such niche oracles (i.e., at what point does the program become non-testable) is outside the scope of this work, but we assume for each of the applications in our studies that such oracles exist, either because the correct output can easily be calculated by hand, or there is an agreed-upon correct output for the given input within the particular domain.

Weyuker points out that *"Experience tells us that it is frequently the 'complicated' cases that are most error-prone"* [37], thus limiting the effectiveness of the niche oracle. However, we assume that many developers of applications without test oracles would use such a technique, and as it is relatively simple and cheap to implement, we choose to include it in our studies.

2.2 Metamorphic Testing

An early approach to testing applications without test oracles was to use a "pseudo-oracle". This approach is based upon the notion of "N-version programming" [8], in which independent programming teams develop an application (perhaps using different technologies or programming languages) from the same specification; then, identical sets of input data are processed and the results are compared. If the results are the same, that does not necessarily mean that they are correct (since the implementations may all have the same defect, for instance), but if the results are *not* the same, then a defect has likely been revealed in at least one of the implementations.

One issue with the pseudo-oracle approach is that multiple versions of a program may simply not exist. In the absence of multiple implementations, however, metamorphic testing [9] can be used to produce a similar effect. Metamorphic testing is designed as a general technique for creating follow-up test cases based on existing ones, particularly those that have not revealed any failure. In metamorphic testing, if input x produces an output $f(x)$, the function's so-called "metamorphic properties" can then be used to guide the creation of a transformation function t, which can then be applied to the input to produce $t(x)$; this transformation then allows the expected value of the output $f(t(x))$ to be predicted based on the (already known) value of $f(x)$. If the output is not as expected, then a defect must exist. Of course, this can only show the existence of defects and cannot demonstrate their absence, since the correct output cannot be known in advance (and even if the outputs are as expected, both could be incorrect), but metamorphic testing provides a mechanism of revealing defects in such non-testable programs by use of a built-in pseudo-oracle.

A simple example of a function to which metamorphic testing could be applied would be one that calculates the standard deviation of a set of numbers. Certain transformations of the set would be expected to produce the same result. For instance, permuting the order of the elements should not affect the calculation; nor would multiplying each value by -1, since the deviation from the mean would still be the same. Furthermore, other transformations will alter the output, but in a predictable way. For instance, if each value in the set were multiplied by 2, then the standard deviation should be twice as much as that of the original set.

Thus, given one set of numbers, three more sets (one with the elements permuted, one with each multiplied by -1, and another with each multiplied by 2) can be created to get a total of four test cases; moreover, given the output of only the first test case (even if it is not known whether it was right or wrong), the output of the other three can be predicted. Clearly metamorphic testing can be applied in the absence of an oracle: regardless of the values, if the different outputs for the different inputs are not as expected, then there must be a defect in the implementation.

2.3 Assertion Checking

The use of assertions for fault detection has been employed since the early days of software development [11], and modern programming languages such as ANNA and Eiffel, as well as C and Java, have built-in support for assertions that allow programmers to check for properties at certain control points in the program. For instance, assertion checking may be used for program safety, such as to ensure that pointers are not null, array indices are in bounds, or that data structure integrity is maintained.

When used in applications without test oracles, assertions can ensure some degree of correctness by checking that function input and output values are within a specified range, the relationships between variables are maintained, and a function's effects on the application state are as expected. For instance, in the standard deviation example above, the assertions may be that the return value of the function is always non-negative, the values in the input array never change, etc. While these statements alone do not ensure correctness (many other functions would have the same properties), any violation of them at runtime indicates a defect.

To aid in the creation of assertions, we used the Daikon invariant detection tool [14]. Daikon observes the execution of multiple program runs and creates a set of likely invariants, which can then be used as assertions for subsequent runs of the program. This has been shown to be effective at detecting defects like the ones we use in our studies [27]. Although it is possible to customize the types of invariants that Daikon can detect, in our experiments we only use its out-of-the-box features. Note that the invariants created by Daikon only include function pre- and post-conditions, and do not incorporate any assertions that are within the function itself; however, since our study only considers system-level testing, we feel that this level of granularity is sufficient, as any defects revealed by in-function assertions would also be captured by the function-level invariants.

2.4 Approaches Not Investigated

Formal specification languages like Z or Alloy could be used to declare the properties of the application, typically in advance of the implementation to communicate intended behavior to the developers. However, Baresi and Young point out that a challenge of using specification languages as oracles is that "*effective procedures for evaluating the predicates*

or carrying out the computations they describe are not generally a concern in the design of these languages" [6], that is, the language may not be suitable for describing how to know whether the implementation is meeting the specification. Although previous work has demonstrated that formal specification-based assertions can be effective in acting as test oracles [12], the specifications need to be complete in order to be of practical use in the general case, as pointed out in [34]. Thus, we do not consider formal specification languages in our evaluation.

Additionally, it may be possible to perform trace or log file analysis to determine whether or not the program is functioning correctly, if for instance it is conforming to certain properties (like a sequence of execution calls or a change in variable values) that are believed to be related to correct behavior; or, conversely, to see if it is *not* conforming to these properties. However, others have demonstrated that the content of the log/trace files must be carefully planned out, often by someone with great knowledge of the application, in order to be of use in the general case [3]. That is, the creation of an oracle to tell if the trace is correct can be just as difficult as creating an oracle to tell if the output is correct in the first place, assuming it is even possible at all.

Last, despite the common use of the pseudo-oracle approach, and even though different implementations existed for many of the applications we considered, we could not use the approach in our study because none of the pseudo-oracles consistently produced the exact same output as the versions that we used in the experiments. In some cases, this was by design; in others, it was because of different interpretations of the algorithm. We note, though, that criticism of the pseudo-oracle approach (and N-version programming) is well-documented from both a practical [1] and theoretical [20] point of view, further justifying its omission.

3. STUDY #1: NON-TESTABLE PROGRAMS

In this study, we applied the testing techniques to six real-world "non-testable programs". The goal of the study is to measure the effectiveness of each technique, and to understand why certain approaches are more suitable than others for testing such applications.

3.1 Test Subjects

Three of the applications used in this study are from the domain of machine learning. In this domain, programs are designed to detect previously unknown properties of data sets, and thus have no test oracle for arbitrary inputs.

The **Support Vector Machines** (SVM) algorithm [36] is a classification algorithm used in numerous real-world applications, ranging from facial recognition to computational biology.[1] We tested the SVM implementation in the Weka 3.5.8 toolkit for machine learning [39], written in Java. **C4.5** [31] is another classification algorithm, which uses decision trees. In our experiment, we tested C4.5 release 8, implemented in C. Last, **MartiRank** [16] is a ranking algorithm that is used as part of a prototype application for predicting electrical device failures, developed in C by researchers at Columbia University's Center for Computational Learning Systems.

We also tested the simulation tool **JSim** [38], developed by members of the Laboratory for Advanced Software Engineering Research at the University of Massachusetts-Amherst. Discrete event simulators can be used to model real-world processes in which events occur at a particular time and affect the state of the system [29], and can be considered to be non-testable programs because they fall into a category of software that Davis and Weyuker describe as *"Programs which were written in order to determine the answer in the first place. There would be no need to write such programs, if the correct answer were known"* [13]. JSim is implemented in Java and has been used in the modeling of the flow of patients through a hospital emergency room [32].

Another application we investigated is **Lucene** [4], an open-source text search engine library that is part of the Apache project and is implemented in Java. When Lucene produces the results of a search query, each item is given a relevance score, and the results are sorted accordingly. With a known, finite set of documents to search, it is trivial to check whether the items in the query result really do contain the query keywords (and that documents not in the result do not contain the keywords), but the relevance scores and the sorted ordering cannot be known in advance (if they could, there would be no need to have the tool).

The final application investigated in this study is **gaffitter** [5], which is an open-source application implemented in C++ that uses a genetic algorithm to arrange an input list of files and directories into volumes of a certain size capacity, such as a CD or DVD, in a way that the total unused (wasted) space is minimized. This is an example of optimization software, specifically for solving the "bin-packing problem," which is known to be NP-hard.[2] Solutions to optimization problems in the domain of NP-hard qualify as non-testable programs: after all, by definition, verifying whether a solution is correct is just as hard as finding the solution in the first place.

3.2 Methodology

In this experiment, we used mutation testing to systematically insert defects into the source code and then determined whether or not the mutants could be killed (i.e., whether the defects could be detected) using each approach. Mutation testing has been shown to be suitable for evaluation of effectiveness, as experiments comparing mutants to real faults have suggested that mutants are a good proxy for comparisons of testing techniques [2]. These mutations fell into three categories: (1) comparison operators were mutated, e.g. "less than" was switched to "greater than or equal"; (2) mathematical operators were mutated, e.g. addition was switched to subtraction; and (3) off-by-one errors were introduced for loop variables, array indices, and other calculations that required adjustment by one. Based on our discussions with the researchers who implemented MartiRank, we chose these types of mutations because we felt that these represented the types of errors most likely to be made in these types of applications. All functions in the programs were candidates for the insertion of mutations; each variant that we created had exactly one mutation (i.e., we did not create any program variants with more than one mutation).

In our experiments, we did not consider mutations that were not on the execution path for the given input data, nor did we include "equivalent mutants" that did not affect the overall output. Additionally, if the mutation yielded a fatal error (crash), an infinite loop, or an output that was clearly

[1] http://www.clopinet.com/isabelle/Projects/SVM/applist.html

[2] http://en.wikipedia.org/wiki/Bin_packing_problem

Application	Comparison	Math	Off-by-one	Total
C4.5	8	15	5	28
MartiRank	20	23	26	69
SVM	30	24	31	85
JSim	0	1	5	6
Lucene	0	4	11	15
gaffitter	15	19	32	66

Table 1: Types of mutants used in Study #1.

wrong (for instance, being nonsensical to someone familiar with the application, or simply being blank), that variant was also discarded since any of the considered approaches would be able to detect such defects.

Table 1 shows the number and type of mutants that were suitable for our experiment for each of the six applications. The variation in the number of mutants for each application is due to the different sizes of the source code, the different number of points in which mutations could be inserted, and the different number of mutants that led to fatal errors or obviously wrong output.

3.2.1 Creating Niche Oracles

As the SVM and C4.5 algorithms are fairly commonly used in the machine learning community, there exist sample data sets for which the correct output is actually known; these are used by developers of SVM and C4.5 implementations to make sure there are no obvious defects in their code. Of course, producing the correct output for this input does not ensure that the implementation is error-free, but these data sets can be used as simple "niche oracles". We used the "iris" and "golf" data sets from the UC-Irvine machine learning repository [26] for both SVM and C4.5, since the data sets are relatively small in size and the correct output can easily be calculated by hand. We created four variants of each data set, using different values or different ordering of inputs, so that the output would still be predictable, for a total of eight data sets.

For MartiRank, we hand-crafted two data sets for which we could predict what the expected output should be, and confirmed our expectation using the "gold standard" implementation, which we assumed to be error-free. As with the SVM and C4.5 data sets, we created four variants of each one, each of which would yield a predictable output.

For JSim, the niche oracle was based on test cases that were created by the developers of the program. These include simple processes that exercise only particular parts of the simulator, such as executing steps in sequential order.

For the Lucene niche oracle, we manually created a simple document corpus in which the first file consisted of only the word "one", the second contained "one two", the third contained "one two three", and so on. Thus, we could easily predict not only the results of queries like "one AND three" but also know the expected relevance ranking. Note that we could not predict the relevance scores themselves, only their relative values; the scores do not have any meaning on their own.[3]

To create a niche oracle for gaffitter, we generated a set of files, each of which had a size that was a power of two (i.e., one byte, two bytes, four bytes, eight bytes, sixteen bytes, etc.). For any given target size, there would be exactly one optimal set of files to reach that value. As an example, if the target were 11 bytes, then a collection including the files of size one, two, and eight bytes would be the only optimal solution. Thus, it would be possible to know what the correct output should be.

Note these data sets for the niche oracle were selected so that the percentage of line coverage was approximately the same for the different applications and the different testing approaches, thus removing any bias due to the number of data sets used. That is, we do not expect that the niche oracles would be significantly more effective if additional data sets were used, assuming the percentage of line coverage did not likewise increase significantly.

3.2.2 Metamorphic Testing

For the three machine learning applications (SVM, C4.5, and MartiRank), we devised metamorphic properties according to the guidelines set forth in [23]. Each property was verified with the "gold standard" version to make sure that the property was, in fact, expected to hold. The properties are defined in Table 2.

(1) Permuting the order of the examples in the input data should create a semantically-equivalent output.
(2) Multiplying each attribute value in the input data by a positive constant (in our case, ten) should create a semantically-equivalent output.
(3) Adding a positive constant (in our case, ten) to each attribute value in the input data should create a semantically-equivalent output.
(4) Negating each attribute value in the input data should create a semantically-equivalent output, but with all values negated.

Table 2: Metamorphic properties used for testing machine learning applications

For SVM and C4.5, we used six data sets from the UC-Irvine machine learning repository [26]. These data sets could not be used for MartiRank because it requires numeric input, so we used 10 randomly-generated data sets that covered a variety of equivalence classes, using a technique described in [22], to try to obtain line coverage similar to the data sets used in the niche oracle experiment. For each data set, each mutated version was executed with both the original and the data set modified according to each metamorphic property, and the results were compared: if the outputs were not as expected, then the mutant was considered to be "killed" (i.e., the defect was found).

For JSim, we identified two metamorphic properties. First, because the event timings as specified in the configuration have no time units, multiplying each by a constant value should effectively increase the overall time to complete the process by the same factor. For instance, if there are N events in the process, and the time to complete each step is t_0, t_1, t_2... t_{N-1} respectively, and the total time to complete the process is observed to be T, then changing the event timings to $10t_0$, $10t_1$, $10t_2$... $10t_{N-1}$ would be expected to change the overall process time to $10\,T$. Second, we considered the effect that changing the event timings would have on the utilization rates for the different resources in the simulation. Specifically, if the event timings were increased by a positive constant, then the utilization rate (i.e., the time the resource is used, divided by the overall process time) of the most utilized resource would be expected to decrease, since

[3]http://article.gmane.org/gmane.comp.jakarta.lucene.user/12076

Property	Description
romeo OR juliet == juliet OR romeo	Commutative property of boolean operator
romeo juliet^0.25 == romeo^4 juliet	Boosting one term by a factor of 4 should be the same as boosting the other term by a factor of 0.25
romeo OR foo == romeo	Inserting an OR term that does not exist should not affect the results
romeo NOT foo == romeo	Excluding term that does not exist should not affect the results

Table 3: Metamorphic Properties Used for Testing Lucene

the total increase in the overall process time would outweigh the small increase in the resource's time spent working. In the metamorphic testing of JSim, we used the data sets from in the emergency room simulation work presented in [32].

The metamorphic properties for Lucene are based on those presented in [41], in which the authors applied metamorphic testing to Internet search engines such as Yahoo!, Google, and Microsoft Live Search. We also included others based on the specific syntactic features of Lucene, as described in its user manual. The properties are listed in Table 3. The corpus used for Lucene was the text of Shakespeare's "Romeo and Juliet".[4]

For gaffitter, we were only able to identify one metamorphic property. Specifically, if the number of generations used in the genetic algorithm were to increase, the overall quality of the result (in this case, the amount of unwasted space) should be non-decreasing. The intuition is that subsequent generations should never reduce the quality, but rather only candidate solutions with better quality should be included. Of course, we cannot say that increasing the number of generations will increase the quality, since an optimal solution may have been reached, but there should not be any decrease in quality. In our metamorphic testing experiment with gaffitter, we used a collection of 84 files ranging in size from 118 bytes to 14.9MB, and set targets of 1kB, 1MB, and 100MB. The non-deterministic aspects of the genetic algorithm were controlled by specifying a seed for the random number generator.

3.2.3 Invariant Detection

To create the set of invariants that we could use for runtime assertion checking, we applied Daikon to the "gold standard" (i.e., with no mutations) of the applications. We executed the applications multiple times, with the different data sets described above. This was done in order to reduce the number of "spurious invariants" that Daikon might produce. For instance, if a program is only run once with an input called "input.data", Daikon will detect that the variable representing the input file is always equal to "input.data" and create an invariant as such; clearly this is spurious because it only happens to be true in this one particular case.

Once the set of invariants had been created, we executed each mutated version with the same data sets used for metamorphic testing and used Daikon's invariant checker tool to ensure that none of the invariants were being violated. If a violation occurred, then the defect had been found.

Note that in practice, it may not be possible to derive a complete set of invariants from the "gold standard": after all, if one knew that the implementation were the gold standard, testing it would not be necessary. However, in this experiment, we wanted to create a set of invariants that would best reflect the correctness of the implementation, and this seemed the best way to automate that. An alternative would have been to hand-generate a set of program invariants, but this would have required more detailed understanding of the implementation details. Our aim was to compare the techniques assuming that the tester only had an understanding of what the applications are meant to do, and not how they are implemented.

[4]http://shakespeare.mit.edu/romeo_juliet/index.html

3.3 Results

Table 4 shows the effectiveness (percentage of distinct defects found in each application) for the niche oracle, metamorphic testing, and assertion checking.

Overall, metamorphic testing was most effective, killing 80.6% of the mutants in the six applications. Assertion checking found 69.8% of the 269 defects, and the niche oracle detected 65.4%.

App.	#	Niche Oracle	Metamorphic Testing	Assertion Checking
SVM	85	78 (91.7%)	83 (97.6%)	68 (80.0%)
C4.5	28	22 (78.5%)	26 (92.8%)	28 (100%)
MartiRank	69	47 (68.1%)	69 (100%)	57 (82.6%)
JSim	6	4 (66.7%)	6 (100%)	6 (100%)
Lucene	15	11 (73.3%)	11 (73.3%)	9 (60%)
gaffitter	66	14 (21.2%)	22 (33.3%)	20 (30.3%)
Total	**269**	**176 (65.4%)**	**217 (80.6%)**	**188 (69.8%)**

Table 4: Distinct Defects Detected in Study #1.

3.4 Analysis

C4.5. As opposed to the other applications, runtime assertion checking was more effective than metamorphic testing at revealing the C4.5 defects, although only slightly so (28 mutants killed, compared to 26). Upon investigation, we realized that the mutants not killed by metamorphic testing were both math mutants, in which a calculation was modified. In both cases, this resulted in a violation of an assertion, but not a violation of a metamorphic property.

One of the two defects is found in the code in Figure 1, which comes from the "FormTree" function used to create nodes of the decision tree. Lines 179-182 use a for-loop macro to initialize the values of an array `ClassFreq`, which is used to calculate the frequency with which a class (or "label") appears in the input data. Line 183 then uses the macro to iterate values of `i` from `Fp` (the first index of the array) to `Lp` (the last index). On line 185, the `Class` function is used to determine the class of the item from the input data, and `ClassFreq` is updated with the corresponding weight.

If there is a defect on line 185, such that the addition of the weight is changed to subtraction, the values in `Class-`

Freq become negative, in violation of the invariant (detected by Daikon) that the values should always be greater than or equal to zero; thus, the defect is detected. However, metamorphic testing does not reveal this defect: the changes made to the training data do not affect the calculation of class frequency, and even though the resulting output is incorrect, none of the metamorphic transformations cause the properties to be violated.

```
179 ForEach(c, 0, MaxClass)
180 {
181    ClassFreq[c] = 0;
182 }
183 ForEach(i, Fp, Lp)
184 {
185    ClassFreq[ Class(Item[i]) ] += Weight[i];
186 }
```

Figure 1: Sample code in C4.5 implementation. A defect on line 185 causes a violation of the assertion checking, but not of the metamorphic property.

We see from Table 1 that C4.5 had a proportionally higher number of math mutants, compared to the other applications; as the example in Figure 1 is a math mutant, as well, it may seem that assertion checking is more effective for these types of defects. However, we observed in all applications that metamorphic testing is approximately as effective for all three types of defects (comparison operator mutations, math operator mutations, and off-by-one errors). Rather, in this case we can see that there are cases when assertion checking is useful for detecting defects related to the *value* of variables, whereas metamorphic testing addresses *changes* to those values.

MartiRank. All 69 of the defects in MartiRank were detected by metamorphic testing, making it much more effective than the other two approaches, especially the niche oracle. One of the reasons why the niche oracle was so ineffective is that the data sets we hand-crafted were necessarily much smaller than the ones for metamorphic testing and runtime assertion checking: the niche oracle data sets consisted of 10-20 values, whereas the data sets used with the other techniques had 200-20,000. The small size of the data sets was necessary so that we could accurately and reliably calculate what the correct output should be; anything much larger would be too complex to calculate, or would be so trivial that it would not be of much interest.

Although this seems to put the niche oracle at a disadvantage, note that both the niche oracle data sets and the data sets used for metamorphic testing were attaining approximately the same overall line coverage (around 70%) regardless of their size, and we only considered mutants that were on the execution path, to ensure that there was a possibility that they would affect the program output. Upon analysis of the mutations that were missed by the niche oracle, we discovered that often the defects only caused incorrect outputs when the data sets were large, as expected.

For instance, in one case there was an off-by-one defect in which an error in the allocation of memory for an array meant that one of the elements could be overwritten, but this did not occur when the input was "small" because there was no need for the creation of the additional arrays that did the overwriting.

In another case, a defect in the "merge" step of the merge sort algorithm caused the indices that delineated the sublists to be incorrect, making it possible for elements to be sorted in the wrong order. This defect only affected the final output if the incorrectly sorted elements happened to be on the boundary between two parts of the MartiRank output; this never occurred for the smaller niche oracles, but was more likely to occur in the larger data sets used for the other approaches, primarily because MartiRank continued to sort the larger data sets multiple times, whereas MartiRank had already generated the correct output with the small data set after only a few rounds of sorting.

Note that the definition of a "non-testable program" is not only limited to software for which an oracle does not exist: it can also include software for which the creation of an oracle is too difficult [37], and the niche oracle by its nature tends to address a smaller, simpler part of the input domain. This does serve to point out, though, that the metamorphic testing and runtime assertion checking techniques can be used with arbitrary input data of any size, thus improving their effectiveness.

SVM. Metamorphic testing was more effective than the other two approaches for SVM, particularly for off-by-one errors: all 31 were found by metamorphic testing, while assertion checking missed seven and the niche oracle missed four.

Figure 2 shows a sample of the SVM code that was tested in this experiment. The array `m_class` identifies the classification, or "label", for each data point in the input. This particular piece of code attempts to segregate the data into those that have a label of 1 (into set `m_I1`), and those that do not (into set `m_I4`). An off-by-one mutant has been inserted into line 524, such that the for loop omits the final element of `m_class`.

```
524 for (int i = 0; i < m_class.length-1; i++) {
525    if (m_class[i] == 1) {
526       m_I1.insert(i);
527    } else {
528       m_I4.insert(i);
529    }
530 }
```

Figure 2: Snippet of code from Weka implementation of SVM, with an off-by-one error in the for-loop condition on line 524

Metamorphic testing detects this defect because, when the ordering of the data in the input is permuted, a different value is omitted in the loop, and thus it is possible that the contents of both `m_I1` and `m_I4` are different (compared to the original invocation) when using the permuted set: if element k had been in `m_I1` in the original execution, but ends up being omitted as a result of permutation, and element k' is placed into `m_I4`, then both sets would be different. The result is that, when the program later performs "validation" (i.e., using the input data to determine the predictive accuracy of the output), the element that was omitted is classified differently in each of the two executions; since the results are expected to be the same, the defect is thus revealed.

On the other hand, when using the niche oracle, either `m_I1` is different or `m_I4` is different (but *not* both, since an element is omitted from one but the other stays the same) in

the execution with the mutated version. For the small data sets we used, this did not affect the validation results. Larger data sets would possibly be affected by this difference, but would no longer be useful as niche oracles since the correct output would no longer be predictable.

Assertion checking fails to find this defect because the relevant invariants simply deal with sizes of the sets (e.g. "`m_I1.size` > 0") and the fact that `m_class` does not change. Despite the mutant in line 524, these invariants still hold.

gaffitter. For gaffitter, none of the results are particularly impressive, though we do see again that metamorphic testing is the most effective of the three. One of the reasons that gaffitter (and, presumably, most other implementations of genetic algorithms) is fairly insensitive to the mutations that we introduced is that many of the calculations have only a subtle effect on the output of the program. This is actually by design, since the point of a genetic algorithm is to simulate small changes to the candidate solutions with the hopes that they increase the quality of the result, but any one particular change is unlikely to have a dramatic effect. This is why the niche oracle, in particular, performed so poorly.

For instance, consider the pseudo-code in Figure 3, which summarizes lines 212-226 in GeneticAlgorithm.cc and is typical in most genetic algorithms. This function takes two candidate solutions (i.e., sets of items) called CS1 and CS2, and creates a child candidate solution, which contains some items from CS1 and some from CS2. On line 2, a random number is used to determine whether CS1 and CS2 should cross over at all. If so, on line 3, a crossover point (i.e., an index in the list of items) is randomly chosen. In line 4, the first items (up to the crossover point) of CS1 are merged with the last items (from the crossover point to the end) of CS2.

```
1  crossover(CS1, CS2) {
2      if (rand() < crossover_rate) {
3          crossover_point = rand() * length(CS1);
4          Child = CS1[1, crossover_point] +
                   CS2[crossover_point + 1, length(CS2)];
5          return Child;
6      }
7      else return null;
8  }
```

Figure 3: Sample code from genetic algorithm

Considering the mutations such as those we inserted into gaffitter, it is clear that many of them will indeed affect the output of this particular function, but are likely to have little or no effect on the quality of the overall solution. For example, in line 3, an off-by-one mutant may change the value of the crossover point, so that an item from CS1 may incorrectly be replaced instead with an item from CS2 (or vice-versa), but including or excluding a single item is unlikely to have a dramatic effect on the overall quality of the result when the number of items is relatively small, as in our niche oracle. The same can be said of potential off-by-one or mathematical mutants on line 4 in which the items from CS1 and CS2 are selected to create the child: if the wrong items are selected, the result of the function will differ from what is "correct", but if the incorrectly-created child candidate solution does not have a quality that is much higher than what the correctly-created child would have, then it will not be selected for inclusion in the next generation, and the fact that there is a defect is moot.

In cases like the niche oracle that we created for gaffitter, in which there is relatively small number of possible combinations of the files to be packed into the bins, an optimal (and correct) solution will eventually be found, even when there is a defect in the implementation. This explains why the niche oracle was so poor at detecting defects.

Note that we were only able to identify a single metamorphic property for this application; if we were able to identify more, it follows that more mutants may have been killed by further testing (we believe that the inability to identify additional metamorphic properties is more a limitation of our understanding of the implementation than it is of the approach itself).

JSim and Lucene. Because of the small number of mutations used in the experiment for JSim and Lucene, the results may not be statistically significant, but serve to demonstrate that metamorphic testing is at least as effective as the other approaches. This may not be an important finding in and of itself, but combined with the other results, indicates that metamorphic testing is no less effective than the current state of the art at finding defects in a variety of applications without test oracles.

4. STUDY #2: NON-DETERMINISM

The challenge of testing applications without test oracles can be compounded by the fact that some applications in these domains can be non-deterministic, in that multiple runs of the program with the same input will not necessarily produce the same output. Note that being non-deterministic does not necessarily imply being non-testable: a program that produces an equally distributed random integer between 1 and 10 is non-deterministic, but has an oracle (specifically, that over many executions, the number of times each integer is returned should be the same; and that no single execution should return a result outside the specified range). However, non-deterministic applications that do not produce numerical output, or for which the distribution or range of results cannot be known in advance, could be considered non-testable programs, and thus we include some of these applications in our study.

4.1 Test Subjects

In our first experiment described above in Section 3, all the applications were deterministic, and thus none of the defects found were a result of non-determinism. In our second experiment, we apply the testing approaches to non-deterministic applications to measure the effectiveness of each technique.

The JSim discrete event simulator (described in the previous experiment) can be non-deterministic depending on the system configuration. Specifically, the amount of time each step in the simulation takes to complete may be random over a range, either using an equal distribution or using a "triangle" distribution with an inflection point at a specified mode; thus, the time it takes for the entire process to complete may be non-deterministic.

The MartiRank application described above also can be non-deterministic. In generating its final output (i.e., the ranking of the input data), MartiRank performs sorting on the elements. If the data set contains missing values (which is very likely to happen when using real-world data [23]), MartiRank will randomly place the missing elements throughout the sorted list. Thus, subsequent executions may yield

different results, making the output non-deterministic. Note that in our first study, the MartiRank data sets had no missing values, thus the output was deterministic.

4.2 Methodology

For JSim, we used mutation testing to insert defects that were related to the non-deterministic parts of the application. To create defects that would affect these parts of JSim, we systematically inserted 19 different off-by-one mutants related to the event timing, so that when the configuration specified a timing range from A to B for a given event, the actual range would be [A+1, B] or [A, B-1] or [A+1, B-1]. We could not use defects that had ranges starting at A-1 or ending at B+1 because of checks that already existed within the code that would notice such out-of-range errors and raise an exception.

For MartiRank, we were able to use mutation testing as described as in the first two experiments and inserted a total of 59 defects (18 mathematical operator mutants and 41 off-by-one mutants). Note that these are not the same set of mutants used in Study #1.

4.2.1 Niche Oracle

Although the applications being tested are non-deterministic, a niche oracle can exist if it is possible to enumerate all of the possible "correct" outputs. For both MartiRank and JSim, we created simple data sets for which we could know what all the possible outputs would be, and confirmed this by using the gold standard version (without any mutations). Then, we executed each mutated version up to 100 times; if it produced an output that was not one of the possible correct outputs, the defect had been found.

Of course, it is possible that there would be false negatives in that the mutated version may randomly not produce an erroneous output during the 100 executions. However, given that the input was contrived so that there were only 2-3 possible correct outputs, we felt that the likelihood of this happening was quite small.

4.2.2 Metamorphic Testing

In this experiment, we used *statistical metamorphic testing* (SMT), which has been proposed as a technique for testing non-deterministic applications that do not have test oracles [17]. SMT can be applied to programs for which the output is numeric, such as the overall event timing in JSim, and is based on the statistical properties of multiple invocations of the program. That is, rather than consider the output from a single execution, the program is run numerous times so that the statistical mean and variance of the values can be computed. Then, a metamorphic property is applied to the input, the program is again run numerous times, and the new statistical mean and variance are again calculated. If they are not as expected, then a defect has been revealed.

For JSim, we specified non-deterministic event timing over a range [α, β] and ran the simulation 100 times to find the statistical mean μ and variance σ of the overall event timing in the process (i.e., the time to complete all the steps). We then configured the simulator to use a range [10α, 10β], ran 100 more simulations, and expected that the mean would be 10μ and the variance would be 10σ. Of course, they results would not *exactly* meet those expectations, so we used a Student T-test [15] to see if any difference was statistically significant; if so, then the defect was revealed. We validated this approach with the gold standard implementation (i.e., in which we had not inserted any defects), and found that the resulting distributions were not significantly different, with $p < 0.05$.

Heuristic metamorphic testing (HMT) [24] is a similar approach that is based on SMT but can be used for non-deterministic applications or functions that produce non-numerical output, such as the ranking of elements in MartiRank. For such applications, certain metamorphic properties can be applied so that the new output is expected to be "similar" to the original. The issue, of course, is knowing just *how* similar they should be. This can be measured by observing multiple executions with the original input, measuring their similarity (according to some heuristic metric depending on the problem domain, as in the "heuristic oracle" approach for general applications [18]), and then checking that the executions with the new input (after applying metamorphic properties) are comparably similar.

In this experiment, we ran 100 executions of the program with a given input, which produced a list of elements (i.e., the ranking of examples in MartiRank). We could then compare the similarly of those lists of elements using the Spearman Footrule Distance [35], which yields a normalized equivalence rating of how similar the lists are, given that they are always expected to contain the same elements. We then permuted the input (by changing the ordering of the examples in the data set for MartiRank), ran 100 executions with the new input, and calculated the normalized equivalence of the new outputs. As with SMT, we again used a Student T-test to compare the results, and we validated this approach using the gold standard implementation.

Note that just as metamorphic testing is essentially a pseudo-oracle approach, in that "expected results" are not necessarily "correct results", SMT and HMT can only indicate defects, not correctness: if the statistical mean and variance are not as expected (in SMT) or if the normalized equivalence is not as expected (as in HMT), that indicates that something is amiss; but if the results *are* as expected, a defect may still exist but not be revealed.

4.2.3 Invariant Detection

As in the previous study, we applied Daikon to the applications and had it observe multiple program executions of the "gold standard" implementation so that it could create a list of invariants. The mutated versions of the software were then executed with the same data sets to see whether any invariants were violated. Because of the non-deterministic nature of the applications, each was run 100 times, since some invariants might only be violated occasionally.

4.3 Results

Table 5 shows the number of defects found by the three techniques in this study.

The statistical and heuristic metamorphic testing approaches were most effective for both applications, detecting a total of 59 (75.6%) of the 78 total mutants.

Application	#	Niche Oracle	Metamorphic Testing	Assertion Checking
MartiRank	59	15 (25.4%)	40 (67.7%)	27 (45.7%)
JSim	19	0 (0%)	19 (100%)	0 (0%)

Table 5: Defects Found in Study #2.

4.4 Analysis

MartiRank. Heuristic Metamorphic Testing was the most effective technique at detecting the defects in MartiRank. When MartiRank is run repeatedly, the final result of the ranked elements is expected to be more or less the same each time: even though MartiRank places missing values randomly throughout the list, the other elements will stay in about the same place, so the normalized equivalence (i.e., the heuristic used to measure the "sameness" of multiple lists) is high. One of the metamorphic properties is that permuting the order of the elements should result in a similar ranking (since sorting does not depend on original input order). However, if there is a defect such that the known elements of the list are sorted incorrectly, then the resulting lists will *not* be similar to the original (since the known elements are out of place and the normalized equivalence will be lower), and the defect will be revealed.

As we observed in the first study, the MartiRank niche oracle data sets were too small to be effective, as errors with sorting the known values only appeared in the larger data sets; of course, once the data set got to be too large, the results were no longer predictable, and a niche oracle could not be used.

One reason that runtime assertion checking was not as effective as it was for the deterministic aspects of MartiRank is that Daikon only detected 1927 invariants in this study, compared to 3666 in the first one. We attribute this to the non-determinism: over multiple executions, it may not be true in *every* case that, for instance, one variable is always greater than another, and thus Daikon disregards this as a likely invariant. With fewer invariants, it follows that there are likely to be fewer violations.

JSim. Both the niche oracle and assertion checking techniques were unable to detect any of the JSim defects related to event timing because each approach only considers a single execution of the program, or of the function that produces the random number in the specified range. Consider a function that is meant to return a number in the range [A, B], but has a defect so that the range is actually [A, B-1]. No single execution will violate the invariant that "the return value is between A and B", so assertion checking does not reveal this defect. As for the niche oracle, if it is known that the program calls the function 10 times, for instance, then we can know that the total overall timing (i.e., the sum of the random numbers) should be a value in the range [$10A$, $10B$]. Even with the defect, though, this still will be true: no program execution will have a total overall timing outside that range.

However, statistical metamorphic testing *will* detect this defect because over 100 executions of the program (as in our experiment), the mean and variance show a statistically significant difference compared to what is expected; the other approaches necessarily only run the program once, and do not consider the trend of the program over a number of independent executions.

5. THREATS TO VALIDITY

Some of the details of the studies may cause the reader to question the generalizability of the results.

The programs that were investigated may not be representative of all "non-testable programs". However, these were selected so as to demonstrate that metamorphic testing is applicable to a range of application domains that do not have test oracles: specifically, machine learning (C4.5, MartiRank, SVM), discrete event simulation (JSim), information retrieval (Lucene), and optimization (gaffitter). Moreover, within the domain of machine learning, we chose applications from different subdomains, specifically decision tree classifiers (C4.5), linear classifiers (SVM), and ranking (MartiRank). Given the number of test subjects and the breadth of application domains, plus the fact that many of these applications are (or are part of) industrial systems, we feel that the results can safely be generalized.

Another potential threat to validity involves the data sets that were used in the experiments. For the niche oracles, the data sets were smaller than the data sets used for metamorphic testing and runtime assertion checking. Of course, the data sets were smaller out of necessity because, for larger data sets, the correct output could not easily be predicted, thus they could not be used as niche oracles. Also, as noted, the line coverage of the data sets was approximately the same for all the testing approaches, and variations of the original niche oracle data sets were generated to attain a better mix of values. It is possible that the results may have been different had different data sets been chosen, but with very few exceptions, we did not see any single data set being particularly good (or particularly bad) at revealing defects, and the results would not have changed significantly had any data set been omitted.

In these studies, we used Daikon to create the program invariants that would be used in runtime assertion checking. Although some researchers have questioned the usefulness of Daikon-generated invariants compared to those generated by humans [30], Daikon is generally accepted as the state-of-the-art in automatic invariant detection [28]. We chose to use the tool so that we could eliminate any human bias or human error in creating the invariants, which would require a great deal of knowledge of the source code, considering the complexity of the applications we evaluated. Although in practice invariants may typically be generated by hand, we did not feel that such an approach would scale to larger, more complex systems such as the ones investigated here, especially considering the complications of deriving invariants *after* the software had already been developed.

Daikon can generate spurious invariants if there are not enough program executions, but we mitigated this by running the programs multiple times with different inputs, and by ignoring any invariants that were obviously a result of the data sets that were being used. Note that even if some of the invariants that detected defects in the study were, in fact, spurious, this would serve to increase the apparent effectiveness of runtime assertion checking by increasing the number of defects it discovered. Given that our results indicate that metamorphic testing is more effective, any experimental error related to spurious Daikon invariants would, in fact, only strengthen that conclusion.

6. DISCUSSION

Although metamorphic testing was shown to be more effective in detecting defects in applications without test oracles, in practice one of the principal issues with the approach is "where do the metamorphic properties come from?". Indeed, in our first study we observed that the properties based on permutation and negation found most of the defects; if we had only used the properties based on addition and mul-

tiplication, the results would have come out quite differently.

In [23], we enumerated six classes of metamorphic properties, but they are generally only applicable to applications that deal with numerical inputs and outputs. Although non-testable programs tend to fall into this category, some applications without test oracles work with non-numeric data, such as those in computational linguistics or compiler optimization; thus, the proposed classes of properties may not always be applicable.

Although there may be no "silver bullet" when it comes to devising the metamorphic properties of a particular application, we note that in *any* software testing approach, the tester still must have *some* knowledge or understanding of the program in order to devise test cases. Semantic knowledge of the program to be tested is required for writing use cases, devising equivalence classes, creating test input, and designing regression tests [25]. Thus, metamorphic testing is no different from other black-box techniques in that it is assumed that the tester will have enough knowledge to create test cases (in this case, metamorphic properties), as guided by the program specifications and a general understanding of what the code is meant to do.

To facilitate this, we suggest that the metamorphic properties be identified as part of the planning and design phase, and included in the program specification. Thus, the application designer, who is likely to have the best understanding of how the program should react when its inputs are changed, can pass this knowledge on to the tester, who can then implement the specific test cases.

Metamorphic testing can also be applied to individual functions, and thus is suitable for use in unit testing and integration testing, particularly when there may be no oracle for the unit(s) being tested. Thus, in the implementation phase, developers can use metamorphic testing for subcomponents of the application; given that they are the ones who implemented the code, it should be easy for them to devise metamorphic properties (in the same way that it is easy for them to devise program invariants [11]).

7. RELATED WORK

Others have empirically evaluated the effectiveness of metamorphic testing when applied to non-testable programs [17, 40]. Our study is most similar to that of Hu et al. [19], in which they compare metamorphic testing and runtime assertion checking. However, their study only compares deterministic applications. To our knowledge, we are the first to compare metamorphic testing to other approaches for testing applications that are non-deterministic and do not have a test oracle. Also, in their study the assertions are created manually by graduate students, and not automatically detected with a tool such as Daikon, as we do here. This stems from the fact that the programs used in their experiments are considerably smaller than the applications we investigate here; manually generating the assertions would likely have been error-prone in our experiment, given the complexity of the applications. Although this seems to reduce the novelty of our work, we point out that the authors did not consider an approach based on simple inputs (i.e., the "niche oracle") in their evaluation; given the apparent ease of such a testing technique, it is an important contribution to empirically show that metamorphic testing can actually reveal more defects, as we have shown here. It is worth mentioning, though, that our results are consistent with theirs in that metamorphic testing is shown to be more effective than assertion checking at revealing defects in non-testable programs.

Coppit and Haddox-Schatz have demonstrated that runtime assertions can be effective in acting as test oracles [12], but they use a formal specification language and do not use any tool to detect invariants automatically. Others have reported on the effectiveness of runtime assertions in general [21, 33], or specifically considered using invariants in place of test oracles, such as [10]. However, none of these compare their results to any other simpler testing techniques such as metamorphic testing, which we have shown to be more effective at detecting defects.

8. CONCLUSION

We have demonstrated through two empirical studies that metamorphic testing is more effective than other suggested techniques at detecting defects in applications without test oracles. Aside from these results, we have also shown that such techniques can also be used for non-deterministic applications, and have made suggestions for both practitioners and researchers who need to develop and test software without the help of a test oracle. Addressing the testing of applications without oracles has been identified as a future challenge for the software testing community [7], and we hope that our findings here help others who are also concerned with the quality of such non-testable programs.

9. ACKNOWLEDGMENTS

We would like to thank Adam Cannon, T.Y. Chen, Lori Clarke, Lee Osterweil, and Sal Stolfo for their guidance and assistance. The authors are members of the Programming Systems Lab, funded in part by NSF grants CNS-0905246, CNS-0717544, CNS-0627473, and CNS-0426623, and NIH grant 1U54CA121852-01A1.

10. REFERENCES

[1] T. Anderson and P. A. Lee. *Fault Tolerance: Principles and Practice*. Prentice Hall, 1981.

[2] J. H. Andrews, L. C. Briand, and Y. Labiche. Is mutation an appropriate tool for testing experiments? In *Proc of the 27th International Conference on Software Engineering (ICSE)*, pages 402–411, 2005.

[3] J. H. Andrews and Y. Zhang. Broad-spectrum studies of log file analysis. In *Proc. of the 22nd International Conference on Software Engineering (ICSE)*, pages 105–114, 2000.

[4] Apache Lucene. http://lucene.apache.org.

[5] D. A. Augusto. Genetic algorithm file fitter. http://gaffitter.sourceforge.net/.

[6] L. Baresi and M. Young. Test oracles. Technical Report CIS-TR01 -02, Dept. of Computer and Information Science, Univ. of Oregon, 2001.

[7] A. Bertolino. Software testing research: Achievements, challenges, dreams. In *Proc. of ICSE Future of Software Engineering*, pages 85–103, May 2007.

[8] L. Chen and A. Avizienis. N-version programming: A fault-tolerance approach to reliability of software operation. In *Proc of the 8th Symposium on Fault-Tolerance Computing*, pages 3–9, 1978.

[9] T. Y. Chen, S. C. Cheung, and S. Yiu. Metamorphic testing: a new approach for generating next test cases. Technical Report HKUST-CS98-01, Dept. of Computer Science, Hong Kong Univ. of Science and Technology, 1998.

[10] Y. Cheon. Abstraction in assertion-based test oracles. In *Proc. of Seventh International Conference on Quality Software*, pages 410–414, 2007.

[11] L. A. Clarke and D. S. Rosenblum. A historical perspective on runtime assertion checking in software development. *ACM SIGSOFT Software Engineering Notes*, 31(3):25–37, May 2006.

[12] D. Coppit and J. Haddox-Schatz. On the use of specification-based assertions as test oracles. In *Proc of the 29th Annual IEEE/NASA Software Engineering Workshop*, 2005.

[13] M. D. Davis and E. J. Weyuker. Pseudo-oracles for non-testable programs. In *Proc. of the ACM '81 Conference*, pages 254–257, 1981.

[14] M. D. Ernst, J. Cockrell, W. G. Griswold, and D. Notkin. Dynamically discovering likely programming invariants to support program evolution. In *Proc. of the 21st International Conference on Software Engineering (ICSE)*, pages 213–224, 1999.

[15] W. S. Gosset. The probable error of a mean. *Biometrika*, 6(1):1–25, March 1908.

[16] P. Gross et al. Predicting electricity distribution feeder failures using machine learning susceptibility analysis. In *Proc. of the 18th Conference on Innovative Applications in Artificial Intelligence*, 2006.

[17] R. Guderlei and J. Mayer. Statistical metamorphic testing - testing programs with random output by means of statistical hypothesis tests and metamorphic testing. In *Proc of the Seventh International Conference on Quality Software*, pages 404–409, 2007.

[18] D. Hoffman. Heuristic test oracles. *Software Testing and Quality Engineering*, pages 29–32, 1999.

[19] P. Hu, Z. Zhang, W. K. Chan, and T. H. Tse. An empirical comparison between direct and indirect test result checking approaches. In *Proc of the 3rd international workshop on software quality assurance*, pages 6–13, 2006.

[20] J. Knight and N. Leveson. An experimental evaluation of the assumption of independence in multi-version programming. *IEEE Transactions on Software Engineering*, 12(1):96–109, 1986.

[21] N. G. Leveson, S. S. Cha, J. C. Knight, and T. J. Shimeall. The use of self checks and voting in software error detection: An empirical study. *IEEE Transactions on Software Engineering*, 16(4):432–443, April 1990.

[22] C. Murphy, G. Kaiser, and M. Arias. Parameterizing random test data according to equivalence classes. In *Proc of the 2nd international workshop on random testing*, pages 38–41, 2007.

[23] C. Murphy, G. Kaiser, L. Hu, and L. Wu. Properties of machine learning applications for use in metamorphic testing. In *Proc. of the 20th international conference on software engineering and knowledge engineering (SEKE)*, pages 867–872, 2008.

[24] C. Murphy, K. Shen, and G. Kaiser. Automated metamorphic system testing. In *Proc of the 2009 ACM International Symposium on Software Testing and Analysis (ISSTA)*, pages 189–199, 2009.

[25] G. J. Myers, T. Badgett, T. M. Thomas, and C. Sandler. *The art of software testing*. Wiley, 1979.

[26] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz. UCI repository of machine learning databases. University of California, Dept of Information and Computer Science, 1998.

[27] J. W. Nimmer and M. D. Ernst. Automatic generation of program specifications. In *Proc of the 2002 International Symposium on Software Testing and Analysis (ISSTA)*, pages 232–242, 2002.

[28] J. W. Nimmer and M. D. Ernst. Invariant inference for static checking: An empirical evaluation. In *Proc of 10th International Symposium on the Foundations of Software Engineering (FSE)*, pages 11–20, 2002.

[29] M. Pidd. *Computer simulation in management science*. Wiley, 1998.

[30] N. Polikarpova, I. Ciupa, and B. Meyer. A comparative study of programmer-written and automatically inferred contracts. In *Proc of the 2009 ACM International Symposium on Software Testing and Analysis (ISSTA)*, pages 93–104, 2009.

[31] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1993.

[32] M. S. Raunak, L. J. Osterweil, A. Wise, L. A. Clarke, and P. L. Henneman. Simulating patient flow through an emergency department using process-driven discrete event simulation. In *Proc of the ICSE Workshop on Software Engineering for Health Care*, pages 73–83, 2009.

[33] D. S. Rosenblum. A practical approach to programming with assertions. *IEEE Transactions on Software Engineering*, 21(1):19–31, January 1995.

[34] S. Sankar, A. Goyal, and P. Sikchi. Software testing using algebraic specification based test oracles. Technical Report CSL-TR-93-566, Dept. of Computer Science, Stanford Univ., 2003.

[35] C. Spearman. Footrule for measuring correlation. *British Journal of Psychology*, 2:89–108, June 1906.

[36] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[37] E. J. Weyuker. On testing non-testable programs. *Computer Journal*, 25(4):465–470, November 1982.

[38] A. Wise. JSim agent behavior specification language. http://laser.cs.umass.edu/documentation/jsim/language.html.

[39] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition*. Morgan Kaufmann, 2005.

[40] P. Wu. Iterative metamorphic testing. In *Proc of 29th Annual International Computer Software and Applications Conference*, pages 19–24, 2005.

[41] Z. Q. Zhou, T. H. Tse, F.-C. Kuo, and T. Y. Chen. Automated functional testing of web search engines in the absence of an oracle. Technical Report TR-2007-06, Department of Computer Science, The University of Hong Kong, 2007.

11. APPENDIX A. INVARIANTS USED IN EXPERIMENT #1

Following are the Daikon invariants (in the Daikon format) for the applications used in the experiment described in Section 3.

11.1 C4.5

==========

..Category()::ENTER
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Subtree
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::TargetClassFreq == ::ThreshErrs
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::ClassSum[] (lexically)
::Weight[] >= ::ClassSum[] (lexically)
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)

::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::ClassSum[] (lexically)
::UnknownRate[] <= ::ClassSum[] (lexically)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitGain[] < ::ClassSum[] (lexically)
::SplitGain[] <= ::ClassSum[] (lexically)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::ClassSum[] (lexically)
::SplitInfo[] <= ::ClassSum[] (lexically)
::Slice1[] == ::Slice2[]
::Slice1[] < ::ClassSum[] (lexically)
::Slice1[] <= ::ClassSum[] (lexically)
::Dev[] > ::ClassSum[] (lexically)
::Dev[] >= ::ClassSum[] (lexically)
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Bar != null
::Bar[] elements >= 0.0
::UnknownRate != null
::UnknownRate[] elements >= 0.0
::SplitGain != null
::SplitGain[] elements != 0
::SplitGain[] elements >= -0.0010
::SplitInfo != null
::SplitInfo[] elements >= 0.0
::Slice1 != null
::Slice1[] elements == 0.0
::Slice1[] one of [0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::Slice2 != null
::Subset != null
::Subsets != null
::Subsets[] elements == 0
::PossibleValues == null
::Changed one of 0, 1
::Val has only one value
::Val[] == [0.0, 0.0010, 0.0050, 0.01, 0.05, 0.1, 0.2, 0.4, 1.0]
::Val[] elements <= 1.0
::Val[] elements >= 0.0
::Val[] sorted by <
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::ClassSum[] elements >= 0.0
::Delimiter == -1
::Subdef has only one value
::TRf != null
::optind == 3
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::Changed
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::Changed
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::Changed
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::Changed
::MaxItem > ::Delimiter

::MaxItem > ::optind
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus == ::Tested
::SpecialStatus < ::Fn
::PossibleValues in ::AttValName[]
::FileName < ::Fn
::FileName == ::optarg
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::VERBOSITY <= ::Changed
::TRIALS >= ::Changed
::MINOBJS > ::Changed
::CF in ::Dev[]
::Weight[] > ::SplitGain[] (elementwise)
::Freq[][0] <= ::ValFreq[] (elementwise)
::Gain[] <= ::Info[] (elementwise)
::SplitGain[] < ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
::Changed > ::Delimiter
::Changed < ::optind
==========
..Category()::EXIT
::MaxAtt == orig(::MaxAtt)
::MaxClass == orig(::MaxClass)
::MaxDiscrVal == orig(::MaxDiscrVal)
::MaxItem == orig(::MaxItem)
::Item == orig(::Item)
::Item[] == orig(::Item[])
::MaxAttVal == orig(::MaxAttVal)
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::MaxAttVal[] == orig(::MaxAttVal[])
::ClassName == orig(::ClassName)
::AttName == orig(::AttName)
::AttValName == orig(::AttValName)
::AttValName[] == orig(::AttValName[])
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Subtree
::VERBOSITY == orig(::VERBOSITY)
::VERBOSITY == orig(::SUBSET)
::VERBOSITY == orig(::UNSEENS)
::VERBOSITY == orig(::PROBTHRESH)
::VERBOSITY == orig(::WINDOW)
::VERBOSITY == orig(::INCREMENT)
::VERBOSITY == orig(::MultiVal)
::VERBOSITY == orig(::Subtree)
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::TRIALS == orig(::TRIALS)
::TRIALS == orig(::GAINRATIO)
::TRIALS == orig(::BATCH)
::TRIALS == orig(::AllKnown)
::MINOBJS == orig(::MINOBJS)

::CF == orig(::CF)
::Pruned == orig(::Pruned)
::Pruned[] == orig(::Pruned[])
::TargetClassFreq == ::ThreshErrs
::TargetClassFreq == orig(::TargetClassFreq)
::TargetClassFreq == orig(::ThreshErrs)
::Raw == orig(::Raw)
::Raw[] == orig(::Raw[])
::Weight == orig(::Weight)
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::ClassSum[] (lexically)
::Weight[] >= ::ClassSum[] (lexically)
::Weight[] == orig(::Weight[])
::Weight[] > orig(::ClassSum[]) (lexically)
::Weight[] >= orig(::ClassSum[]) (lexically)
::Freq == orig(::Freq)
::Freq[] == orig(::Freq[])
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::Freq[][0] == orig(::Freq[][0])
::ValFreq == orig(::ValFreq)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ValFreq[] == orig(::ValFreq[])
::ClassFreq == orig(::ClassFreq)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] == orig(::ClassFreq[])
::Gain == orig(::Gain)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Gain[] == orig(::Gain[])
::Info == orig(::Info)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] == orig(::Info[])

::Bar == orig(::Bar)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::Bar[] == orig(::Bar[])
::UnknownRate == orig(::UnknownRate)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::ClassSum[] (lexically)
::UnknownRate[] <= ::ClassSum[] (lexically)
::UnknownRate[] == orig(::UnknownRate[])
::UnknownRate[] < orig(::ClassSum[]) (lexically)
::UnknownRate[] <= orig(::ClassSum[]) (lexically)
::SplitGain == orig(::SplitGain)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitGain[] < ::ClassSum[] (lexically)
::SplitGain[] <= ::ClassSum[] (lexically)
::SplitGain[] == orig(::SplitGain[])
::SplitGain[] < orig(::ClassSum[]) (lexically)
::SplitGain[] <= orig(::ClassSum[]) (lexically)
::SplitInfo == orig(::SplitInfo)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::ClassSum[] (lexically)
::SplitInfo[] <= ::ClassSum[] (lexically)
::SplitInfo[] == orig(::SplitInfo[])
::SplitInfo[] < orig(::ClassSum[]) (lexically)
::SplitInfo[] <= orig(::ClassSum[]) (lexically)
::Slice1 == orig(::Slice1)
::Slice1[] == ::Slice2[]
::Slice1[] < ::ClassSum[] (lexically)
::Slice1[] <= ::ClassSum[] (lexically)
::Slice1[] == orig(::Slice1[])
::Slice1[] == orig(::Slice2[])
::Slice1[] < orig(::ClassSum[]) (lexically)
::Slice1[] <= orig(::ClassSum[]) (lexically)
::Slice2 == orig(::Slice2)
::Subset == orig(::Subset)
::Subset[] == orig(::Subset[])
::Subsets == orig(::Subsets)
::Subsets[] == orig(::Subsets[])
::PossibleValues == orig(::PossibleValues)
::Changed == orig(::Changed)
::Val == orig(::Val)
::Val[] == orig(::Val[])
::Dev == orig(::Dev)
::Dev[] > ::ClassSum[] (lexically)
::Dev[] >= ::ClassSum[] (lexically)
::Dev[] == orig(::Dev[])
::Dev[] > orig(::ClassSum[]) (lexically)
::Dev[] >= orig(::ClassSum[]) (lexically)
::CVals == orig(::CVals)
::Delimiter == orig(::Delimiter)
::Subdef == orig(::Subdef)
::TRf == orig(::TRf)
::optind == orig(::optind)

::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::ClassSum != null
::ClassSum[] elements >= 0.0
::Delimiter == -1
::Subdef has only one value
::TRf != null
::optind == 3
return >= 0
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::Changed
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxAtt > return
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::Changed
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxClass >= return
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::Changed
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::Changed
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxItem > return
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus == ::Tested
::SpecialStatus < ::Fn
::SpecialStatus == orig(::SpecialStatus)
::SpecialStatus == orig(::Tested)
::PossibleValues in ::AttValName[]
::FileName < ::Fn
::FileName == ::optarg
::FileName == orig(::FileName)
::FileName == orig(::optarg)
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::VERBOSITY <= ::Changed

::VERBOSITY <= return
::TRIALS >= ::Changed
::MINOBJS > ::Changed
::CF in ::Dev[]
::Weight[] > ::SplitGain[] (elementwise)
::Freq[][0] <= ::ValFreq[] (elementwise)
::Gain[] <= ::Info[] (elementwise)
::SplitGain[] < ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
::Changed > ::Delimiter
::Changed < ::optind
::Delimiter < return
::Fn == orig(::Fn)
==========
..Classify():::ENTER
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Subtree
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::TargetClassFreq == ::ThreshErrs
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::ClassSum[] (lexically)
::Weight[] >= ::ClassSum[] (lexically)
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)

::Info[] >= ::UnknownRate[] (lexically)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::Bar[] > ::ClassSum[] (lexically)
::Bar[] >= ::ClassSum[] (lexically)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitGain[] < ::ClassSum[] (lexically)
::SplitGain[] <= ::ClassSum[] (lexically)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::Slice1[] == ::Slice2[]
::Dev[] > ::ClassSum[] (lexically)
::Dev[] >= ::ClassSum[] (lexically)
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Gain[] elements != 0
::Gain[] elements >= -0.0010
::Info != null
::Info[] elements >= 0.0
::Bar != null
::Bar[] elements >= 0.0
::UnknownRate != null
::UnknownRate[] elements >= 0.0
::SplitGain != null
::SplitGain[] elements != 0
::SplitGain[] elements >= -0.0010
::SplitInfo != null
::SplitInfo[] elements >= 0.0
::Slice1 != null
::Slice1[] elements == 0.0
::Slice1[] one of [0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::Slice2 != null
::Subset != null
::Subsets != null
::Subsets[] elements == 0
::PossibleValues == null
::Changed one of 0, 1
::Val has only one value
::Val[] == [0.0, 0.0010, 0.0050, 0.01, 0.05, 0.1, 0.2, 0.4, 1.0]
::Val[] elements <= 1.0
::Val[] elements >= 0.0
::Val[] sorted by <
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::ClassSum != null
::ClassSum[] elements >= 0.0
::Delimiter == -1
::Subdef has only one value
::TRf != null
::optind == 3
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::Changed
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::Changed
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::Changed
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem

::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::Changed
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus == ::Tested
::SpecialStatus < ::Fn
::PossibleValues in ::AttValName[]
::FileName < ::Fn
::FileName == ::optarg
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::VERBOSITY <= ::Changed
::TRIALS >= ::Changed
::MINOBJS > ::Changed
::CF in ::Dev[]
::Weight[] > ::SplitGain[] (elementwise)
::Freq[][0] <= ::ValFreq[] (elementwise)
::Gain[] <= ::Info[] (elementwise)
::SplitGain[] < ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
::Changed > ::Delimiter
::Changed < ::optind
==========
..Classify()::EXIT
::MaxAtt == orig(::MaxAtt)
::MaxClass == orig(::MaxClass)
::MaxDiscrVal == orig(::MaxDiscrVal)
::MaxItem == orig(::MaxItem)
::Item == orig(::Item)
::Item[] == orig(::Item[])
::MaxAttVal == orig(::MaxAttVal)
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::MaxAttVal[] == orig(::MaxAttVal[])
::ClassName == orig(::ClassName)
::AttName == orig(::AttName)
::AttValName == orig(::AttValName)
::AttValName[] == orig(::AttValName[])
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Subtree
::VERBOSITY == orig(::VERBOSITY)
::VERBOSITY == orig(::SUBSET)
::VERBOSITY == orig(::UNSEENS)
::VERBOSITY == orig(::PROBTHRESH)
::VERBOSITY == orig(::WINDOW)
::VERBOSITY == orig(::INCREMENT)
::VERBOSITY == orig(::MultiVal)
::VERBOSITY == orig(::Subtree)
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown

::TRIALS == orig(::TRIALS)
::TRIALS == orig(::GAINRATIO)
::TRIALS == orig(::BATCH)
::TRIALS == orig(::AllKnown)
::MINOBJS == orig(::MINOBJS)
::CF == orig(::CF)
::Pruned == orig(::Pruned)
::Pruned[] == orig(::Pruned[])
::TargetClassFreq == ::ThreshErrs
::TargetClassFreq == orig(::TargetClassFreq)
::TargetClassFreq == orig(::ThreshErrs)
::Raw == orig(::Raw)
::Raw[] == orig(::Raw[])
::Weight == orig(::Weight)
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::ClassSum[] (lexically)
::Weight[] >= ::ClassSum[] (lexically)
::Weight[] == orig(::Weight[])
::Weight[] > orig(::ClassSum[]) (lexically)
::Weight[] >= orig(::ClassSum[]) (lexically)
::Freq == orig(::Freq)
::Freq[] == orig(::Freq[])
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::Freq[][0] == orig(::Freq[][0])
::ValFreq == orig(::ValFreq)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ValFreq[] == orig(::ValFreq[])
::ClassFreq == orig(::ClassFreq)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] == orig(::ClassFreq[])
::Gain == orig(::Gain)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Gain[] == orig(::Gain[])
::Info == orig(::Info)
::Info[] >= ::UnknownRate[] (lexically)

::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] == orig(::Info[])
::Bar == orig(::Bar)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::Bar[] == orig(::Bar[])
::Bar[] > orig(::ClassSum[]) (lexically)
::Bar[] >= orig(::ClassSum[]) (lexically)
::UnknownRate == orig(::UnknownRate)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::ClassSum[] (lexically)
::UnknownRate[] <= ::ClassSum[] (lexically)
::UnknownRate[] == orig(::UnknownRate[])
::SplitGain == orig(::SplitGain)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitGain[] < ::ClassSum[] (lexically)
::SplitGain[] <= ::ClassSum[] (lexically)
::SplitGain[] == orig(::SplitGain[])
::SplitGain[] < orig(::ClassSum[]) (lexically)
::SplitGain[] <= orig(::ClassSum[]) (lexically)
::SplitInfo == orig(::SplitInfo)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::ClassSum[] (lexically)
::SplitInfo[] <= ::ClassSum[] (lexically)
::SplitInfo[] == orig(::SplitInfo[])
::Slice1 == orig(::Slice1)
::Slice1[] == ::Slice2[]
::Slice1[] < ::ClassSum[] (lexically)
::Slice1[] <= ::ClassSum[] (lexically)
::Slice1[] == orig(::Slice1[])
::Slice1[] == orig(::Slice2[])
::Slice2 == orig(::Slice2)
::Subset == orig(::Subset)
::Subset[] == orig(::Subset[])
::Subsets == orig(::Subsets)
::Subsets[] == orig(::Subsets[])
::PossibleValues == orig(::PossibleValues)
::Changed == orig(::Changed)
::Val == orig(::Val)
::Val[] == orig(::Val[])
::Dev == orig(::Dev)
::Dev[] > ::ClassSum[] (lexically)
::Dev[] >= ::ClassSum[] (lexically)
::Dev[] == orig(::Dev[])
::Dev[] > orig(::ClassSum[]) (lexically)
::Dev[] >= orig(::ClassSum[]) (lexically)
::CVals == orig(::CVals)
::ClassSum == orig(::ClassSum)
::ClassSum[] > orig(::ClassSum[]) (lexically)
::Delimiter == orig(::Delimiter)

::Subdef == orig(::Subdef)
::TRf == orig(::TRf)
::optind == orig(::optind)
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Val[] elements <= 1.0
::Val[] elements >= 0.0
::Val[] sorted by <
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::ClassSum != null
::ClassSum[] elements >= 0.0
::Delimiter == -1
::Subdef has only one value
::TRf != null
::optind == 3
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::Changed
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxAtt < return
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::Changed
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxClass < return
return
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::Changed
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxDiscrVal < return
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::Changed
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxItem < return
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::MaxAttVal[] elements < return
::SpecialStatus < ::FileName
::SpecialStatus == ::Tested
::SpecialStatus < ::Fn
::SpecialStatus == orig(::SpecialStatus)
::SpecialStatus == orig(::Tested)
::PossibleValues in ::AttValName[]
::FileName < ::Fn
::FileName == ::optarg

::FileName == orig(::FileName)
::FileName == orig(::optarg)
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::VERBOSITY <= ::Changed
::VERBOSITY < return
::TRIALS >= ::Changed
::TRIALS < return
::MINOBJS > ::Changed
::MINOBJS < return
return
::CF in ::Dev[]
::Weight[] > ::SplitGain[] (elementwise)
::Freq[][0] <= ::ValFreq[] (elementwise)
::Gain[] <= ::Info[] (elementwise)
::SplitGain[] < ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
::Changed > ::Delimiter
::Changed < ::optind
::Changed < return
::ClassSum[] >= orig(::ClassSum[]) (elementwise)
::Delimiter < return
::Fn == orig(::Fn)
::optind < return
==========
..ComputeGain()::ENTER
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::Pruned[] == ::Raw[]
::TargetClassFreq == ::ThreshErrs
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitGain[] (lexically)
::Weight[] >= ::SplitGain[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::Val[] (lexically)
::Weight[] >= ::Val[] (lexically)
::Weight[] < ::Dev[] (lexically)
::Weight[] <= ::Dev[] (lexically)
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::ValFreq[] > ::UnknownRate[] (lexically)
::ValFreq[] >= ::UnknownRate[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ValFreq[] > ::Slice1[] (lexically)
::ValFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] > ::UnknownRate[] (lexically)

::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Dev[] (lexically)
::Gain[] <= ::Dev[] (lexically)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] < ::Dev[] (lexically)
::Info[] <= ::Dev[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::Val[] (lexically)
::UnknownRate[] <= ::Val[] (lexically)
::UnknownRate[] < ::Dev[] (lexically)
::UnknownRate[] <= ::Dev[] (lexically)
::SplitGain[] < ::Val[] (lexically)
::SplitGain[] <= ::Val[] (lexically)
::SplitGain[] < ::Dev[] (lexically)
::SplitGain[] <= ::Dev[] (lexically)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::Dev[] (lexically)
::SplitInfo[] <= ::Dev[] (lexically)
::Slice1[] == ::Slice2[]
::CVals == ::ClassSum
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::ClassFreq[] elements >= 0.0
::Gain != null
::Gain[] elements <= 1.0
::Gain[] elements >= -0.0010
::Info != null
::Info[] elements >= 0.0
::Bar != null
::Bar[] elements >= 0.0
::UnknownRate != null
::UnknownRate[] elements >= 0.0
::SplitGain != null
::SplitGain[] elements >= -0.0010
::SplitInfo != null
::SplitInfo[] elements >= 0.0
::Slice1 != null
::Slice1[] elements == 0.0
::Slice1[] one of [0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::Slice2 != null
::Subset != null
::Subsets != null
::Subsets[] elements == 0
::PossibleValues == null
::Val has only one value
::Val[] == [0.0, 0.0010, 0.0050, 0.01, 0.05, 0.1, 0.2, 0.4, 1.0]
::Val[] elements <= 1.0
::Val[] elements >= 0.0
::Val[] sorted by <
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::Delimiter == -1
::Subdef has only one value
::TRf == null
::optind == 3
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::AllKnown
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::AllKnown
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::AllKnown
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS

::MaxItem > ::MINOBJS
::MaxItem > ::AllKnown
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus <= ::Tested
::SpecialStatus == ::Fn
::PossibleValues in ::AttValName[]
::FileName > ::Tested
::FileName == ::optarg
::VERBOSITY <= ::AllKnown
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::TRIALS >= ::AllKnown
::MINOBJS > ::AllKnown
::Weight[] elements > ::CF
::AllKnown > ::Delimiter
::AllKnown < ::optind
::Freq[][0] <= ::ValFreq[] (elementwise)
::Gain[] <= ::Info[] (elementwise)
::SplitGain[] <= ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
==========
..ComputeGain()::EXIT
::MaxAtt == orig(::MaxAtt)
::MaxClass == orig(::MaxClass)
::MaxDiscrVal == orig(::MaxDiscrVal)
::MaxItem == orig(::MaxItem)
::Item == orig(::Item)
::Item[] == orig(::Item[])
::MaxAttVal == orig(::MaxAttVal)
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::MaxAttVal[] == orig(::MaxAttVal[])
::ClassName == orig(::ClassName)
::AttName == orig(::AttName)
::AttValName == orig(::AttValName)
::AttValName[] == orig(::AttValName[])
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::VERBOSITY == orig(::VERBOSITY)
::VERBOSITY == orig(::SUBSET)
::VERBOSITY == orig(::UNSEENS)
::VERBOSITY == orig(::PROBTHRESH)
::VERBOSITY == orig(::WINDOW)
::VERBOSITY == orig(::INCREMENT)
::VERBOSITY == orig(::MultiVal)
::VERBOSITY == orig(::Changed)
::VERBOSITY == orig(::Subtree)
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == orig(::TRIALS)
::TRIALS == orig(::GAINRATIO)

::TRIALS == orig(::BATCH)
::MINOBJS == orig(::MINOBJS)
::Pruned == orig(::Pruned)
::Pruned[] == ::Raw[]
::Pruned[] == orig(::Pruned[])
::Pruned[] == orig(::Raw[])
::AllKnown == orig(::AllKnown)
::TargetClassFreq == ::ThreshErrs
::TargetClassFreq == orig(::TargetClassFreq)
::TargetClassFreq == orig(::ThreshErrs)
::Raw == orig(::Raw)
::Weight == orig(::Weight)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitGain[] (lexically)
::Weight[] >= ::SplitGain[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::Val[] (lexically)
::Weight[] >= ::Val[] (lexically)
::Weight[] < ::Dev[] (lexically)
::Weight[] <= ::Dev[] (lexically)
::Weight[] == orig(::Weight[])
::Freq == orig(::Freq)
::Freq[] == orig(::Freq[])
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::Freq[][0] == orig(::Freq[][0])
::ValFreq == orig(::ValFreq)
::ValFreq[] > ::UnknownRate[] (lexically)
::ValFreq[] >= ::UnknownRate[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ValFreq[] > ::Slice1[] (lexically)
::ValFreq[] >= ::Slice1[] (lexically)
::ValFreq[] == orig(::ValFreq[])
::ClassFreq == orig(::ClassFreq)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] == orig(::ClassFreq[])
::Gain == orig(::Gain)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Dev[] (lexically)
::Gain[] <= ::Dev[] (lexically)
::Gain[] == orig(::Gain[])
::Info == orig(::Info)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] < ::Dev[] (lexically)
::Info[] <= ::Dev[] (lexically)
::Info[] == orig(::Info[])
::Bar == orig(::Bar)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::Bar[] == orig(::Bar[])

::UnknownRate == orig(::UnknownRate)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::Val[] (lexically)
::UnknownRate[] <= ::Val[] (lexically)
::UnknownRate[] < ::Dev[] (lexically)
::UnknownRate[] <= ::Dev[] (lexically)
::UnknownRate[] == orig(::UnknownRate[])
::SplitGain == orig(::SplitGain)
::SplitGain[] < ::Val[] (lexically)
::SplitGain[] <= ::Val[] (lexically)
::SplitGain[] < ::Dev[] (lexically)
::SplitGain[] <= ::Dev[] (lexically)
::SplitGain[] == orig(::SplitGain[])
::SplitInfo == orig(::SplitInfo)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::Dev[] (lexically)
::SplitInfo[] <= ::Dev[] (lexically)
::SplitInfo[] == orig(::SplitInfo[])
::Slice1 == orig(::Slice1)
::Slice1[] == ::Slice2[]
::Slice1[] == orig(::Slice1[])
::Slice1[] == orig(::Slice2[])
::Slice2 == orig(::Slice2)
::Subset == orig(::Subset)
::Subset[] == orig(::Subset[])
::Subsets == orig(::Subsets)
::Subsets[] == orig(::Subsets[])
::PossibleValues == orig(::PossibleValues)
::Val == orig(::Val)
::Val[] == orig(::Val[])
::Dev == orig(::Dev)
::Dev[] == orig(::Dev[])
::CVals == ::ClassSum
::CVals == orig(::CVals)
::CVals == orig(::ClassSum)
::Delimiter == orig(::Delimiter)
::Subdef == orig(::Subdef)
::TRf == orig(::TRf)
::optind == orig(::optind)
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Pruned[] elements == null
::AllKnown one of 0, 1
::TargetClassFreq == null
::Raw != null
::Weight != null
::Weight[] elements <= 1.00000011
::Freq != null
::Freq[] elements != null
::ValFreq != null
::ValFreq[] elements >= 0.0
::ClassFreq != null
::ClassFreq[] elements >= 0.0
::Gain != null
::Gain[] elements <= 1.0
::Gain[] elements >= -0.0010
::Info != null
::Info[] elements >= 0.0
::Bar != null
::Bar[] elements >= 0.0
::UnknownRate != null
::UnknownRate[] elements >= 0.0
::SplitGain != null
::SplitGain[] elements >= -0.0010
::SplitInfo != null
::SplitInfo[] elements >= 0.0
::Slice1 != null
::Slice1[] elements == 0.0
::Slice1[] one of [0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::Slice2 != null
::Subset != null
::Subsets != null
::Subsets[] elements == 0
::PossibleValues == null
::Val has only one value
::Val[] == [0.0, 0.0010, 0.0050, 0.01, 0.05, 0.1, 0.2, 0.4, 1.0]
::Val[] elements <= 1.0
::Val[] elements >= 0.0
::Val[] sorted by <
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::Delimiter == -1
::Subdef has only one value
::TRf == null
::optind == 3
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::AllKnown
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::AllKnown
::MaxClass > ::Delimiter
::MaxClass != ::optind

::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::AllKnown
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::AllKnown
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus <= ::Tested
::SpecialStatus == ::Fn
::SpecialStatus == orig(::SpecialStatus)
::SpecialStatus == orig(::Fn)
::PossibleValues in ::AttValName[]
::FileName > ::Tested
::FileName == ::optarg
::FileName == orig(::FileName)
::FileName == orig(::optarg)
::VERBOSITY <= ::AllKnown
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::TRIALS >= ::AllKnown
::MINOBJS > ::AllKnown
::CF in ::Dev[]
::AllKnown > ::Delimiter
::AllKnown < ::optind
::Weight[] elements > orig(::CF)
::Freq[][0] <= ::ValFreq[] (elementwise)
::Gain[] <= ::Info[] (elementwise)
::Tested == orig(::Tested)
::SplitGain[] <= ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
==========
..Evaluate()::ENTER
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::TargetClassFreq == ::ThreshErrs
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)

::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Freq[][0] < ::ValFreq[] (lexically)
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::UnknownRate[] (lexically)
::ValFreq[] >= ::UnknownRate[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ValFreq[] > ::SplitInfo[] (lexically)
::ValFreq[] >= ::SplitInfo[] (lexically)
::ValFreq[] > ::Slice1[] (lexically)
::ValFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] > ::Gain[] (lexically)
::ClassFreq[] >= ::Gain[] (lexically)
::ClassFreq[] > ::Info[] (lexically)
::ClassFreq[] >= ::Info[] (lexically)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] < ::SplitInfo[] (lexically)
::UnknownRate[] <= ::SplitInfo[] (lexically)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::Slice1[] == ::Slice2[]
::CVals == ::ClassSum
::MaxAtt one of 3, 8, 12
::MaxClass one of 1, 2
::MaxDiscrVal one of 2, 3
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

0, 0, 0,

::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::VERBOSITY == orig(::VERBOSITY)
::VERBOSITY == orig(::SUBSET)
::VERBOSITY == orig(::UNSEENS)
::VERBOSITY == orig(::PROBTHRESH)
::VERBOSITY == orig(::WINDOW)
::VERBOSITY == orig(::INCREMENT)
::VERBOSITY == orig(::MultiVal)
::VERBOSITY == orig(::Changed)
::VERBOSITY == orig(::Subtree)
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::TRIALS == return
::TRIALS == orig(::TRIALS)
::TRIALS == orig(::GAINRATIO)
::TRIALS == orig(::BATCH)
::TRIALS == orig(::AllKnown)
::MINOBJS == orig(::MINOBJS)
::CF == orig(::CF)
::Pruned == orig(::Pruned)
::Pruned[] == orig(::Pruned[])
::TargetClassFreq == ::ThreshErrs
::TargetClassFreq == orig(::TargetClassFreq)
::TargetClassFreq == orig(::ThreshErrs)
::Raw == orig(::Raw)
::Raw[] == orig(::Raw[])
::Weight == orig(::Weight)
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::ClassSum[] (lexically)
::Weight[] >= ::ClassSum[] (lexically)
::Weight[] == orig(::Weight[])
::Freq == orig(::Freq)
::Freq[] == orig(::Freq[])
::Freq[][0] < ::ValFreq[] (lexically)
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::Freq[][0] == orig(::Freq[][0])
::ValFreq == orig(::ValFreq)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::UnknownRate[] (lexically)
::ValFreq[] >= ::UnknownRate[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ValFreq[] > ::SplitInfo[] (lexically)

::ValFreq[] >= ::SplitInfo[] (lexically)
::ValFreq[] > ::Slice1[] (lexically)
::ValFreq[] >= ::Slice1[] (lexically)
::ValFreq[] == orig(::ValFreq[])
::ClassFreq == orig(::ClassFreq)
::ClassFreq[] > ::Gain[] (lexically)
::ClassFreq[] >= ::Gain[] (lexically)
::ClassFreq[] > ::Info[] (lexically)
::ClassFreq[] >= ::Info[] (lexically)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] == orig(::ClassFreq[])
::Gain == orig(::Gain)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Gain[] == orig(::Gain[])
::Info == orig(::Info)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] == orig(::Info[])
::Bar == orig(::Bar)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::Bar[] == orig(::Bar[])
::UnknownRate == orig(::UnknownRate)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] < ::SplitInfo[] (lexically)
::UnknownRate[] <= ::SplitInfo[] (lexically)
::UnknownRate[] < ::ClassSum[] (lexically)
::UnknownRate[] <= ::ClassSum[] (lexically)
::UnknownRate[] == orig(::UnknownRate[])
::SplitGain == orig(::SplitGain)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitGain[] < ::ClassSum[] (lexically)
::SplitGain[] <= ::ClassSum[] (lexically)
::SplitGain[] == orig(::SplitGain[])
::SplitInfo == orig(::SplitInfo)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::ClassSum[] (lexically)
::SplitInfo[] <= ::ClassSum[] (lexically)
::SplitInfo[] == orig(::SplitInfo[])
::Slice1 == orig(::Slice1)
::Slice1[] == ::Slice2[]
::Slice1[] < ::ClassSum[] (lexically)
::Slice1[] <= ::ClassSum[] (lexically)
::Slice1[] == orig(::Slice1[])

::Slice1[] == orig(::Slice2[])
::Slice2 == orig(::Slice2)
::Subset == orig(::Subset)
::Subset[] == orig(::Subset[])
::Subsets == orig(::Subsets)
::Subsets[] == orig(::Subsets[])
::PossibleValues == orig(::PossibleValues)
::Val == orig(::Val)
::Val[] == orig(::Val[])
::Dev == orig(::Dev)
::Dev[] == orig(::Dev[])
::CVals == orig(::CVals)
::CVals == orig(::ClassSum)
::Delimiter == orig(::Delimiter)
::Subdef == orig(::Subdef)
::TRf == orig(::TRf)
::optind == orig(::optind)
::MaxAtt one of 3, 8, 12
::MaxClass one of 1, 2
::MaxDiscrVal one of 2, 3
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::PossibleValues in ::AttValName[]
::FileName == ::optarg
::FileName == orig(::FileName)
::FileName == orig(::optarg)
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::CF in ::Dev[]
::PossibleValues in ::Subset[]
::Fn == orig(::Fn)
==========
..FormTree():::ENTER
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::Pruned[] == ::Raw[]
::TargetClassFreq == ::ThreshErrs
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitGain[] (lexically)
::Weight[] >= ::SplitGain[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::Val[] (lexically)
::Weight[] >= ::Val[] (lexically)
::Weight[] < ::Dev[] (lexically)
::Weight[] <= ::Dev[] (lexically)
::Freq[][0] < ::Dev[] (lexically)
::Freq[][0] <= ::Dev[] (lexically)
::Gain[] < ::Dev[] (lexically)
::Gain[] <= ::Dev[] (lexically)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] < ::Dev[] (lexically)
::Info[] <= ::Dev[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::Val[] (lexically)
::UnknownRate[] <= ::Val[] (lexically)
::UnknownRate[] < ::Dev[] (lexically)
::UnknownRate[] <= ::Dev[] (lexically)
::SplitGain[] < ::Val[] (lexically)
::SplitGain[] <= ::Val[] (lexically)
::SplitGain[] < ::Dev[] (lexically)
::SplitGain[] <= ::Dev[] (lexically)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::Dev[] (lexically)
::SplitInfo[] <= ::Dev[] (lexically)
::Slice1[] == ::Slice2[]
::CVals == ::ClassSum
::MaxClass one of 1, 2, 4

::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Subdef has only one value
::TRf == null
::optind == 3
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::AllKnown
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::AllKnown
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::AllKnown
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::AllKnown
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus <= ::Tested
::SpecialStatus == ::Fn
::PossibleValues in ::AttValName[]
::FileName > ::Tested
::FileName == ::optarg
::VERBOSITY <= ::AllKnown
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::TRIALS >= ::AllKnown
::MINOBJS > ::AllKnown
::CF in ::Dev[]
::AllKnown > ::Delimiter
::AllKnown < ::optind
::Gain[] <= ::Info[] (elementwise)
::SplitGain[] <= ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
==========
..FormTree()::EXIT
::MaxAtt == orig(::MaxAtt)
::MaxClass == orig(::MaxClass)
::MaxDiscrVal == orig(::MaxDiscrVal)
::MaxItem == orig(::MaxItem)
::Item == orig(::Item)
::MaxAttVal == orig(::MaxAttVal)

::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::MaxAttVal[] == orig(::MaxAttVal[])
::ClassName == orig(::ClassName)
::AttName == orig(::AttName)
::AttValName == orig(::AttValName)
::AttValName[] == orig(::AttValName[])
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::VERBOSITY == orig(::VERBOSITY)
::VERBOSITY == orig(::SUBSET)
::VERBOSITY == orig(::UNSEENS)
::VERBOSITY == orig(::PROBTHRESH)
::VERBOSITY == orig(::WINDOW)
::VERBOSITY == orig(::INCREMENT)
::VERBOSITY == orig(::MultiVal)
::VERBOSITY == orig(::Changed)
::VERBOSITY == orig(::Subtree)
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == orig(::TRIALS)
::TRIALS == orig(::GAINRATIO)
::TRIALS == orig(::BATCH)
::MINOBJS == orig(::MINOBJS)
::Pruned == orig(::Pruned)
::Pruned[] == ::Raw[]
::Pruned[] == orig(::Pruned[])
::Pruned[] == orig(::Raw[])
::AllKnown == orig(::AllKnown)
::TargetClassFreq == ::ThreshErrs
::TargetClassFreq == orig(::TargetClassFreq)
::TargetClassFreq == orig(::ThreshErrs)
::Raw == orig(::Raw)
::Weight == orig(::Weight)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitGain[] (lexically)
::Weight[] >= ::SplitGain[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Weight[] > ::Val[] (lexically)
::Weight[] >= ::Val[] (lexically)
::Weight[] < ::Dev[] (lexically)
::Weight[] <= ::Dev[] (lexically)
::Weight[] > orig(::UnknownRate[]) (lexically)
::Weight[] >= orig(::UnknownRate[]) (lexically)
::Weight[] > orig(::SplitGain[]) (lexically)
::Weight[] >= orig(::SplitGain[]) (lexically)
::Weight[] > orig(::SplitInfo[]) (lexically)
::Weight[] >= orig(::SplitInfo[]) (lexically)
::Freq == orig(::Freq)
::Freq[] == orig(::Freq[])
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)

::Freq[][0] >= ::SplitGain[] (lexically)
::Freq[][0] < ::Dev[] (lexically)
::Freq[][0] <= ::Dev[] (lexically)
::ValFreq == orig(::ValFreq)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ClassFreq == orig(::ClassFreq)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::ClassFreq[] > orig(::UnknownRate[]) (lexically)
::ClassFreq[] >= orig(::UnknownRate[]) (lexically)
::ClassFreq[] > orig(::SplitGain[]) (lexically)
::ClassFreq[] >= orig(::SplitGain[]) (lexically)
::Gain == orig(::Gain)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Gain[] < ::Dev[] (lexically)
::Gain[] <= ::Dev[] (lexically)
::Info == orig(::Info)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Info[] < ::Dev[] (lexically)
::Info[] <= ::Dev[] (lexically)
::Bar == orig(::Bar)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::Bar[] > ::Val[] (lexically)
::Bar[] >= ::Val[] (lexically)
::Bar[] > orig(::Freq[][0]) (lexically)
::Bar[] >= orig(::Freq[][0]) (lexically)
::Bar[] > orig(::ValFreq[]) (lexically)
::Bar[] >= orig(::ValFreq[]) (lexically)
::Bar[] > orig(::UnknownRate[]) (lexically)
::Bar[] >= orig(::UnknownRate[]) (lexically)
::Bar[] > orig(::SplitGain[]) (lexically)
::Bar[] >= orig(::SplitGain[]) (lexically)
::Bar[] > orig(::SplitInfo[]) (lexically)
::Bar[] >= orig(::SplitInfo[]) (lexically)
::UnknownRate == orig(::UnknownRate)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::UnknownRate[] < ::Val[] (lexically)
::UnknownRate[] <= ::Val[] (lexically)
::UnknownRate[] < ::Dev[] (lexically)
::UnknownRate[] <= ::Dev[] (lexically)
::UnknownRate[] < orig(::Weight[]) (lexically)

::UnknownRate[] <= orig(::Weight[]) (lexically)
::SplitGain == orig(::SplitGain)
::SplitGain[] < ::SplitInfo[] (lexically)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitGain[] < ::Val[] (lexically)
::SplitGain[] <= ::Val[] (lexically)
::SplitGain[] < ::Dev[] (lexically)
::SplitGain[] <= ::Dev[] (lexically)
::SplitGain[] < orig(::Weight[]) (lexically)
::SplitGain[] <= orig(::Weight[]) (lexically)
::SplitGain[] < orig(::Freq[][0]) (lexically)
::SplitGain[] <= orig(::Freq[][0]) (lexically)
::SplitGain[] < orig(::ValFreq[]) (lexically)
::SplitGain[] <= orig(::ValFreq[]) (lexically)
::SplitGain[] < orig(::ClassFreq[]) (lexically)
::SplitGain[] <= orig(::ClassFreq[]) (lexically)
::SplitGain[] < orig(::Info[]) (lexically)
::SplitGain[] <= orig(::Info[]) (lexically)
::SplitGain[] < orig(::Bar[]) (lexically)
::SplitGain[] <= orig(::Bar[]) (lexically)
::SplitGain[] < orig(::UnknownRate[]) (lexically)
::SplitGain[] <= orig(::UnknownRate[]) (lexically)
::SplitGain[] < orig(::SplitInfo[]) (lexically)
::SplitGain[] <= orig(::SplitInfo[]) (lexically)
::SplitInfo == orig(::SplitInfo)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::SplitInfo[] < ::Dev[] (lexically)
::SplitInfo[] <= ::Dev[] (lexically)
::SplitInfo[] < orig(::Weight[]) (lexically)
::SplitInfo[] <= orig(::Weight[]) (lexically)
::SplitInfo[] > orig(::SplitGain[]) (lexically)
::SplitInfo[] >= orig(::SplitGain[]) (lexically)
::Slice1 == orig(::Slice1)
::Slice1[] == ::Slice2[]
::Slice1[] < orig(::Weight[]) (lexically)
::Slice1[] <= orig(::Weight[]) (lexically)
::Slice1[] <= orig(::Info[]) (lexically)
::Slice1[] <= orig(::Bar[]) (lexically)
::Slice1[] <= orig(::UnknownRate[]) (lexically)
::Slice1[] < orig(::SplitInfo[]) (lexically)
::Slice1[] <= orig(::SplitInfo[]) (lexically)
::Slice1[] == orig(::Slice1[])
::Slice1[] == orig(::Slice2[])
::Slice2 == orig(::Slice2)
::Subset == orig(::Subset)
::Subset[] == orig(::Subset[])
::Subsets == orig(::Subsets)
::Subsets[] == orig(::Subsets[])
::PossibleValues == orig(::PossibleValues)
::Val == orig(::Val)
::Val[] < orig(::Weight[]) (lexically)
::Val[] <= orig(::Weight[]) (lexically)
::Val[] > orig(::UnknownRate[]) (lexically)
::Val[] >= orig(::UnknownRate[]) (lexically)
::Val[] > orig(::SplitGain[]) (lexically)
::Val[] >= orig(::SplitGain[]) (lexically)
::Val[] == orig(::Val[])
::Dev == orig(::Dev)
::Dev[] > orig(::Weight[]) (lexically)
::Dev[] >= orig(::Weight[]) (lexically)
::Dev[] > orig(::Freq[][0]) (lexically)

::Dev[] >= orig(::Freq[][0]) (lexically)
::Dev[] > orig(::Gain[]) (lexically)
::Dev[] >= orig(::Gain[]) (lexically)
::Dev[] > orig(::Info[]) (lexically)
::Dev[] >= orig(::Info[]) (lexically)
::Dev[] > orig(::UnknownRate[]) (lexically)
::Dev[] >= orig(::UnknownRate[]) (lexically)
::Dev[] > orig(::SplitGain[]) (lexically)
::Dev[] >= orig(::SplitGain[]) (lexically)
::Dev[] > orig(::SplitInfo[]) (lexically)
::Dev[] >= orig(::SplitInfo[]) (lexically)
::Dev[] == orig(::Dev[])
::CVals == ::ClassSum
::CVals == orig(::CVals)
::CVals == orig(::ClassSum)
::Delimiter == orig(::Delimiter)
::Subdef == orig(::Subdef)
::TRf == orig(::TRf)
::optind == orig(::optind)
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::Item != null
::MaxAttVal != null
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::SplitGain[] elements >= -0.0010
::SplitInfo != null
::SplitInfo[] elements >= 0.0
::Slice1 != null
::Slice1[] elements == 0.0
::Slice1[] one of [0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::Slice2 != null
::Subset != null
::Subsets != null
::Subsets[] elements == 0
::PossibleValues == null
::Val has only one value
::Val[] == [0.0, 0.0010, 0.0050, 0.01, 0.05, 0.1, 0.2, 0.4, 1.0]
::Val[] sorted by <
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Dev[] sorted by >
::CVals == null
::Delimiter == -1
::Subdef has only one value
::TRf == null
::optind == 3
return != null
::MaxAtt > ::MaxClass
::MaxAtt >= ::MaxDiscrVal
::MaxAtt < ::MaxItem
::MaxAttVal[] elements <= ::MaxAtt
::MaxAtt > ::VERBOSITY
::MaxAtt > ::TRIALS
::MaxAtt > ::MINOBJS
::MaxAtt > ::AllKnown
::MaxAtt > ::Delimiter
::MaxAtt >= ::optind
::MaxClass < ::MaxItem
::MaxClass > ::VERBOSITY
::MaxClass >= ::TRIALS
::MaxClass >= ::AllKnown
::MaxClass > ::Delimiter
::MaxClass != ::optind
::MaxDiscrVal < ::MaxItem
::MaxAttVal[] elements <= ::MaxDiscrVal
::MaxDiscrVal > ::VERBOSITY
::MaxDiscrVal > ::TRIALS
::MaxDiscrVal >= ::MINOBJS
::MaxDiscrVal > ::AllKnown
::MaxDiscrVal > ::Delimiter
::MaxDiscrVal <= ::optind
::MaxAttVal[] elements < ::MaxItem
::MaxItem > ::VERBOSITY
::MaxItem > ::TRIALS
::MaxItem > ::MINOBJS
::MaxItem > ::AllKnown
::MaxItem > ::Delimiter
::MaxItem > ::optind
::MaxAttVal[] elements >= ::VERBOSITY
::VERBOSITY in ::MaxAttVal[]
::MaxAttVal[] elements > ::Delimiter
::MaxAttVal[] elements <= ::optind
::SpecialStatus < ::FileName
::SpecialStatus <= ::Tested
::SpecialStatus == ::Fn
::SpecialStatus == orig(::SpecialStatus)

::SpecialStatus == orig(::Fn)
::PossibleValues in ::AttValName[]
::FileName > ::Tested
::FileName == ::optarg
::FileName == orig(::FileName)
::FileName == orig(::optarg)
::VERBOSITY <= ::AllKnown
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::TRIALS >= ::AllKnown
::MINOBJS > ::AllKnown
::CF in ::Dev[]
::CF <= orig(::CF)
::AllKnown > ::Delimiter
::AllKnown < ::optind
::Gain[] <= ::Info[] (elementwise)
::Tested == orig(::Tested)
::SplitGain[] <= ::SplitInfo[] (elementwise)
::PossibleValues in ::Subset[]
orig(::CF) in ::Dev[]
==========
..OneTree():::ENTER
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Changed
::VERBOSITY == ::Subtree
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::Pruned == ::Raw
::TargetClassFreq == ::ThreshErrs
::Weight == ::ValFreq
::Weight == ::ClassFreq
::Weight == ::Gain
::Weight == ::Info
::Weight == ::Bar
::Weight == ::UnknownRate
::Weight == ::SplitGain
::Weight == ::SplitInfo
::Weight == ::Slice1
::Weight == ::Slice2
::Weight == ::CVals
::Weight == ::ClassSum
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::Pruned == null
::TargetClassFreq == null
::Weight == null
::Freq == null
::Subset == null
::Subsets == null
::PossibleValues == null
::Val has only one value
::Val[] == [0.0, 0.0010, 0.0050, 0.01, 0.05, 0.1, 0.2, 0.4, 1.0]
::Dev has only one value
::Dev[] == [4.0, 3.08999991, 2.57999992, 2.32999992, 1.64999997, 1.27999997, 0.83999997, 0.25, 0.0]
::Dev[] elements >= 0.0
::Delimiter == -1
::Subdef has only one value
::TRf == null
::optind == 3
::VERBOSITY in ::MaxAttVal[]
::SpecialStatus == ::Fn
::PossibleValues in ::AttValName[]
::FileName == ::optarg
::CF in ::Dev[]
==========
..OneTree()::EXIT
::MaxAtt == orig(::MaxAtt)
::MaxClass == orig(::MaxClass)
::MaxDiscrVal == orig(::MaxDiscrVal)
::MaxItem == orig(::MaxItem)
::Item == orig(::Item)
::MaxAttVal == orig(::MaxAttVal)
::MaxAttVal[] > ::Subsets[] (lexically)
::MaxAttVal[] >= ::Subsets[] (lexically)
::MaxAttVal[] == orig(::MaxAttVal[])
::ClassName == orig(::ClassName)
::AttName == orig(::AttName)
::AttValName == orig(::AttValName)
::AttValName[] == orig(::AttValName[])
::VERBOSITY == ::SUBSET
::VERBOSITY == ::UNSEENS
::VERBOSITY == ::PROBTHRESH
::VERBOSITY == ::WINDOW
::VERBOSITY == ::INCREMENT
::VERBOSITY == ::MultiVal
::VERBOSITY == ::Subtree
::VERBOSITY == orig(::VERBOSITY)
::VERBOSITY == orig(::SUBSET)
::VERBOSITY == orig(::UNSEENS)
::VERBOSITY == orig(::PROBTHRESH)
::VERBOSITY == orig(::WINDOW)
::VERBOSITY == orig(::INCREMENT)
::VERBOSITY == orig(::MultiVal)
::VERBOSITY == orig(::Changed)
::VERBOSITY == orig(::Subtree)
::TRIALS == ::GAINRATIO
::TRIALS == ::BATCH
::TRIALS == ::AllKnown
::TRIALS == orig(::TRIALS)
::TRIALS == orig(::GAINRATIO)
::TRIALS == orig(::BATCH)
::TRIALS == orig(::AllKnown)
::MINOBJS == orig(::MINOBJS)
::CF == orig(::CF)
::TargetClassFreq == ::ThreshErrs
::TargetClassFreq == orig(::TargetClassFreq)

::TargetClassFreq == orig(::ThreshErrs)
::Weight[] > ::Freq[][0] (lexically)
::Weight[] >= ::Freq[][0] (lexically)
::Weight[] > ::ValFreq[] (lexically)
::Weight[] >= ::ValFreq[] (lexically)
::Weight[] > ::Gain[] (lexically)
::Weight[] >= ::Gain[] (lexically)
::Weight[] > ::UnknownRate[] (lexically)
::Weight[] >= ::UnknownRate[] (lexically)
::Weight[] > ::SplitInfo[] (lexically)
::Weight[] >= ::SplitInfo[] (lexically)
::Weight[] > ::Slice1[] (lexically)
::Weight[] >= ::Slice1[] (lexically)
::Freq[][0] < ::ClassFreq[] (lexically)
::Freq[][0] <= ::ClassFreq[] (lexically)
::Freq[][0] < ::Bar[] (lexically)
::Freq[][0] <= ::Bar[] (lexically)
::Freq[][0] > ::SplitGain[] (lexically)
::Freq[][0] >= ::SplitGain[] (lexically)
::ValFreq[] < ::Bar[] (lexically)
::ValFreq[] <= ::Bar[] (lexically)
::ValFreq[] > ::SplitGain[] (lexically)
::ValFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::UnknownRate[] (lexically)
::ClassFreq[] >= ::UnknownRate[] (lexically)
::ClassFreq[] > ::SplitGain[] (lexically)
::ClassFreq[] >= ::SplitGain[] (lexically)
::ClassFreq[] > ::SplitInfo[] (lexically)
::ClassFreq[] >= ::SplitInfo[] (lexically)
::ClassFreq[] > ::Slice1[] (lexically)
::ClassFreq[] >= ::Slice1[] (lexically)
::Gain[] < ::Info[] (lexically)
::Gain[] < ::Bar[] (lexically)
::Gain[] <= ::Bar[] (lexically)
::Info[] >= ::UnknownRate[] (lexically)
::Info[] > ::SplitGain[] (lexically)
::Info[] >= ::SplitGain[] (lexically)
::Info[] > ::Slice1[] (lexically)
::Info[] >= ::Slice1[] (lexically)
::Bar[] > ::UnknownRate[] (lexically)
::Bar[] >= ::UnknownRate[] (lexically)
::Bar[] > ::SplitGain[] (lexically)
::Bar[] >= ::SplitGain[] (lexically)
::Bar[] > ::SplitInfo[] (lexically)
::Bar[] >= ::SplitInfo[] (lexically)
::Bar[] > ::Slice1[] (lexically)
::Bar[] >= ::Slice1[] (lexically)
::UnknownRate[] > ::SplitGain[] (lexically)
::UnknownRate[] >= ::SplitGain[] (lexically)
::UnknownRate[] >= ::Slice1[] (lexically)
::SplitGain[] < ::Slice1[] (lexically)
::SplitGain[] <= ::Slice1[] (lexically)
::SplitInfo[] > ::Slice1[] (lexically)
::SplitInfo[] >= ::Slice1[] (lexically)
::Slice1[] == ::Slice2[]
::PossibleValues == orig(::PossibleValues)
::Val == orig(::Val)
::Val[] == orig(::Val[])
::Dev == orig(::Dev)
::Dev[] == orig(::Dev[])
::CVals == ::ClassSum
::CVals == orig(::Weight)
::CVals == orig(::ValFreq)

::CVals == orig(::ClassFreq)
::CVals == orig(::Gain)
::CVals == orig(::Info)
::CVals == orig(::Bar)
::CVals == orig(::UnknownRate)
::CVals == orig(::SplitGain)
::CVals == orig(::SplitInfo)
::CVals == orig(::Slice1)
::CVals == orig(::Slice2)
::CVals == orig(::CVals)
::CVals == orig(::ClassSum)
::Delimiter == orig(::Delimiter)
::Subdef == orig(::Subdef)
::optind == orig(::optind)
::MaxClass one of 1, 2, 4
::MaxDiscrVal one of 2, 3
::MaxAttVal[] elements one of 0, 2, 3
::MaxAttVal[] one of [0, 0, 0,

::FileName == orig(::FileName)
::FileName == orig(::optarg)
::Subsets[] elements == ::VERBOSITY
::VERBOSITY in ::Subsets[]
::CF in ::Dev[]
::PossibleValues in ::Subset[]

11.2 MartiRank

==========
..check_var():::ENTER
::n_ids == num_ids
::sublist_order[] == ::this_order[]
::sublist_order[] == ::best_order[]
::scores[].var == ::scores[].order[0]
::scores[].auc == ::scores[].weight
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::unknown_limit == unknown_limit
::prob_dist_flag == ::exti
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 9], [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99]
::order[] elements >= 0
::order[] sorted by <
::sublist_order != null
::sublist_order[] elements == 0
::sublist_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::gargc == 7
::gargv != null
::gargv[] == [./marti]
::gargv[] elements == "./marti"
::genvp != null
::genvp[] == [LESSOPEN=| /usr/bin/lesspipe
::genvp[] elements == "LESSOPEN=| /usr/bin/lesspipe
::tabula has only one value
::tabula != null
::tabula[] elements != null
::tabula[][0] elements >= 1.0
var_idx >= 0
::n_ids != ::n_names
::n_ids >= ::n_rounds
::rounds[] elements <= ::n_ids
::order[] elements < ::n_ids
::n_ids > ::topk
::n_ids > ::prob_dist_flag
::n_ids > ::gargc
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > var_idx
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::rounds[] > ::scores[].var (elementwise)
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::sublist_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::prob_dist_flag in ::sublist_order[]
::sublist_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::scores[].var
::scores[].var elements == ::prob_dist_flag
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= var_idx
==========
..check_var()::EXIT
::n_ids == orig(::n_ids)
::n_ids == orig(num_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] == ::this_order[]
::sublist_order[] == ::best_order[]
::sublist_order[] == orig(::sublist_order[])
::sublist_order[] == orig(::this_order[])
::sublist_order[] == orig(::best_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var == ::scores[].order[0]

::scores[].var == orig(::scores[].var)
::scores[].var == orig(::scores[].order[0])
::scores[].auc == ::scores[].weight
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].auc == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::min_score_ptr == orig(::min_score_ptr)
::this_order == orig(::this_order)
::best_order == orig(::best_order)
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::unknown_limit == orig(unknown_limit)
::prob_dist_flag == ::exti
::prob_dist_flag == return
::prob_dist_flag == orig(::prob_dist_flag)
::prob_dist_flag == orig(::exti)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 9], [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99]
::order[] elements >= 0
::order[] sorted by <
::sublist_order != null
::sublist_order[] elements == 0
::sublist_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0,

::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0,

::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements < ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==========
..compasc():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].var == orig(::scores[].var)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::scores[].order[0] == orig(::scores[].order[0])
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var == orig(::min_score_ptr[].var)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].auc == orig(::min_score_ptr[].auc)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order == orig(::min_score_ptr[].order)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)

::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::min_score_ptr[].order[0] == orig(::min_score_ptr[].order[0])
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::ignore_set[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements < ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::topk >= return
::prob_dist_flag <= ::exti
::gargc > return
==========
..compdesc()::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0

::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of ”a”, ”d”
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of ”a”, ”d”
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::n_ids > ::prob_dist_flag
::n_ids > ::gargc
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::scores[].var elements < ::n_names
::min_score_ptr[].var elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==========
..compdesc():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].var == orig(::scores[].var)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)

::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::scores[].order[0] == orig(::scores[].order[0])
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var == orig(::min_score_ptr[].var)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].auc == orig(::min_score_ptr[].auc)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order == orig(::min_score_ptr[].order)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::min_score_ptr[].order[0] == orig(::min_score_ptr[].order[0])
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null

::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of ”a”, ”d”
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of ”a”, ”d”
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::n_ids > ::prob_dist_flag
::n_ids > ::gargc
::n_ids > return
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::scores[].var elements < ::n_names
::min_score_ptr[].var elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_names > return
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::n_rounds > return
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::rounds[] elements >= return
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::topk >= return
::prob_dist_flag <= ::exti
::gargc > return
==========
..dleg():::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)

::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::last[] has only one value, of length 13
::gargc == 7
::gargv != null
::gargv[] == [./marti]
::gargv[] elements == "./marti"
::genvp != null
::genvp[] == [LESSOPEN=| /usr/bin/lesspipe
::genvp[] elements == "LESSOPEN=| /usr/bin/lesspipe
::tabula has only one value
::tabula != null
::tabula[] elements != null
::tabula[][0] elements >= 1.0
::exti >= 0
::n_ids != ::n_names
::n_ids >= ::n_rounds
::rounds[] elements <= ::n_ids
::order[] elements < ::n_ids
::sublist_order[] elements < ::n_ids
::scores[].order[0] elements < ::n_ids
::min_score_ptr[].order[0] elements < ::n_ids
::this_order[] elements < ::n_ids
::n_ids > ::topk
::n_ids > ::prob_dist_flag
::n_ids > ::gargc
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==========
..dleg()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])

::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var == orig(::scores[].var)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::scores[].order[0] == orig(::scores[].order[0])
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var == orig(::min_score_ptr[].var)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].auc == orig(::min_score_ptr[].auc)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order == orig(::min_score_ptr[].order)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::min_score_ptr[].order[0] == orig(::min_score_ptr[].order[0])
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)

::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::tabula != null
::tabula[] elements != null
::tabula[][0] elements >= 1.0
::exti >= 0
return one of -1, 0, 1
::n_ids != ::n_names
::n_ids >= ::n_rounds
::rounds[] elements <= ::n_ids
::order[] elements < ::n_ids
::sublist_order[] elements < ::n_ids
::scores[].order[0] elements < ::n_ids
::min_score_ptr[].order[0] elements < ::n_ids
::this_order[] elements < ::n_ids
::n_ids > ::topk
::n_ids > ::prob_dist_flag
::n_ids > ::gargc
::n_ids > return
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_names > return
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::n_rounds > return
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::rounds[] elements >= return
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::topk >= return
::prob_dist_flag <= ::exti
::gargc > return
==========
..insert_score():::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores == scores

::scores[] == scores[]
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].var == scores[].var
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == scores[].auc
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == scores[].weight
::scores[].order == scores[].order
::scores[].order[0] == scores[].order[0]
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var < num_scores[] (lexically)
::min_score_ptr[].var <= num_scores[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order == order
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == order[]
::exti == var
::n_ids == 100
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order has only one value
::order != null
::order[] elements >= 0
::sublist_order has only one value
::sublist_order != null
::sublist_order[] elements >= 0
::scores has only one value
::scores != null
::scores[] contains no nulls and has only one value, of length 10
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of ”a”, ”d”
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null

::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of ”a”, ”d”
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order has only one value
::this_order != null
::this_order[] elements >= 0
::best_order has only one value
::best_order != null
::best_order[] == [0, 0, 0,

::n_rounds in ::order[]
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::n_rounds > ::exti
num_scores[] elements <= ::n_rounds
::rounds[] > ::ignore_set[] (elementwise)
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::exti in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var >= ::ignore_set[] (elementwise)
::scores[].var elements >= ::prob_dist_flag
dir in ::scores[].dir
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] >= ::ignore_set[] (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::topk < bin_size
::prob_dist_flag <= ::exti
num_scores[] elements >= ::prob_dist_flag
::prob_dist_flag < bin_size
==========
..insert_score()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores == orig(scores)
::scores[] == scores[]
::scores[] == orig(::scores[])
::scores[] == orig(scores[])
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].var == scores[].var
::scores[].auc > ::scores[].weight (lexically)

::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == scores[].auc
::scores[].auc > orig(::min_score_ptr[].weight) (lexically)
::scores[].auc >= orig(::min_score_ptr[].weight) (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == scores[].weight
::scores[].weight == orig(::scores[].weight)
::scores[].weight < orig(::min_score_ptr[].auc) (lexically)
::scores[].weight <= orig(::min_score_ptr[].auc) (lexically)
::scores[].weight >= orig(::min_score_ptr[].weight) (lexically)
::scores[].weight == orig(scores[].weight)
::scores[].order == scores[].order
::scores[].order == orig(::scores[].order)
::scores[].order == orig(scores[].order)
::scores[].order[0] > ::best_order[] (lexically)
::scores[].order[0] >= ::best_order[] (lexically)
::scores[].order[0] > ::ignore_set[] (lexically)
::scores[].order[0] == scores[].order[0]
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var < num_scores[] (lexically)
::min_score_ptr[].var <= num_scores[] (lexically)
::min_score_ptr[].var >= orig(::scores[].var) (lexically)
::min_score_ptr[].var < orig(num_scores[]) (lexically)
::min_score_ptr[].var <= orig(num_scores[]) (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < orig(::scores[].auc) (lexically)
::min_score_ptr[].weight <= orig(::scores[].auc) (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order == orig(::this_order)
::this_order == orig(order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == order[]
::this_order[] == orig(::this_order[])
::this_order[] == orig(order[])
::best_order == orig(::best_order)
::best_order[] < num_scores[] (lexically)
::best_order[] <= num_scores[] (lexically)
::best_order[] < orig(::min_score_ptr[].var) (lexically)
::best_order[] <= orig(::min_score_ptr[].var) (lexically)
::best_order[] < orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] <= orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] < num_scores[] (lexically)
::ignore_set[] <= num_scores[] (lexically)

::ignore_set[] < orig(::min_score_ptr[].var) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].var) (lexically)
::ignore_set[] < orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] > orig(::scores[].auc) (lexically)
::tabula[][0] >= orig(::scores[].auc) (lexically)
::tabula[][0] > orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] >= orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] > orig(::min_score_ptr[].weight) (lexically)
::tabula[][0] >= orig(::min_score_ptr[].weight) (lexically)
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::exti == orig(var)
num_scores[] > orig(::min_score_ptr[].var) (lexically)
num_scores[] >= orig(::min_score_ptr[].var) (lexically)
::n_ids == 100
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order has only one value
::order != null
::order[] elements >= 0
::sublist_order has only one value
::sublist_order != null
::sublist_order[] elements >= 0
::scores has only one value
::scores != null
::scores[] contains no nulls and has only one value, of length 10
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order has only one value
::this_order != null
::this_order[] elements >= 0

::best_order has only one value
::best_order != null
::best_order[] == [0, 0, 0,

::n_rounds in ::order[]
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::n_rounds > ::exti
num_scores[] elements <= ::n_rounds
::n_rounds > return
orig(::scores[].var) elements < ::n_rounds
orig(::min_score_ptr[].var) elements < ::n_rounds
orig(num_scores[]) elements <= ::n_rounds
::rounds[] > ::ignore_set[] (elementwise)
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::rounds[] elements >= return
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::exti in ::order[]
return in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var >= ::ignore_set[] (elementwise)
::scores[].var elements >= ::prob_dist_flag
orig(dir) in ::scores[].dir
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].weight <= orig(::scores[].auc) (elementwise)
::scores[].order[0] >= ::ignore_set[] (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[] == orig(::min_score_ptr[]) (elementwise)
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].var >= orig(::min_score_ptr[].var) (elementwise)
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].auc >= orig(::min_score_ptr[].auc) (elementwise)
::min_score_ptr[].auc > orig(::min_score_ptr[].weight) (elementwise)
::min_score_ptr[].weight < orig(::min_score_ptr[].auc) (elementwise)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight) (elementwise)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)**2 (elementwise)
::min_score_ptr[].weight is the reverse of orig(::min_score_ptr[].weight)
orig(::min_score_ptr[].weight) == ::min_score_ptr[].weight**2 (elementwise)
::min_score_ptr[].order == orig(::min_score_ptr[].order) (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::ignore_set[] <= orig(::scores[].var) (elementwise)
::ignore_set[] <= orig(::scores[].order[0]) (elementwise)
num_scores[] elements >= ::topk
::topk >= return
::topk < orig(bin_size)
::prob_dist_flag <= ::exti
num_scores[] elements > ::prob_dist_flag
::prob_dist_flag <= return
orig(::scores[].var) elements >= ::prob_dist_flag
orig(::scores[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements >= ::prob_dist_flag
orig(::min_score_ptr[].order[0]) elements >= ::prob_dist_flag

orig(num_scores[]) elements >= ::prob_dist_flag
::prob_dist_flag < orig(bin_size)
::gargc > return
orig(::min_score_ptr[].var) elements <= ::exti
num_scores[] elements >= return
num_scores[] >= orig(num_scores[]) (elementwise)
return < orig(bin_size)
==========
..mergesort():::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null

::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::n_names > ::exti
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==========
..mergesort()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc >= orig(::scores[].auc) (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].weight < orig(::min_score_ptr[].auc) (lexically)
::scores[].weight <= orig(::min_score_ptr[].auc) (lexically)
::min_score_ptr == orig(::min_score_ptr)

::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < orig(::scores[].auc) (lexically)
::min_score_ptr[].weight <= orig(::scores[].auc) (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] < orig(::min_score_ptr[].var) (lexically)
::best_order[] <= orig(::min_score_ptr[].var) (lexically)
::best_order[] < orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] <= orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] < orig(::min_score_ptr[].var) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].var) (lexically)
::ignore_set[] < orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == return
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] > orig(::scores[].auc) (lexically)
::tabula[][0] >= orig(::scores[].auc) (lexically)
::tabula[][0] > orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] >= orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
orig(::scores[].var) elements < ::n_names
orig(::min_score_ptr[].var) elements < ::n_names
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
orig(::scores[].var) elements < ::n_rounds
orig(::min_score_ptr[].var) elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].weight <= orig(::scores[].auc) (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].weight < orig(::min_score_ptr[].auc) (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
orig(::scores[].var) elements >= ::prob_dist_flag
orig(::scores[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements >= ::prob_dist_flag
orig(::min_score_ptr[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements <= ::exti
==========
..sort_examples()::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)

::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::topk == unknown_meth
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::last[] has only one value, of length 13
::gargc == 7
::gargv != null
::gargv[] == [./marti]
::gargv[] elements == "./marti"
::genvp != null
::genvp[] == [LESSOPEN=| /usr/bin/lesspipe
::genvp[] elements == "LESSOPEN=| /usr/bin/lesspipe
::tabula has only one value
::tabula != null
::tabula[] elements != null
::tabula[][0] elements >= 1.0
::exti >= 0
base != null
nmemb >= 0
size == 4
cmp != null
::n_ids != ::n_names
::n_ids >= ::n_rounds
::rounds[] elements <= ::n_ids
::order[] elements < ::n_ids
::sublist_order[] elements < ::n_ids
::scores[].var elements < ::n_ids
::scores[].order[0] elements < ::n_ids
::min_score_ptr[].var elements < ::n_ids
::min_score_ptr[].order[0] elements < ::n_ids
::this_order[] elements < ::n_ids
::n_ids > ::topk
::n_ids > ::prob_dist_flag
::n_ids > ::gargc
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::scores[].var elements < ::n_names
::min_score_ptr[].var elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag

::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==========
..sort_examples():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].var == orig(::scores[].var)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::scores[].order[0] == orig(::scores[].order[0])
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var == orig(::min_score_ptr[].var)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].auc == orig(::min_score_ptr[].auc)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order == orig(::min_score_ptr[].order)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::min_score_ptr[].order[0] == orig(::min_score_ptr[].order[0])
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::best_order == orig(::best_order)
::best_order[] < orig(::this_order[]) (lexically)

::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] < orig(::this_order[]) (lexically)
::ignore_set[] <= orig(::this_order[]) (lexically)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::topk == orig(unknown_meth)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == return
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::prob_dist_flag == 0
::failures has only one value
::failures != null
::failures[] has only one value, of length 13
::last has only one value
::last != null
::last[] has only one value, of length 13
::gargc == 7
::gargv != null
::gargv[] == [./marti]
::gargv[] elements == "./marti"
::genvp != null
::genvp[] == [LESSOPEN=| /usr/bin/lesspipe
::genvp[] elements == "LESSOPEN=| /usr/bin/lesspipe
::tabula has only one value
::tabula != null
::tabula[] elements != null
::tabula[][0] elements >= 1.0
::exti >= 0
::n_ids != ::n_names
::n_ids >= ::n_rounds
::rounds[] elements <= ::n_ids
::order[] elements < ::n_ids
::sublist_order[] elements < ::n_ids
::scores[].var elements < ::n_ids
::scores[].order[0] elements < ::n_ids
::min_score_ptr[].var elements < ::n_ids
::min_score_ptr[].order[0] elements < ::n_ids
::this_order[] elements < ::n_ids
::n_ids > ::topk
::n_ids > ::prob_dist_flag
::n_ids > ::gargc
orig(::this_order[]) elements < ::n_ids
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::scores[].var elements < ::n_names
::min_score_ptr[].var elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag

::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::best_order[] <= orig(::this_order[]) (elementwise)
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
orig(::this_order[]) elements >= ::prob_dist_flag
==========
merge.c.insertionsort():::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null

::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==========
merge.c.insertionsort():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc >= orig(::scores[].auc) (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)

::scores[].weight == orig(::scores[].weight)
::scores[].weight < orig(::min_score_ptr[].auc) (lexically)
::scores[].weight <= orig(::min_score_ptr[].auc) (lexically)
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var >= orig(::scores[].var) (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < orig(::scores[].auc) (lexically)
::min_score_ptr[].weight <= orig(::scores[].auc) (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] < orig(::min_score_ptr[].var) (lexically)
::best_order[] <= orig(::min_score_ptr[].var) (lexically)
::best_order[] < orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] <= orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] < orig(::min_score_ptr[].var) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].var) (lexically)
::ignore_set[] < orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] > orig(::scores[].auc) (lexically)
::tabula[][0] >= orig(::scores[].auc) (lexically)
::tabula[][0] > orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] >= orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1

::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::min_score_ptr[].var elements < ::n_ids
::min_score_ptr[].order[0] elements < ::n_ids
::this_order[] elements < ::n_ids
::n_ids > ::topk
::n_ids > ::prob_dist_flag
::n_ids > ::gargc
orig(::scores[].var) elements < ::n_ids
orig(::scores[].order[0]) elements < ::n_ids
orig(::min_score_ptr[].var) elements < ::n_ids
orig(::min_score_ptr[].order[0]) elements < ::n_ids
::n_names > ::n_rounds
::rounds[] elements < ::n_names
::scores[].var elements < ::n_names
::min_score_ptr[].var elements < ::n_names
::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
orig(::scores[].var) elements < ::n_names
orig(::min_score_ptr[].var) elements < ::n_names
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
orig(::scores[].var) elements < ::n_rounds
orig(::min_score_ptr[].var) elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].weight <= orig(::scores[].auc) (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].weight < orig(::min_score_ptr[].auc) (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
orig(::scores[].var) elements >= ::prob_dist_flag
orig(::scores[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements >= ::prob_dist_flag
orig(::min_score_ptr[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements <= ::exti
==========
merge.c.setup():::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)

::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].var <= ::min_score_ptr[].var (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
::ignore_set != null
::ignore_set[] elements == 0
::ignore_set[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,

::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
==
merge.c.setup()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc >= orig(::scores[].auc) (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].weight < orig(::min_score_ptr[].auc) (lexically)
::scores[].weight <= orig(::min_score_ptr[].auc) (lexically)
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < orig(::scores[].auc) (lexically)

::min_score_ptr[].weight <= orig(::scores[].auc) (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] < orig(::min_score_ptr[].var) (lexically)
::best_order[] <= orig(::min_score_ptr[].var) (lexically)
::best_order[] < orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] <= orig(::min_score_ptr[].order[0]) (lexically)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] < orig(::min_score_ptr[].var) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].var) (lexically)
::ignore_set[] < orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] <= orig(::min_score_ptr[].order[0]) (lexically)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] > orig(::scores[].auc) (lexically)
::tabula[][0] >= orig(::scores[].auc) (lexically)
::tabula[][0] > orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] >= orig(::min_score_ptr[].auc) (lexically)
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0

::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::n_names > ::topk
::n_names > ::prob_dist_flag
::n_names > ::gargc
::n_names > ::exti
orig(::scores[].var) elements < ::n_names
orig(::min_score_ptr[].var) elements < ::n_names
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::scores[].var elements < ::n_rounds
::min_score_ptr[].var elements < ::n_rounds
orig(::scores[].var) elements < ::n_rounds
orig(::min_score_ptr[].var) elements < ::n_rounds
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].weight <= orig(::scores[].auc) (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].weight < orig(::min_score_ptr[].auc) (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
orig(::scores[].var) elements >= ::prob_dist_flag
orig(::scores[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements >= ::prob_dist_flag
orig(::min_score_ptr[].order[0]) elements >= ::prob_dist_flag
orig(::min_score_ptr[].var) elements <= ::exti
==========
ml3.c.comp_auc_desc()::ENTER
::n_rounds == ::exti
::rounds[] < ::this_order[] (lexically)
::rounds[] <= ::this_order[] (lexically)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].var > ::best_order[] (lexically)
::scores[].var >= ::best_order[] (lexically)
::scores[].var > ::ignore_set[] (lexically)
::scores[].auc > ::scores[].weight (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)

::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].order[0] > ::best_order[] (lexically)
::scores[].order[0] >= ::best_order[] (lexically)
::scores[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)
::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::n_ids == 100
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order has only one value
::order != null
::order[] elements >= 0
::sublist_order has only one value
::sublist_order != null
::sublist_order[] elements >= 0
::scores has only one value
::scores != null
::scores[] contains no nulls and has only one value, of length 10
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of ”a”, ”d”
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of ”a”, ”d”
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order has only one value
::this_order != null
::this_order[] elements >= 0
::best_order has only one value
::best_order != null
::best_order[] == [0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
::best_order[] elements == 0
::ignore_set has only one value
::ignore_set != null
::ignore_set[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
::ignore_set[] elements == 0
::topk == 1
::unknown_limit == 0.4
::prob_dist_flag == 0
::failures has only one value
::failures != null
::failures[] has only one value, of length 13
::last has only one value
::last != null
::last[] has only one value, of length 13
::gargc == 7
::gargv != null
::gargv[] == [./marti]
::gargv[] elements == "./marti"
::genvp != null
::genvp[] == [LESSOPEN=| /usr/bin/lesspipe
::genvp[] elements == "LESSOPEN=| /usr/bin/lesspipe
::tabula has only one value
::tabula != null
::tabula[] contains no nulls and has only one value, of length 100
::tabula[] elements != null
::tabula[][0] elements >= 1.0
ap != null
bp != null
::rounds[] elements < ::n_ids
::order[] elements < ::n_ids
::sublist_order[] elements < ::n_ids
::scores[].order[0] elements < ::n_ids
::min_score_ptr[].order[0] elements < ::n_ids
::this_order[] elements < ::n_ids
::rounds[] elements < ::n_names
::n_names in ::order[]
::n_rounds in ::rounds[]
::rounds[] elements <= ::n_rounds
::n_rounds in ::order[]
::rounds[] > ::ignore_set[] (elementwise)
::rounds[] elements >= ::topk
::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var >= ::ignore_set[] (elementwise)
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] >= ::ignore_set[] (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag

::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
==========
ml3.c.comp_auc_desc()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == ::exti
::n_rounds == orig(::n_rounds)
::n_rounds == orig(::exti)
::rounds == orig(::rounds)
::rounds[] < ::this_order[] (lexically)
::rounds[] <= ::this_order[] (lexically)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var > ::best_order[] (lexically)
::scores[].var >= ::best_order[] (lexically)
::scores[].var > ::ignore_set[] (lexically)
::scores[].var == orig(::scores[].var)
::scores[].auc > ::scores[].weight (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::scores[].order[0] > ::best_order[] (lexically)
::scores[].order[0] >= ::best_order[] (lexically)
::scores[].order[0] > ::ignore_set[] (lexically)
::scores[].order[0] == orig(::scores[].order[0])
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var > ::best_order[] (lexically)
::min_score_ptr[].var >= ::best_order[] (lexically)
::min_score_ptr[].var > ::ignore_set[] (lexically)
::min_score_ptr[].var >= ::ignore_set[] (lexically)
::min_score_ptr[].var == orig(::min_score_ptr[].var)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].auc == orig(::min_score_ptr[].auc)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order == orig(::min_score_ptr[].order)
::min_score_ptr[].order[0] > ::best_order[] (lexically)
::min_score_ptr[].order[0] >= ::best_order[] (lexically)

::min_score_ptr[].order[0] > ::ignore_set[] (lexically)
::min_score_ptr[].order[0] >= ::ignore_set[] (lexically)
::min_score_ptr[].order[0] == orig(::min_score_ptr[].order[0])
::this_order == orig(::this_order)
::this_order[] > ::best_order[] (lexically)
::this_order[] > ::ignore_set[] (lexically)
::this_order[] >= ::ignore_set[] (lexically)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::n_ids == 100
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order has only one value
::order != null
::order[] elements >= 0
::sublist_order has only one value
::sublist_order != null
::sublist_order[] elements >= 0
::scores has only one value
::scores != null
::scores[] contains no nulls and has only one value, of length 10
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].auc elements <= 1.0
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order has only one value
::this_order != null
::this_order[] elements >= 0
::best_order has only one value
::best_order != null

::best_order[] == [0, 0, 0,

::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::topk >= return
::gargc > return
==
ml3.c.pauc():::ENTER
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::best_order[] < orderref[] (lexically)
::ignore_set[] < orderref[] (lexically)
::ignore_set[] <= orderref[] (lexically)
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null

::this_order[] elements >= 0
::best_order != null
::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::topk in ::rounds[]
::rounds[] elements > ::prob_dist_flag
::gargc in ::rounds[]
::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::best_order[] <= orderref[] (elementwise)
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
orderref[] elements >= ::prob_dist_flag
::prob_dist_flag <= n
==========
ml3.c.pauc()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::order == orig(::order)
::order[] > ::best_order[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::sublist_order == orig(::sublist_order)
::sublist_order[] > ::best_order[] (lexically)
::sublist_order[] > ::ignore_set[] (lexically)
::sublist_order[] >= ::ignore_set[] (lexically)
::sublist_order[] == orig(::sublist_order[])
::scores == orig(::scores)
::scores[] == orig(::scores[])
::scores[].var == orig(::scores[].var)
::scores[].auc > ::min_score_ptr[].weight (lexically)
::scores[].auc >= ::min_score_ptr[].weight (lexically)
::scores[].auc < ::tabula[][0] (lexically)
::scores[].auc <= ::tabula[][0] (lexically)
::scores[].auc == orig(::scores[].auc)
::scores[].weight < ::min_score_ptr[].auc (lexically)
::scores[].weight <= ::min_score_ptr[].auc (lexically)
::scores[].weight >= ::min_score_ptr[].weight (lexically)
::scores[].weight < ::tabula[][0] (lexically)
::scores[].weight <= ::tabula[][0] (lexically)
::scores[].weight == orig(::scores[].weight)
::scores[].order == orig(::scores[].order)
::scores[].order[0] == orig(::scores[].order[0])
::min_score_ptr == orig(::min_score_ptr)
::min_score_ptr[] == orig(::min_score_ptr[])
::min_score_ptr[].var == orig(::min_score_ptr[].var)

::min_score_ptr[].auc < ::tabula[][0] (lexically)
::min_score_ptr[].auc <= ::tabula[][0] (lexically)
::min_score_ptr[].auc == orig(::min_score_ptr[].auc)
::min_score_ptr[].weight < ::tabula[][0] (lexically)
::min_score_ptr[].weight <= ::tabula[][0] (lexically)
::min_score_ptr[].weight == orig(::min_score_ptr[].weight)
::min_score_ptr[].order == orig(::min_score_ptr[].order)
::min_score_ptr[].order[0] == orig(::min_score_ptr[].order[0])
::this_order == orig(::this_order)
::this_order[] == orig(::this_order[])
::best_order == orig(::best_order)
::best_order[] < orderref[] (lexically)
::best_order[] == orig(::best_order[])
::ignore_set == orig(::ignore_set)
::ignore_set[] < orderref[] (lexically)
::ignore_set[] <= orderref[] (lexically)
::ignore_set[] == orig(::ignore_set[])
::topk == orig(::topk)
::unknown_limit == orig(::unknown_limit)
::prob_dist_flag == orig(::prob_dist_flag)
::failures == orig(::failures)
::failures[] == orig(::failures[])
::last == orig(::last)
::last[] == orig(::last[])
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::genvp == orig(::genvp)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
orderref[] == orig(orderref[])
::n_ids one of 10, 100
::n_names one of 11, 101
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[] elements >= 1
::rounds[] sorted by <
::order != null
::order[] elements >= 0
::sublist_order != null
::sublist_order[] elements >= 0
::scores != null
::scores[] elements != null
::scores[].var elements >= 0
::scores[].dir elements one of "a", "d"
::scores[].auc elements <= 1.0
::scores[].auc elements >= 0.0
::scores[].weight == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
::scores[].weight elements == 0.0
::scores[].order elements != null
::scores[].order[0] elements >= 0
::min_score_ptr[] elements != null
::min_score_ptr[].var elements >= 0
::min_score_ptr[].dir elements one of "a", "d"
::min_score_ptr[].weight elements == 0.0
::min_score_ptr[].order elements != null
::min_score_ptr[].order[0] elements >= 0
::this_order != null
::this_order[] elements >= 0
::best_order != null

::best_order[] elements == 0
::best_order[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0,

::order[] >= ::best_order[] (elementwise)
::topk in ::order[]
::order[] elements >= ::prob_dist_flag
::prob_dist_flag in ::order[]
::gargc in ::order[]
::sublist_order[] >= ::best_order[] (elementwise)
::sublist_order[] elements >= ::prob_dist_flag
::scores[].var elements >= ::prob_dist_flag
::scores[].auc >= ::scores[].weight (elementwise)
::scores[].order[0] elements >= ::prob_dist_flag
::min_score_ptr[].var elements >= ::prob_dist_flag
::min_score_ptr[].var elements <= ::exti
::min_score_ptr[].auc > ::min_score_ptr[].weight (elementwise)
::min_score_ptr[].order[0] elements >= ::prob_dist_flag
::this_order[] >= ::best_order[] (elementwise)
::this_order[] elements >= ::prob_dist_flag
::best_order[] elements == ::prob_dist_flag
::prob_dist_flag in ::best_order[]
::best_order[] <= orderref[] (elementwise)
::ignore_set[] elements == ::prob_dist_flag
::prob_dist_flag in ::ignore_set[]
::prob_dist_flag <= ::exti
orderref[] elements >= ::prob_dist_flag
::prob_dist_flag <= orig(n)

11.3 PAYL

==========
CalibratePacketHandler:::OBJECT
this.MM.UDP_model[] == this.MM.TCP_model_in[]
this.MM.UDP_model[] == this.MM.UDP_model_in[]
this.MM.UDP_model[] == this.MM.TCP_model_out[]
this.MM.UDP_model[] == this.MM.UDP_model_out[]
this.MM.UDP_model[] == this.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_outsrc[]
this.MM.trunc_length == this.trunc_length
this.MM.min_sample == this.MM.modification_time
this.MM.host_ip == this.host_ip
this.MM.lenProb_out[] == this.MM.lenProb_in[]
this.MM.lenProb_out[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_outsrc[]
this has only one value
this.MM has only one value
this.MM != null
this.MM.TCP_model has only one value
this.MM.TCP_model != null
this.MM.TCP_model.getClass() == java.util.HashMap[].class
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null,

null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.MM.anomalyLengthThreshold_out[] elements == 0.0
this.MM.lenProb_in has only one value
this.MM.lenProb_in != null
this.MM.anomalyLengthThreshold_in has only one value
this.MM.anomalyLengthThreshold_in != null
this.MM.lenProb_outsrc has only one value
this.MM.lenProb_outsrc != null
this.MM.anomalyLengthThreshold_outsrc has only one value
this.MM.anomalyLengthThreshold_outsrc != null
this.MM.UDPNormalLength has only one value
this.MM.UDPNormalLength != null
this.MM.UDPNormalLength_out has only one value
this.MM.UDPNormalLength_out != null
this.MM.UDPNormalLength_in has only one value
this.MM.UDPNormalLength_in != null
this.MM.UDPNormalLength_outsrc has only one value
this.MM.UDPNormalLength_outsrc != null
this.MM.normal_length_perc == 0.999
this.trunc_length == -1
this.host_ip has only one value
this.host_ip != null
this.host_ip.toString == ”128.59.16.101”
this.TCP == 6
this.UDP == 17
this.MM.TCP_model.getClass() == this.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() != this.MM.TCP_lengthModel.getClass()
this.MM.TCP_model.getClass() != this.MM.lenProb.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() != this.MM.lenProb.getClass()
this.MM.host_ip.toString == this.host_ip.toString
this.MM.anomalyLengthThreshold[] elements < this.MM.stabledis
this.MM.lenProb.getClass() == this.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_outsrc.getClass()
this.MM.anomalyLengthThreshold[] >= this.MM.anomalyLengthThreshold_out[] (elementwise)
this.MM.anomalyLengthThreshold[] elements < this.MM.normal_length_perc
==========
CalibratePacketHandler.CalibratePacketHandler(Models, java.lang.String):::ENTER
arg0.UDP_model[] == arg0.TCP_model_in[]
arg0.UDP_model[] == arg0.UDP_model_in[]
arg0.UDP_model[] == arg0.TCP_model_out[]
arg0.UDP_model[] == arg0.UDP_model_out[]

arg0.UDP_model[] == arg0.TCP_model_outsrc[]
arg0.UDP_model[] == arg0.UDP_model_outsrc[]
arg0.UDP_model[] == arg0.UDPNormalLength[]
arg0.UDP_model[] == arg0.UDPNormalLength_out[]
arg0.UDP_model[] == arg0.UDPNormalLength_in[]
arg0.UDP_model[] == arg0.UDPNormalLength_outsrc[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_outsrc[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_outsrc[]
arg0.min_sample == arg0.modification_time
arg0.host_ip == arg1
arg0.lenProb_out[] == arg0.lenProb_in[]
arg0.lenProb_out[] == arg0.lenProb_outsrc[]
arg0.anomalyLengthThreshold_out[] == arg0.anomalyLengthThreshold_in[]
arg0.anomalyLengthThreshold_out[] == arg0.anomalyLengthThreshold_outsrc[]
arg0 has only one value
arg0.TCP_model has only one value
arg0.TCP_model.getClass() == java.util.HashMap[].class
arg0.TCP_model[] has only one value, of length 1024
arg0.TCP_model[].getClass() == [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null,
null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, java.util.HashMap, null, null, null, null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null

null, null, null,

null, null, null,

0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

arg0.anomalyLengthThreshold_out[] elements == 0.0
arg0.lenProb_in has only one value
arg0.anomalyLengthThreshold_in has only one value
arg0.lenProb_outsrc has only one value
arg0.anomalyLengthThreshold_outsrc has only one value
arg0.UDPNormalLength has only one value
arg0.UDPNormalLength_out has only one value
arg0.UDPNormalLength_in has only one value
arg0.UDPNormalLength_outsrc has only one value
arg0.normal_length_perc == 0.999
arg1 has only one value
arg1.toString == ”128.59.16.101”
arg0.TCP_model.getClass() == arg0.UDP_model.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_out.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_in.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_outsrc.getClass()
arg0.host_ip.toString == arg1.toString
arg0.lenProb.getClass() == arg0.lenProb_out.getClass()
arg0.lenProb.getClass() == arg0.lenProb_in.getClass()
arg0.lenProb.getClass() == arg0.lenProb_outsrc.getClass()

==========

CalibratePacketHandler.CalibratePacketHandler(Models, java.lang.String):::EXIT
this.MM == orig(arg0)
this.MM.TCP_model == arg0.TCP_model
this.MM.TCP_model[] == arg0.TCP_model[]
this.MM.UDP_model == arg0.UDP_model
this.MM.UDP_model[] == arg0.UDP_model[]
this.MM.TCP_model_in == arg0.TCP_model_in
this.MM.TCP_model_in[] == arg0.UDP_model[]
this.MM.UDP_model_in == arg0.UDP_model_in
this.MM.UDP_model_in[] == arg0.UDP_model[]
this.MM.TCP_model_out == arg0.TCP_model_out
this.MM.TCP_model_out[] == arg0.UDP_model[]
this.MM.UDP_model_out == arg0.UDP_model_out
this.MM.UDP_model_out[] == arg0.UDP_model[]
this.MM.TCP_model_outsrc == arg0.TCP_model_outsrc

this.MM.TCP_model_outsrc[] == arg0.UDP_model[]
this.MM.UDP_model_outsrc == arg0.UDP_model_outsrc
this.MM.UDP_model_outsrc[] == arg0.UDP_model[]
this.MM.TCP_lengthModel == arg0.TCP_lengthModel
this.MM.TCP_lengthModel[] == arg0.TCP_lengthModel[]
this.MM.UDP_lengthModel == arg0.UDP_lengthModel
this.MM.UDP_lengthModel[] == arg0.UDP_lengthModel[]
this.MM.TCP_lengthModel_in == arg0.TCP_lengthModel_in
this.MM.TCP_lengthModel_in[] == arg0.UDP_lengthModel[]
this.MM.UDP_lengthModel_in == arg0.UDP_lengthModel_in
this.MM.UDP_lengthModel_in[] == arg0.UDP_lengthModel[]
this.MM.TCP_lengthModel_out == arg0.TCP_lengthModel_out
this.MM.TCP_lengthModel_out[] == arg0.UDP_lengthModel[]
this.MM.UDP_lengthModel_out == arg0.UDP_lengthModel_out
this.MM.UDP_lengthModel_out[] == arg0.UDP_lengthModel[]
this.MM.TCP_lengthModel_outsrc == arg0.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel_outsrc[] == arg0.UDP_lengthModel[]
this.MM.UDP_lengthModel_outsrc == arg0.UDP_lengthModel_outsrc
this.MM.UDP_lengthModel_outsrc[] == arg0.UDP_lengthModel[]
this.MM.port_trunc_length == arg0.port_trunc_length
this.MM.min_sample == arg0.min_sample
this.MM.max_sample == arg0.max_sample
this.MM.sample_rate == arg0.sample_rate
this.MM.stablesize == arg0.stablesize
this.MM.stabledis == arg0.stabledis
this.MM.smooth_factor == arg0.smooth_factor
this.MM.modification_time == arg0.min_sample
this.MM.aggregation_done == arg0.aggregation_done
this.MM.lenProb == arg0.lenProb
this.MM.lenProb[] == arg0.lenProb[]
this.MM.anomalyLengthThreshold == arg0.anomalyLengthThreshold
this.MM.anomalyLengthThreshold[] == arg0.anomalyLengthThreshold[]
this.MM.lenProb_out == arg0.lenProb_out
this.MM.lenProb_out[] == arg0.lenProb_out[]
this.MM.anomalyLengthThreshold_out == arg0.anomalyLengthThreshold_out
this.MM.anomalyLengthThreshold_out[] == arg0.anomalyLengthThreshold_out[]
this.MM.lenProb_in == arg0.lenProb_in
this.MM.lenProb_in[] == arg0.lenProb_out[]
this.MM.anomalyLengthThreshold_in == arg0.anomalyLengthThreshold_in
this.MM.anomalyLengthThreshold_in[] == arg0.anomalyLengthThreshold_out[]
this.MM.lenProb_outsrc == arg0.lenProb_outsrc
this.MM.lenProb_outsrc[] == arg0.lenProb_out[]
this.MM.anomalyLengthThreshold_outsrc == arg0.anomalyLengthThreshold_outsrc
this.MM.anomalyLengthThreshold_outsrc[] == arg0.anomalyLengthThreshold_out[]
this.MM.UDPNormalLength == arg0.UDPNormalLength
this.MM.UDPNormalLength[] == arg0.UDP_model[]
this.MM.UDPNormalLength_out == arg0.UDPNormalLength_out
this.MM.UDPNormalLength_out[] == arg0.UDP_model[]
this.MM.UDPNormalLength_in == arg0.UDPNormalLength_in
this.MM.UDPNormalLength_in[] == arg0.UDP_model[]
this.MM.UDPNormalLength_outsrc == arg0.UDPNormalLength_outsrc
this.MM.UDPNormalLength_outsrc[] == arg0.UDP_model[]
this.MM.normal_length_perc == arg0.normal_length_perc
this.trunc_length == arg0.trunc_length
this.trunc_length == orig(arg0.trunc_length)
this.port_trunc_length == this.port_keywords
this.port_trunc_length == this.port_keywords_range
this.in_dst_ports_set == this.in_src_ports_set
this.in_dst_ports_set == this.out_dst_ports_set
this.in_dst_ports_set == this.out_src_ports_set
this.host_ip == arg0.host_ip
this.host_ip == orig(arg0.host_ip)
this.host_ip == orig(arg1)

arg0.TCP_model == orig(arg0.TCP_model)
arg0.TCP_model[] == orig(arg0.TCP_model[])
arg0.UDP_model == orig(arg0.UDP_model)
arg0.UDP_model[] == arg0.TCP_model_in[]
arg0.UDP_model[] == arg0.UDP_model_in[]
arg0.UDP_model[] == arg0.TCP_model_out[]
arg0.UDP_model[] == arg0.UDP_model_out[]
arg0.UDP_model[] == arg0.TCP_model_outsrc[]
arg0.UDP_model[] == arg0.UDP_model_outsrc[]
arg0.UDP_model[] == arg0.UDPNormalLength[]
arg0.UDP_model[] == arg0.UDPNormalLength_out[]
arg0.UDP_model[] == arg0.UDPNormalLength_in[]
arg0.UDP_model[] == arg0.UDPNormalLength_outsrc[]
arg0.UDP_model[] == orig(arg0.UDP_model[])
arg0.UDP_model[] == orig(arg0.TCP_model_in[])
arg0.UDP_model[] == orig(arg0.UDP_model_in[])
arg0.UDP_model[] == orig(arg0.TCP_model_out[])
arg0.UDP_model[] == orig(arg0.UDP_model_out[])
arg0.UDP_model[] == orig(arg0.TCP_model_outsrc[])
arg0.UDP_model[] == orig(arg0.UDP_model_outsrc[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength_out[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength_in[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength_outsrc[])
arg0.TCP_model_in == orig(arg0.TCP_model_in)
arg0.UDP_model_in == orig(arg0.UDP_model_in)
arg0.TCP_model_out == orig(arg0.TCP_model_out)
arg0.UDP_model_out == orig(arg0.UDP_model_out)
arg0.TCP_model_outsrc == orig(arg0.TCP_model_outsrc)
arg0.UDP_model_outsrc == orig(arg0.UDP_model_outsrc)
arg0.TCP_lengthModel == orig(arg0.TCP_lengthModel)
arg0.TCP_lengthModel[] == orig(arg0.TCP_lengthModel[])
arg0.UDP_lengthModel == orig(arg0.UDP_lengthModel)
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_outsrc[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_outsrc[]
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel[])
arg0.UDP_lengthModel[] == orig(arg0.TCP_lengthModel_in[])
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel_in[])
arg0.UDP_lengthModel[] == orig(arg0.TCP_lengthModel_out[])
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel_out[])
arg0.UDP_lengthModel[] == orig(arg0.TCP_lengthModel_outsrc[])
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel_outsrc[])
arg0.TCP_lengthModel_in == orig(arg0.TCP_lengthModel_in)
arg0.UDP_lengthModel_in == orig(arg0.UDP_lengthModel_in)
arg0.TCP_lengthModel_out == orig(arg0.TCP_lengthModel_out)
arg0.UDP_lengthModel_out == orig(arg0.UDP_lengthModel_out)
arg0.TCP_lengthModel_outsrc == orig(arg0.TCP_lengthModel_outsrc)
arg0.UDP_lengthModel_outsrc == orig(arg0.UDP_lengthModel_outsrc)
arg0.port_trunc_length == orig(arg0.port_trunc_length)
arg0.min_sample == arg0.modification_time
arg0.min_sample == orig(arg0.min_sample)
arg0.min_sample == orig(arg0.modification_time)
arg0.max_sample == orig(arg0.max_sample)
arg0.sample_rate == orig(arg0.sample_rate)
arg0.stablesize == orig(arg0.stablesize)
arg0.stabledis == orig(arg0.stabledis)
arg0.smooth_factor == orig(arg0.smooth_factor)
arg0.aggregation_done == orig(arg0.aggregation_done)
arg0.lenProb == orig(arg0.lenProb)

arg0.lenProb[] == orig(arg0.lenProb[])
arg0.anomalyLengthThreshold == orig(arg0.anomalyLengthThreshold)
arg0.anomalyLengthThreshold[] == orig(arg0.anomalyLengthThreshold[])
arg0.lenProb_out == orig(arg0.lenProb_out)
arg0.lenProb_out[] == arg0.lenProb_in[]
arg0.lenProb_out[] == arg0.lenProb_outsrc[]
arg0.lenProb_out[] == orig(arg0.lenProb_out[])
arg0.lenProb_out[] == orig(arg0.lenProb_in[])
arg0.lenProb_out[] == orig(arg0.lenProb_outsrc[])
arg0.anomalyLengthThreshold_out == orig(arg0.anomalyLengthThreshold_out)
arg0.anomalyLengthThreshold_out[] == arg0.anomalyLengthThreshold_in[]
arg0.anomalyLengthThreshold_out[] == arg0.anomalyLengthThreshold_outsrc[]
arg0.anomalyLengthThreshold_out[] == orig(arg0.anomalyLengthThreshold_out[])
arg0.anomalyLengthThreshold_out[] == orig(arg0.anomalyLengthThreshold_in[])
arg0.anomalyLengthThreshold_out[] == orig(arg0.anomalyLengthThreshold_outsrc[])
arg0.lenProb_in == orig(arg0.lenProb_in)
arg0.anomalyLengthThreshold_in == orig(arg0.anomalyLengthThreshold_in)
arg0.lenProb_outsrc == orig(arg0.lenProb_outsrc)
arg0.anomalyLengthThreshold_outsrc == orig(arg0.anomalyLengthThreshold_outsrc)
arg0.UDPNormalLength == orig(arg0.UDPNormalLength)
arg0.UDPNormalLength_out == orig(arg0.UDPNormalLength_out)
arg0.UDPNormalLength_in == orig(arg0.UDPNormalLength_in)
arg0.UDPNormalLength_outsrc == orig(arg0.UDPNormalLength_outsrc)
arg0.normal_length_perc == orig(arg0.normal_length_perc)
this.port_trunc_length == null
this.in_dst_ports_set == null
arg0.TCP_model has only one value
arg0.TCP_model.getClass() == java.util.HashMap[].class
arg0.TCP_model[] has only one value, of length 1024
arg0.TCP_model[].getClass() == [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null,
null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, java.util.HashMap, null, null, null, null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null

null, null, null,

arg0.TCP_lengthModel has only one value
arg0.TCP_lengthModel.getClass() == java.util.Vector[].class
arg0.TCP_lengthModel[] has only one value, of length 1024
arg0.TCP_lengthModel[].getClass() == [null, null, null,

arg0.aggregation_done == false
arg0.lenProb has only one value
arg0.lenProb.getClass() == double[][].class
arg0.lenProb[] has only one value, of length 1024
arg0.anomalyLengthThreshold has only one value
arg0.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
arg0.anomalyLengthThreshold_out[] elements == 0.0
arg0.lenProb_in has only one value
arg0.anomalyLengthThreshold_in has only one value
arg0.lenProb_outsrc has only one value
arg0.anomalyLengthThreshold_outsrc has only one value
arg0.UDPNormalLength has only one value
arg0.UDPNormalLength_out has only one value
arg0.UDPNormalLength_in has only one value
arg0.UDPNormalLength_outsrc has only one value
arg0.normal_length_perc == 0.999
arg1.toString == ”128.59.16.101”
this.MM.TCP_model.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_model_in.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model_in.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_model_out.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model_out.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_model_outsrc.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model_outsrc.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_lengthModel.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel_in.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel_in.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel_out.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel_out.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel_outsrc.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel_outsrc.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.host_ip.toString == arg1.toString
this.MM.lenProb.getClass() == arg0.lenProb.getClass()
this.MM.lenProb_out.getClass() == arg0.lenProb.getClass()
this.MM.lenProb_in.getClass() == arg0.lenProb.getClass()
this.MM.lenProb_outsrc.getClass() == arg0.lenProb.getClass()
this.MM.UDPNormalLength.getClass() == arg0.TCP_model.getClass()
this.MM.UDPNormalLength_out.getClass() == arg0.TCP_model.getClass()
this.MM.UDPNormalLength_in.getClass() == arg0.TCP_model.getClass()
this.MM.UDPNormalLength_outsrc.getClass() == arg0.TCP_model.getClass()
this.port_trunc_length in arg0.TCP_model[]
arg0.UDP_model[] elements == this.port_trunc_length
this.port_trunc_length in arg0.UDP_model[]
this.host_ip.toString == arg1.toString
arg0.TCP_model.getClass() == arg0.UDP_model.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_out.getClass()

arg0.TCP_model.getClass() == arg0.UDP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_out.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_in.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_outsrc.getClass()
arg0.TCP_model.getClass() == orig(arg0.TCP_model.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model.getClass())
arg0.TCP_model.getClass() == orig(arg0.TCP_model_in.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model_in.getClass())
arg0.TCP_model.getClass() == orig(arg0.TCP_model_out.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model_out.getClass())
arg0.TCP_model.getClass() == orig(arg0.TCP_model_outsrc.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model_outsrc.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength_out.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength_in.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength_outsrc.getClass())
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel_in.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel_in.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel_out.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel_out.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel_outsrc.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel_outsrc.getClass())
arg0.host_ip.toString == arg1.toString
arg0.lenProb.getClass() == arg0.lenProb_out.getClass()
arg0.lenProb.getClass() == arg0.lenProb_in.getClass()
arg0.lenProb.getClass() == arg0.lenProb_outsrc.getClass()
arg0.lenProb.getClass() == orig(arg0.lenProb.getClass())
arg0.lenProb.getClass() == orig(arg0.lenProb_out.getClass())
arg0.lenProb.getClass() == orig(arg0.lenProb_in.getClass())
arg0.lenProb.getClass() == orig(arg0.lenProb_outsrc.getClass())
arg1.toString == orig(arg0.host_ip.toString)
arg1.toString == orig(arg1.toString)
==========
CalibratePacketHandler.packetArrived(net.sourceforge.jpcap.net.Packet):::ENTER
this.MM.port_trunc_length == this.port_trunc_length
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
arg0 != null
arg0._rcsid has only one value

arg0._rcsid != null
arg0._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this.host_ip.toString > arg0._rcsid.toString
==========
CalibratePacketHandler.packetArrived(net.sourceforge.jpcap.net.Packet):::EXIT118
==========
CalibratePacketHandler.packetArrived(net.sourceforge.jpcap.net.Packet):::EXIT
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.port_trunc_length == this.port_trunc_length
this.MM.min_sample == orig(this.MM.min_sample)
this.MM.min_sample == orig(this.MM.modification_time)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.stablesize == orig(this.MM.stablesize)
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.aggregation_done == orig(this.MM.aggregation_done)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])

this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.trunc_length == orig(this.MM.trunc_length)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
arg0._rcsid == orig(arg0._rcsid)
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
arg0._rcsid has only one value
arg0._rcsid != null
arg0._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())

this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.host_ip.toString > arg0._rcsid.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
arg0._rcsid.toString == orig(arg0._rcsid.toString)
==========
Models:::OBJECT
this.UDP_model[] == this.TCP_model_in[]
this.UDP_model[] == this.UDP_model_in[]
this.UDP_model[] == this.TCP_model_out[]
this.UDP_model[] == this.UDP_model_out[]
this.UDP_model[] == this.TCP_model_outsrc[]
this.UDP_model[] == this.UDP_model_outsrc[]
this.UDP_model[].getClass() == this.UDP_lengthModel[].getClass()
this.UDP_model[].getClass() == this.UDPNormalLength[].getClass()
this.UDP_lengthModel[] == this.TCP_lengthModel_in[]
this.UDP_lengthModel[] == this.UDP_lengthModel_in[]
this.UDP_lengthModel[] == this.TCP_lengthModel_out[]
this.UDP_lengthModel[] == this.UDP_lengthModel_out[]
this.UDP_lengthModel[] == this.TCP_lengthModel_outsrc[]
this.UDP_lengthModel[] == this.UDP_lengthModel_outsrc[]
this.UDP_lengthModel[].getClass() == this.UDPNormalLength[].getClass()
this.min_sample == this.modification_time
this.lenProb_out[] == this.lenProb_in[]
this.lenProb_out[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold_out[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold_out[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength[] == this.UDPNormalLength_out[]
this.UDPNormalLength[] == this.UDPNormalLength_in[]
this.UDPNormalLength[] == this.UDPNormalLength_outsrc[]
this has only one value
this.TCP_model has only one value
this.TCP_model != null
this.TCP_model.getClass() == java.util.HashMap[].class
this.TCP_model[].getClass() one of [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null, null, null, null, null,

null, null, null,

this.UDP_model[] elements == null
this.UDP_model[].getClass() == [null, null, null,

null, null, null,

null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

null, null,

arg5.toString == "128.59.16.101"
arg6 == 1
arg7 == 0.05
arg8 == 0.0010
==========
Models.Models(int, java.util.HashMap, int, int, float, java.lang.String, int, double, float):::EXIT
this.TCP_model[] == this.UDP_model[]
this.TCP_model[] == this.TCP_model_in[]
this.TCP_model[] == this.UDP_model_in[]
this.TCP_model[] == this.TCP_model_out[]
this.TCP_model[] == this.UDP_model_out[]
this.TCP_model[] == this.TCP_model_outsrc[]
this.TCP_model[] == this.UDP_model_outsrc[]
this.TCP_lengthModel == this.UDP_lengthModel
this.TCP_lengthModel == this.TCP_lengthModel_in
this.TCP_lengthModel == this.UDP_lengthModel_in
this.TCP_lengthModel == this.TCP_lengthModel_out
this.TCP_lengthModel == this.UDP_lengthModel_out
this.TCP_lengthModel == this.TCP_lengthModel_outsrc
this.TCP_lengthModel == this.UDP_lengthModel_outsrc
this.trunc_length == orig(arg0)
this.port_trunc_length == orig(arg1)
this.min_sample == orig(arg2)
this.max_sample == orig(arg3)
this.sample_rate == orig(arg4)
this.host_ip == orig(arg5)
this.stablesize == orig(arg6)
this.stabledis == orig(arg7)
this.smooth_factor == orig(arg8)
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] contains only nulls and has only one value, of length 1024
this.TCP_model[] elements == null
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

arg5.toString == ”128.59.16.101”
this.host_ip.toString == arg5.toString
this.anomalyLengthThreshold[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold[]
arg5.toString == orig(arg5.toString)
==========
Models.addDistance(java.lang.String, java.lang.String, int, byte[]):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.UDP_lengthModel_in has only one value
this.UDP_lengthModel_in != null
this.TCP_lengthModel_out has only one value
this.TCP_lengthModel_out != null
this.UDP_lengthModel_out has only one value
this.UDP_lengthModel_out != null
this.TCP_lengthModel_outsrc has only one value
this.TCP_lengthModel_outsrc != null
this.UDP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc != null
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

arg3 != null
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
arg3[] elements > this.trunc_length
arg3[] elements > this.min_sample
arg3[] elements < this.max_sample
this.host_ip.toString < arg0.toString
this.host_ip.toString < arg1.toString
arg3[] elements > this.stablesize
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg0.toString < arg1.toString
arg2 in arg3[]
==========
Models.addDistance(java.lang.String, java.lang.String, int, byte[]):::EXIT1656
==========
Models.addDistance(java.lang.String, java.lang.String, int, byte[]):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)

this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg3[] == orig(arg3[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.UDPNormalLength has only one value
this.UDPNormalLength != null
this.UDPNormalLength_out has only one value
this.UDPNormalLength_out != null
this.UDPNormalLength_in has only one value
this.UDPNormalLength_in != null
this.UDPNormalLength_outsrc has only one value
this.UDPNormalLength_outsrc != null
this.normal_length_perc == 0.999
arg0.toString == "TCP_model"
arg1.toString == "tcp"
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
arg3[] elements > this.trunc_length
arg3[] elements > this.min_sample
arg3[] elements < this.max_sample
this.host_ip.toString < arg0.toString
this.host_ip.toString < arg1.toString
this.host_ip.toString == orig(this.host_ip.toString)
arg3[] elements > this.stablesize
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg0.toString < arg1.toString
arg0.toString == orig(arg0.toString)
arg1.toString == orig(arg1.toString)
orig(arg2) in arg3[]
==========
Models.adjustModel(double):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
arg0 == 0.2
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
==========
Models.adjustModel(double):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])

this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.anomalyLengthThreshold[] elements == 0.0
this.UDPNormalLength == null
this.normal_length_perc == 0.0
arg0 has only one value
arg0 in this.TCP_model[]
this.anomalyLengthThreshold[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold[]
==========
Models.buildLength(java.util.HashMap):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)

this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.lenProb[] == orig(this.lenProb[])
this.lenProb[] == orig(this.lenProb_out[])
this.lenProb[] == orig(this.lenProb_in[])
this.lenProb[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_out == orig(this.lenProb_out)
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.anomalyLengthThreshold[] elements == 0.0
this.UDPNormalLength == null
this.normal_length_perc == 0.0
return has only one value
return[] contains no nulls and has only one value, of length 386
return[].getClass() == [java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer,
java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer, java.lang.Integer,
java.lang.Integer,

this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
orig(arg0) in this.TCP_model[]
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold[]
==========
Models.byteToFloat(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
Models.byteToFloat(byte[], int):::EXIT
arg0[] == orig(arg0[])
return != null
return[] elements >= 0.0
arg0[] elements > orig(arg1)
==========
Models.byteToShort(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
Models.byteToShort(byte[], int):::EXIT
arg0[] > return[] (lexically)
arg0[] >= return[] (lexically)
arg0[] == orig(arg0[])
return != null
return[] elements >= 0
arg0[] elements > orig(arg1)
return[] elements > orig(arg1)
==========
Models.computeSmoothLenProb(double):::ENTER
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb[])
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.UDPNormalLength == null
this.normal_length_perc == 0.999
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
==========
Models.computeT(java.util.HashMap[], java.util.Vector[], java.lang.String):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == arg0.getClass()
this.TCP_lengthModel.getClass() == arg1.getClass()
this.host_ip.toString < arg2.toString
this.anomalyLengthThreshold[] elements < this.normal_length_perc
==========
Models.computeT(java.util.HashMap[], java.util.Vector[], java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)

this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_model.getClass() == orig(arg0.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(arg1.getClass())
this.host_ip.toString < arg2.toString
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg2.toString == orig(arg2.toString)
==========
Models.computeTCPLenProb(double, double):::ENTER
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.anomalyLengthThreshold[] elements == 0.0
this.UDPNormalLength == null
this.normal_length_perc == 0.0
arg0 == 5.0
arg1 == 0.999
this.anomalyLengthThreshold[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold[]
==========
Models.computeTCPLenProb(double, double):::EXIT423
==========
Models.computeTCPLenProb(double, double):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb[])
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold[])

this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(arg1)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.TCP_lengthModel_in has only one value
this.UDP_lengthModel_in has only one value
this.TCP_lengthModel_out has only one value
this.UDP_lengthModel_out has only one value
this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
orig(this.normal_length_perc) in this.anomalyLengthThreshold[]
orig(this.normal_length_perc) in this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold_out[] elements == orig(this.normal_length_perc)
==========
Models.computeUDPLength():::ENTER
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()

==========

Models.decideThreshold():::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)

this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null,
null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, java.util.HashMap, null, null, null, null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null

null, null, null,

this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
==========
Models.endMergeCentroid(java.util.Vector, java.util.Vector, java.util.HashMap):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

java.lang.Integer, java.lang.Integer, java.lang.Integer,

this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

java.lang.Integer, java.lang.Integer, java.lang.Integer,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == arg0.getClass()
this.TCP_lengthModel.getClass() == arg1.getClass()
==========
Models.fillTCPModel_onedirection(java.util.HashMap[], java.util.Vector[]):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)

this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_model.getClass() == orig(arg0.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(arg1.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
==========
Models.findMergePos(OneModel, int, OneModel, double):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.min_sample == arg0.min_sample
this.min_sample == arg2.min_sample
this.max_sample == arg0.max_sample
this.max_sample == arg2.max_sample
this.sample_rate == arg0.sample_rate
this.sample_rate == arg2.sample_rate
this.aggregation_done == arg0.batch
this.aggregation_done == arg2.batch
arg0.NMax == arg2.NMax
arg0.start == arg0.end
arg0.b_packets[] == arg0.b_nums[]
arg0.b_packets[] == arg2.b_packets[]
arg0.b_packets[] == arg2.b_nums[]
arg2.start == arg2.end
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.trunc_length < arg2.start
this.trunc_length < arg2.totalPacketNum
this.trunc_length < arg2.reduceN
this.min_sample < arg0.start
this.min_sample < arg0.totalPacketNum
this.min_sample <= arg0.reduceN
this.min_sample < arg2.start
this.min_sample < arg2.totalPacketNum
this.min_sample <= arg2.reduceN
this.max_sample != arg0.start
this.max_sample > arg0.totalPacketNum
this.max_sample > arg0.reduceN
this.max_sample > arg1
this.max_sample != arg2.start
this.max_sample > arg2.totalPacketNum
this.max_sample > arg2.reduceN
this.stablesize < arg0.start
this.stablesize <= arg0.totalPacketNum
this.stablesize < arg2.start
this.stablesize <= arg2.totalPacketNum
this.anomalyLengthThreshold[] elements < this.normal_length_perc
this.anomalyLengthThreshold[] elements < arg3
arg0.NMax < arg0.start
arg0.NMax > arg0.reduceN
arg0.NMax > arg1
arg0.NMax < arg2.start
arg0.NMax > arg2.reduceN
arg0.start > arg0.totalPacketNum
arg0.start > arg0.reduceN
arg0.start > arg1
arg0.start < arg2.start
arg0.start > arg2.totalPacketNum
arg0.start > arg2.reduceN
arg0.totalPacketNum > arg0.reduceN
arg0.totalPacketNum > arg1
arg0.totalPacketNum < arg2.start
arg0.reduceN < arg2.start
arg0.reduceN != arg2.totalPacketNum
arg1 < arg2.start
arg2.start > arg2.totalPacketNum
arg2.start > arg2.reduceN
arg2.totalPacketNum > arg2.reduceN
==========
Models.findMergePos(OneModel, int, OneModel, double):::EXIT771
orig(arg1) >= 0
this.trunc_length < orig(arg1)
this.min_sample <= orig(arg1)
orig(arg1) in return[]
==========
Models.findMergePos(OneModel, int, OneModel, double):::EXIT795
this.trunc_length == orig(arg1)
==========
Models.findMergePos(OneModel, int, OneModel, double):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])

this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == arg0.min_sample
this.min_sample == arg2.min_sample
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.min_sample == orig(arg0.min_sample)
this.min_sample == orig(arg2.min_sample)
this.max_sample == arg0.max_sample
this.max_sample == arg2.max_sample
this.max_sample == orig(this.max_sample)
this.max_sample == orig(arg0.max_sample)
this.max_sample == orig(arg2.max_sample)
this.sample_rate == arg0.sample_rate
this.sample_rate == arg2.sample_rate
this.sample_rate == orig(this.sample_rate)
this.sample_rate == orig(arg0.sample_rate)
this.sample_rate == orig(arg2.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == arg0.batch
this.aggregation_done == arg2.batch
this.aggregation_done == orig(this.aggregation_done)
this.aggregation_done == orig(arg0.batch)
this.aggregation_done == orig(arg2.batch)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])

this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg0.centroids == orig(arg0.centroids)
arg0.centroids[] == orig(arg0.centroids[])
arg0.NMax == arg2.NMax
arg0.NMax == orig(arg0.NMax)
arg0.NMax == orig(arg2.NMax)
arg0.start == arg0.end
arg0.start == orig(arg0.start)
arg0.start == orig(arg0.end)
arg0.totalPacketNum == orig(arg0.totalPacketNum)
arg0.b_packets == orig(arg0.b_packets)
arg0.b_packets[] == arg0.b_nums[]
arg0.b_packets[] == arg2.b_packets[]
arg0.b_packets[] == arg2.b_nums[]
arg0.b_packets[] == orig(arg0.b_packets[])
arg0.b_packets[] == orig(arg0.b_nums[])
arg0.b_packets[] == orig(arg2.b_packets[])
arg0.b_packets[] == orig(arg2.b_nums[])
arg0.b_nums == orig(arg0.b_nums)
arg0.reduceN == orig(arg0.reduceN)
arg2.centroids == orig(arg2.centroids)
arg2.centroids[] == orig(arg2.centroids[])
arg2.start == arg2.end
arg2.start == orig(arg2.start)
arg2.start == orig(arg2.end)
arg2.totalPacketNum == orig(arg2.totalPacketNum)
arg2.b_packets == orig(arg2.b_packets)
arg2.b_nums == orig(arg2.b_nums)
arg2.reduceN == orig(arg2.reduceN)
(orig(arg1) == -1) <==> (this.min_sample > orig(arg1))
(orig(arg1) == -1) <==> (this.trunc_length == orig(arg1))
(orig(arg1) == -1) ==> (arg0.reduceN > orig(arg1))
(orig(arg1) == -1) ==> (arg2.reduceN > orig(arg1))
(orig(arg1) == -1) ==> (arg2.totalPacketNum > orig(arg1))
(orig(arg1) == -1) ==> (return[] elements >= orig(arg1))
(orig(arg1) == -1) ==> (this.stablesize > orig(arg1))
(orig(arg1) >= 0) <==> (this.min_sample <= orig(arg1))
(orig(arg1) >= 0) <==> (this.trunc_length < orig(arg1))
(orig(arg1) >= 0) ==> (orig(arg1) in return[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.trunc_length < arg0.start
this.trunc_length < arg0.totalPacketNum
this.trunc_length < arg0.reduceN
this.trunc_length < arg2.start
this.trunc_length < arg2.totalPacketNum
this.trunc_length < arg2.reduceN
return[] elements >= this.trunc_length
this.trunc_length <= orig(arg1)
this.min_sample < arg0.start
this.min_sample < arg0.totalPacketNum
this.min_sample <= arg0.reduceN
this.min_sample < arg2.start
this.min_sample < arg2.totalPacketNum
this.min_sample <= arg2.reduceN
this.max_sample != arg0.start
this.max_sample > arg0.totalPacketNum
this.max_sample > arg0.reduceN
this.max_sample != arg2.start
this.max_sample > arg2.totalPacketNum
this.max_sample > arg2.reduceN
return[] elements < this.max_sample
this.max_sample > orig(arg1)
this.host_ip.toString == orig(this.host_ip.toString)
this.stablesize < arg0.start
this.stablesize <= arg0.totalPacketNum
this.stablesize < arg2.start
this.stablesize <= arg2.totalPacketNum
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements < this.normal_length_perc
this.anomalyLengthThreshold[] elements < orig(arg3)
arg0.NMax < arg0.start
arg0.NMax > arg0.reduceN
arg0.NMax < arg2.start
arg0.NMax > arg2.reduceN
return[] elements < arg0.NMax
arg0.NMax > orig(arg1)
arg0.start > arg0.totalPacketNum
arg0.start > arg0.reduceN
arg0.start < arg2.start
arg0.start > arg2.totalPacketNum
arg0.start > arg2.reduceN
return[] elements < arg0.start
arg0.start > orig(arg1)
arg0.totalPacketNum > arg0.reduceN
arg0.totalPacketNum < arg2.start
arg0.totalPacketNum > orig(arg1)

arg0.reduceN < arg2.start
arg0.reduceN != arg2.totalPacketNum
arg2.start > arg2.totalPacketNum
arg2.start > arg2.reduceN
return[] elements < arg2.start
arg2.start > orig(arg1)
arg2.totalPacketNum > arg2.reduceN
==========
Models.getLenThreshold(double[]):::ENTER
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.TCP_lengthModel_in has only one value
this.UDP_lengthModel_in has only one value
this.TCP_lengthModel_out has only one value
this.UDP_lengthModel_out has only one value
this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

141, 4.731500155384394E-139, 6.88746071952711E-137, 9.632938386778845E-135, 1.2944886722124481E-132, 1.6713908569841767E-
130, 2.0734739156952656E-128, 2.4714989332903388E-126, 2.830506172326177E-124, 3.114656557653991E-122, 3.293056482470505E-
120, 3.345274400072606E-118, 3.2651902378701637E-116, 3.06217533335801E-114, 2.7592885161858923E-112, 2.3889681986943288E-
110, 1.98733255779926E-108, 1.5884677715892435E-106, 1.2199295119449431E-104, 9.002027941111209E-103, 6.38257779680831E-
101, 4.3481338612578574E-99, 2.8461752954003564E-97, 1.790083513379461E-95, 1.081779748857572E-93, 6.281441864364563E-
92, 3.504579075702084E-90, 1.8787522992744526E-88, 9.67746244377759E-87, 4.7897515198294255E-85, 2.2778506875201948E-
83, 1.0408774761200865E-81, 4.57022328964554E-80, 1.9281467849932478E-78, 7.816441854366254E-77, 3.0447102843313616E-
75, 1.1395996266774018E-73, 4.0985451512357487E-72, 1.4163805844422793E-70, 4.703322971239878E-69, 1.500742168071292E-
67, 4.601354949412003E-66, 1.3556442894777107E-64, 3.837851761151291E-63, 1.044035407920269E-61, 2.729160503714384E-
60, 6.85538836152681E-59, 1.6547278974442446E-57, 3.8380899722227855E-56, 8.554612443191246E-55, 1.832250454563194E-
53, 3.7711293863925446E-52, 7.458697810692291E-51, 1.4176289321406552E-49, 2.5892453410679414E-48, 4.544625830577738E-
47, 7.66550347630047E-46, 1.242522633748804E-44, 1.935497314930632E-43, 2.8974047864657866E-42, 4.168290045856985E-
41, 5.7629227861461015E-40, 7.657162601484798E-39, 9.777714729267097E-38, 1.1999292607814969E-36, 1.4152306378920357E-
35, 1.6041934887833055E-34, 1.7476281311984218E-33, 1.8298226779912348E-32, 1.8413730788580957E-31, 1.7809537264347663E-
30, 1.6555677499002913E-29, 1.4792116794103138E-28, 1.2703063618674378E-27, 1.0485455081500603E-26, 8.319032263289717E-
26, 6.344118035551926E-25, 4.650385864655815E-24, 3.276668760523394E-23, 2.219262072562808E-22, 1.4448540562459613E-
21, 9.04244147393558E-21, 5.440025214570166E-20, 3.1461316783789997E-19, 1.7491260077087374E-18, 9.348482131378308E-
18, 4.803354434177367E-17, 2.372682084186244E-16, 1.1267662839825846E-15, 5.14441063822216E-15, 2.2581518609276873E-
14, 9.530040710553601E-14, 3.8669534043945895E-13, 1.5086353811076154E-12, 5.659127976132414E-12, 2.0411430643408098E-
11, 7.07888778458995E-11, 2.36065943651551E-10, 7.569855476719277E-10, 2.33419506112745E-9, 6.921361791802057E-9,
1.9736064072978912E-8, 5.411954200392378E-8, 1.42719237772584E-7, 3.6195649720885026E-7, 8.828496132441961E-7, 2.0710240645411293
6, 4.672656227407467E-6, 1.013994968792008E-5, 2.116484349786022E-5, 4.249294778666767E-5, 8.206500128933713E-5, 1.524604272237740
4, 2.724804982484997E-4, 4.6850796130801063E-4, 7.750499222254525E-4, 0.0012336903764099878, 0.0018896663352045513,
0.0027855617321086512, 0.003952256419525375, 0.005398218297181328, 0.007099258979757903, 0.008991612935086298, 0.0109712354799931
0.012901278270128687, 0.014627843171027067, 0.016001782867529474, 0.0169022541417285, 0.017256718073602836, 0.017052606486148647,
0.016337883993674005, 0.015210615966186754, 0.013800453197510856, 0.012246754283354094, 0.010678375577960378, 0.00919902526834216
0.007880052229383216, 0.006760390611773864, 0.005851775735521873, 0.00514663108228486, 0.004626173301901328, 0.004267000431968937
0.004045341056670655, 0.003939448884174375, 0.003927133883680862, 0.003989791960875527, 0.004106110760860398, 0.0042537976784365
0.004409233893271828, 0.004548767852501136, 0.004650976767760023, 0.004699392177273306, 0.004684962433424812, 0.0046075065367958 0
0.004475636868206238, 0.004305041979997472, 0.004115491768913182, 0.003927289696519069, 0.003758020154952174 3, 0.0036202865723674
0.003520778629728057, 0.003460590617261679 6, 0.003436392106770937, 0.003441925751356038, 0.003469384249376831, 0.0035104270794641
0.003556826418150153 3, 0.003600883756093427 5, 0.003635789991912224 6, 0.003656030183018395 4, 0.003657820238571508 5,
0.003639474626813423 4, 0.003601584331367741, 0.003546934326602306 7, 0.003480175594733015 3, 0.00340733978550191, 0.003335307837032
0.003271308712959960 7, 0.003222453535044666 2, 0.003195245505040058, 0.003194987316677505 7, 0.00322505417457127, 0.003286099019834
0.003375365311612505, 0.003486347729006416, 0.003609020113257213 6, 0.003730733431237834 5, 0.003837706321908539 5, 0.00391684992176
0.003957555885795163, 0.003953076383831 83, 0.003901237912008196, 0.003804413770834822 6, 0.003668866000307968, 0.00350369494871997
0.003319671453586125, 0.003128178126462001 6, 0.002940385890414892 6, 0.002766682732551889 7, 0.002616288934884318, 0.00249695558050
0.002414653031616846 4, 0.002373203480892898 4, 0.002373879402375465 6, 0.002415056371816933 6, 0.002492051734984706 2,
0.002597281373244049, 0.002720817684716092, 0.002851341884163665 3, 0.002977378783370665, 0.003088617772950411 2, 0.003177091315442
0.003238015796767208, 0.003270187908018460 3, 0.003275939594436779, 0.003260743697579438, 0.003232597309324307 3, 0.003201281341315
0.003177523646061676, 0.003172022770908353 7, 0.003194268134325256, 0.003251149749021688, 0.003345479917463862 4, 0.003474703558360
0.003630179894004358 7, 0.003797406579274481 4, 0.003957395796152812, 0.004089128129254101, 0.004172690106278791, 0.004192459885414
0.004139639676951161, 0.004013579069713708, 0.003821646787138919 3, 0.003577782363340219 7, 0.003300167460976971, 0.003008604841950
0.002722153696949497 4, 0.002457385334626544 2, 0.002227379147171690 5, 0.002041366321134752 2, 0.001904808687494935 8,
0.001819689333046298 4, 0.001784864902599377 8, 0.001796438793703923 6, 0.001848207707216467 2, 0.001932272716921083,
0.002039874959744697, 0.002162426652551311, 0.002292594965978552, 0.002425206768578813, 0.002557720908795428 5, 0.0026900856336521
0.002823952944492780 7, 0.002961415588192438, 0.003103600447801940 3, 0.003249530185497432 6, 0.003395615541993630 4,
0.003535969189392850 5, 0.003663487463630321 6, 0.003771405287381695 5, 0.003854870529845209 6, 0.003912061102273386, 0.003944493699888
0.003956413568755276, 0.003953439021886528, 0.003940873967431521, 0.003922216162696538, 0.003898333200711099, 0.00386756197857642
0.003826678103446334, 0.003772386165491810 3, 0.003702808444164429 8, 0.003618464792621945 6, 0.003524243434981218 3,
0.003419691664537255, 0.003315896111482897 6, 0.003216074707717654, 0.003123622063654236 8, 0.003039871359859733 6, 0.002964252238 29
0.002894849022305 9, 0.002829081408348825 5, 0.002764266714177591, 0.002697947168202328, 0.002628005129401238, 0.002552677248779314
0.002470584919811142 2, 0.002380837009392964, 0.002283176939550553 7, 0.002178089332953146, 0.002066780686217427, 0.001951000608306
0.001832744965926407 5, 0.001713940748513732 5, 0.001596226429875275, 0.001480905864892779 8, 0.001369085792082721, 0.001261936355 51
0.001160960726446440 4, 0.001068211821775332, 9.862191361712134E-4, 9.178547892777621E-4, 8.659141958801867E-4, 8.326484091934924E-
4, 8.193085684633738E-4, 8.258240449821245E-4, 8.507033450924647E-4, 8.911978979917742E-4, 9.437058892951416E-4, 0.001004332040502
0.001069478393361436 7, 0.001136335716445065 4, 0.001203178681266639, 0.001269428657933580 4, 0.001335512134239499 4,
0.001402585290064559 9, 0.001472202358996824 4, 0.001545981935167257 5, 0.001625291937090446 5, 0.001710954562447147 7,
0.001802979950232294 9, 0.001900364525853309 3, 0.002001014515177268 6, 0.002101852612605417 3, 0.002199126408120428 5,
0.002288873463827705 3, 0.002367439451539789 8, 0.002431922964290105 7, 0.002480447198331054 5, 0.002512223014084513,
0.002527438687964703, 0.002527055279823826 3, 0.002515855445551814 5, 0.002485896944260093 3, 0.002449033931256072 5, 0.00240403243070
0.002352715282861119, 0.002296500563910737 5, 0.002236294715972692, 0.002172547207810443 7, 0.002105499205325969 6, 0.00203557983679
0.001963826633794812 3, 0.001892171458035774 9, 0.001823461616656799, 0.001761170071915266 5, 0.001708854758875569 7,

0.0016695118554435 61, 0.0016449993597891 77, 0.0016356780052114613, 0.0016403438251845 81, 0.0016564419038100815, 0.0016804837896900
0.0017085583119387927, 0.0017368274520972867, 0.0017619244966604142, 0.0017812070229654142, 0.0017928522628012, 0.0017958127139760
0.0017896745524294926, 0.001774478908787376, 0.0017505722495007765, 0.0017185415825433602, 0.0016792606100192234,
0.0016340290371534589, 0.0015847419845095 83, 0.0015339973851998842, 0.0014850512460644 8, 0.0014415683415477 23, 0.0014071798659337
0.0013849286954020254, 0.0013767324156264786, 0.0013830056738507025, 0.0014025518471646604, 0.0014327694927948804,
0.0014701415480928204, 0.0015109065920763936, 0.0015517680692774 39, 0.0015904867926540 25, 0.0016262237506090382, 0.0016595484343415
0.0016920939993987 49, 0.0017259141956695043, 0.0017626658543914 43, 0.0018027911472055652, 0.0018448926665899407, 0.0018854725514 1
0.0019191429197809 43, 0.0019393173804197613, 0.0019393281130434 41, 0.0019134360584124824, 0.0018584559929278276, 0.0017740854333 33
0.0016633839779022365, 0.0015323418917196677, 0.0013888942692495575, 0.0012418825299898 76, 0.0010992351288427577, 9.67352131585662
4, 8.502136144527702E-4, 7.493178741883239E-4, 6.640628417020584E-4, 5.92462964332234E-4, 5.319913114274598E-4, 4.8034341552616533
4, 4.359746874202063E-4, 3.9834331249848613E-4, 3.678694809788672E-4, 3.456798158984995E-4, 3.332335133009281E-4,
3.319250851923093E-4, 3.427363444218923E-4, 3.659793193504541E-4, 4.011432808703266E-4, 4.468404385101648 6E-4, 5.008383811631753E
4, 5.601701700382261E-4, 6.213187447017 05E-4, 6.804737534106662E-4, 7.338512357057908E-4, 7.780499013216787E-4, 8.103978262674 03E-
4, 8.292296858742524E-4, 8.340362510555869E-4, 8.254489121082E-4, 8.050587328345 43E-4, 7.751108650717634E-4, 7.381466085205525E-
4, 6.966751691489361E-4, 6.529413351886163E-4, 6.088198043552862E-4, 5.658252124142802E-4, 5.251942937062946E-4, 4.879837234032 51E
4, 4.551359905631671E-4, 4.2748928203338674E-4, 4.05734049713763E-4, 3.903377447882791E-4, 3.814647819347341E-4, 3.789129377090753
4, 3.820768761669764E-4, 3.899416899459897E-4, 4.0110801057248154E-4, 4.138538425705775E-4, 4.26241457566334E-4, 4.362748697048391
4, 4.421023718277864E-4, 4.422420465087339E-4, 4.3579259688748763E-4, 4.225845385470117E-4, 4.032324034681412E-4,
3.790670539062601E-4, 3.51953624019864E-4, 3.2402713897421555E-4, 2.97396450727050 3E-4, 2.738723068573093E-4, 2.547660814060243E-
4, 2.4078559108901699E-4, 2.3203042897568 54E-4, 2.2806885591161106E-4, 2.2806707167366127E-4, 2.3094016254909211E-
4, 2.3550076743524475E-4, 2.4059024987979023E-4, 2.451850134775436E-4, 2.4847504600246975E-4, 2.4991381846486586E-
4, 2.4924003793847047E-4, 2.4647356654566 84E-4, 2.4188965129442368E-4, 2.3597614700773952E-4, 2.2937693976407406E-
4, 2.2282226000214301E-4, 2.1704540682716467E-4, 2.1268784194192469E-4, 2.1020093030655094E-4, 2.0976030679904705E-
4, 2.112135900170844E-4, 2.1408001934467523E-4, 2.1761039141417 95E-4, 2.2090006838372523E-4, 2.2303240267330772E-
4, 2.2322072743956217E-4, 2.2091748968237 15E-4, 2.1587247141567 93E-4, 2.0812806322863427E-4, 1.9797528945985553E-
4, 1.8587896581901062E-4, 1.7240203506359221E-4, 1.5814325361435412E-4, 1.436955438803984E-4, 1.296224797642684E-
4, 1.1644434104516672E-4, 1.0462396380697229E-4, 9.454576268412863E-5, 8.648709191809737E-5, 8.058730759125847E-5,
7.682434777283488E-5, 7.500979591734337E-5, 7.481064482156735E-5, 7.57999492551063E-5, 7.75309719643596E-5, 7.96226032028747E-
5, 8.183983441181913E-5, 8.415292332765869E-5, 8.676225856943825E-5, 9.008142217789898E-5, 9.467694915033153E-5, 1.011688642157731
4, 1.1010138730529323E-4, 1.2179903697575837E-4, 1.3623004689240696E-4, 1.529052295993704E-4, 1.7084291005750902E-
4, 1.8862522385704 69E-4, 2.0455538703978944E-4, 2.1690094891310894E-4, 2.2418095776744593E-4, 2.2543518494073442E-
4, 2.2041026198735 82E-4, 2.0961483098164371E-4, 1.9422969638566 45E-4, 1.7589842880364956E-4, 1.5645494706864382E-
4, 1.3765635640053853E-4, 1.2097852054289518E-4, 1.0750419283079451E-4, 9.79008666503951E-5, 9.246048047760164E-5,
9.116386418975 13E-5, 9.374003100838916E-5, 9.970788901513955E-5, 1.0840601452063162E-4, 1.1902630099206842E-4, 1.3066585320474567
4, 1.4240064754369 01E-4, 1.5337075805730 33E-4, 1.6285777422369655E-4, 1.7033513688846942E-4, 1.7548143577442114E-
4, 1.7816067665896642E-4, 1.7838543517963165E-4, 1.7628308612273265E-4, 1.7208004872142486E-4, 1.6610677835223215E-
4, 1.5881274951535512E-4, 1.5077208560956 57E-4, 1.4266074291933704E-4, 1.351953652605042E-4, 1.290384774111004E-4,
1.2468886494641657E-4, 1.2238436050256604E-4, 1.2204377933565853E-4, 1.2326578097439164E-4, 1.253883231363422E-4,
1.2759788661807433E-4, 1.2906703621318 3E-4, 1.2909442874226392E-4, 1.2722301708156689E-4, 1.233181700186848E-4, 1.175954643628004
4, 1.1059621377361445E-4, 1.0311637065562352E-4, 9.610073264772225E-5, 9.051883245669764E-5, 8.72407241643605E-5,
8.692950280767974E-5, 8.996298595811762E-5, 9.639082619521176E-5, 1.0592743882610716E-4, 1.179773348977724E-4, 1.3168833123492157
4, 1.4602863663789714E-4, 1.5988394176416295E-4, 1.7216863454795064E-4, 1.819409598970065E-4, 1.8850711990036555E-
4, 1.9149684307848234E-4, 1.9089518917379195E-4, 1.8702269597261535E-4, 1.8046649052565946E-4, 1.7197534064353787E-
4, 1.6233867170609487E-4, 1.5227170495510 95E-4, 1.4232634336375642E-4, 1.328417088974457E-4, 1.2394096230403876E-4,
1.1557335544309476E-4, 1.0759305073858 09E-4, 9.98597817347044E-5, 9.234198611182886E-5, 8.520187668174743E-5, 7.884482702137558E-
5, 7.392211813241322E-5, 7.128493158601542E-5, 7.189615978775581E-5, 7.67130317840026E-5, 8.655667532283662E-5, 1.019848267347513 7
4, 1.2318215189857605E-4, 1.4988036658274 43E-4, 1.8131824523854318E-4, 2.1624948888229005E-4, 2.5300344970058294E-
4, 2.8959902330931 28E-4, 3.2390537587547 54E-4, 3.5383530307228 03E-4, 3.7754955628055106E-4, 3.936455009312178E-4,
4.013028444108105E-4, 4.0036383305046605E-4, 3.913348579019873E-4, 3.7530900595301 85E-4, 3.538219488113172E-4, 3.2866373818676 07E
4, 3.016744698263001E-4, 2.7455179582009184E-4, 2.4869389521078 95E-4, 2.2509467185841985E-4, 2.0430045926815476E-
4, 1.8643027136002168E-4, 1.7125448749165083E-4, 1.5831929555245798E-4, 1.4709601483961648E-4, 1.3712873131811832E-
4, 1.2814889811978925E-4, 1.2013194994011998E-4, 1.1328054958373598E-4, 1.0793765743662255E-4, 1.0445136059850663E-
4, 1.0302800885834253E-4, 1.0361514647955107E-4, 1.0584868727176373E-4, 1.0908131029508668E-4, 1.1248626906686755E-
4, 1.1520973633316042E-4, 1.1653203624772508E-4, 1.1599753966711648E-4, 1.1348443914147104E-4, 1.0920484210427414E-
4, 1.0364580892192361E-4, 9.747622076766626E-5, 9.144831757619173E-5, 8.631619061123978E-5, 8.278033101074056E-5,
8.145368986142538E-5, 8.283630144436787E-5, 8.728521093226921E-5, 9.497329230556289E-5, 1.058406795139103E-4, 1.1955127087493498E
4, 1.3547073824082107E-4, 1.5268078929 29912E-4, 1.7003840244179336E-4, 1.8628061152025881E-4, 2.0016696048233336E-
4, 2.1064381815886856E-4, 2.1700764770237 05E-4, 2.1903890078311226E-4, 2.1707639582460323E-4, 2.1200648298821723E-
4, 2.0515382469716144E-4, 1.9808047984741 83E-4, 1.9232319347056186E-4, 1.891177683998561E-4, 1.8916925592502018E-
4, 1.9251931048984517E-4, 1.9854235527204 31E-4, 2.0607181435712 45E-4, 2.1362732163127276E-4, 2.1969145037416474E-
4, 2.2297624578036 13E-4, 2.2262703628268157E-4, 2.1833037508891265E-4, 2.1031805644783117E-4, 1.992825871639209E-
4, 1.8623526571220233E-4, 1.7234309124811 57E-4, 1.587755832285 99E-4, 1.4658072911393993E-4, 1.3659579542530593E-4,
1.2938854454182298E-4, 1.2522054206076517E-4, 1.2402692376515222E-4, 1.2541365891057606E-4, 1.2867908 0723757068E-4,

1.3287448196013923E-4, 1.3689404894455268E-4, 1.3961522592049704E-4, 1.4004756859585258E-4, 1.3747906677644007E-4,
1.3158540814163516E-4, 1.2247884622161313E-4, 1.1068540918369363E-4, 9.705558166410483E-5, 8.26284687701035E-5, 6.847830541325716
5, 5.557274353110707E-5, 4.4665336875862844E-5, 3.623325507637488E-5, 3.045978767741286E-5, 2.7253263734594942E-5,
2.6291216670733722E-5, 2.7080179186720412E-5, 2.902491353560309E-5, 3.1503372157146584E-5, 3.3943538938830565E-5,
3.589567933661055E-5, 3.709025230645374E-5, 3.747028630125971E-5, 3.718922163225235E-5, 3.657129063320398E-5, 3.603988716617866E-
5, 3.602738557655897E-5, 3.688481788398481E-5, 3.881006597841922E-5, 4.180873205096693E-5, 4.56940674182416E-5, 5.0123579657842716
5, 5.4662526400953405E-5, 5.886005284818947E-5, 6.232276821616348E-5, 6.47726070763469E-5, 6.607983448390429E-5, 6.626695483616886
5, 6.548434958651268E-5, 6.39633267323978E-5, 6.195656959398477E-5, 5.967903785971012E-5, 5.726313366243597E-5, 5.4739407245335015
5, 5.204807225347627E-5, 4.907833655422737E-5, 4.572449210743998E-5, 4.1942668554332804E-5, 3.779205152588941E-5,
3.344924478259995E-5, 2.9192421938699655E-5, 2.53600169229732E-5, 2.229429755089271E-5, 2.0282026367115217E-5, 1.95030175022278E-
5, 1.9994329339697405E-5, 2.1634701770757594E-5, 2.4151412640536916E-5, 2.714969233308966E-5, 3.016236843845304E-
5, 3.2714022281186647E-5, 3.439010712381323E-5, 3.489863578113836E-5, 3.411187379001532E-5, 3.2078915537211116E-5,
2.900658251335153E-5, 2.521385539166005E-5, 2.1071462613772338E-5, 1.6941168832449468E-5, 1.3127940362764351E-5, 9.85329095551559
6, 7.251620384625635E-6, 5.385382322376234E-6, 4.271016599391116E-6, 3.906343708162147E-6, 4.2912633721232945E-6,
5.436376182806878E-6, 7.357758928270751E-6, 1.0059914051409074E-5, 1.3512307248245645E-5, 1.76272529158797E-5, 2.2247447016326538
5, 2.7149557676895525E-5, 3.206592511359135E-5, 3.672052602503616E-5, 4.086982548942958E-5, 4.4336267511302996E-5,
4.7023621420062306E-5, 4.8909280990440966E-5, 5.00170011432285E-5, 5.038114498688449E-5, 5.001698671166474 3E-5, 4.890923490108206
5, 4.70234922960956E-5, 4.4335923094801404E-5, 4.0868943688454876E-5, 3.671835711843306E-5, 3.2060799633355646E-5,
2.7137920278062124E-5, 2.2222060370814245E-5, 1.757404420601965E-5, 1.3405158274773905E-5, 9.852603721112165E-6, 6.97238606994671
6, 4.748091823762641E-6, 3.1101810392556746E-6, 1.9591188707233857E-6, 1.1866030805859301E-6, 6.913670123930401E-
7, 3.884754624591 46E-7, 2.1291124241975207E-7, 1.1926064459510177E-7, 7.952398360914685E-8, 8.078061420693403E-8,
1.2391776419125612E-7, 2.2407663201588472E-7, 4.1293425086107574E-7, 7.42390970342881E-7, 1.2886217573968435E-6, 2.15479176995569
6, 3.470171947653351E-6, 5.383182942738018E-6, 8.046364292434154E-6, 1.1592642152000667E-5, 1.6104527352867 27E-5,
2.158061072088778E-5, 2.790616376772666E-5, 3.4835859789531076E-5, 4.199583204903081E-5, 4.890923437353662E-5, 5.504464508667889E
5, 5.988120288377637E-5, 6.29797649449887E-5, 6.404698092231923E-5, 6.297976494497918E-5, 5.988120288373188E-5, 5.504464508648691E
5, 4.8909234372742285E-5, 4.1995832045873444E-5, 3.48358597774729E-5, 2.790616372348153E-5, 2.1580610564904442E-5,
1.6104526824519 68E-5, 1.1592640432547694E-5, 8.046358916050416E-6, 5.383166790971133E-6, 3.4701253267973437E-6, 2.154662477939804
6, 1.2882772523616239E-6, 7.415090011386023E-7, 4.107648119765624E-7, 2.189494284551982 9E-7, 1.1227497706557278E-
7, 5.537777404506445E-8, 2.626850246693238E-8, 1.1981871918600643E-8, 5.254764904869093E-9, 2.215531402968553E-9,
8.979762537340598E-10, 3.498812269619042E-10, 1.3115534739552233E-10, 4.76743739277747E-11, 1.8026362153890266E-11,
1.0796628176650562E-11, 1.7882439031407876E-11, 4.714919364648731E-11, 1.2943673333588474E-10, 3.4450522722175906E-
10, 8.81825325761886E-10, 2.168913714268285E-9, 5.125484949199558E-9, 1.1637411128518465E-8, 2.5386689243950053E-
8, 5.320886349820675E-8, 1.0714949291327256E-7, 2.0731201752125474E-7, 3.853781227872543E-7, 6.883001376950374E-7,
1.1811277596429258E-6, 1.9473504612154458E-6, 3.084747207177483E-6, 4.694866665378789E-6, 6.865231200844193E-6, 9.645290128107799
6, 1.3019780148837283E-5, 1.6885745630992562E-5, 2.1040937943075904E-5, 2.519058595632884E-5, 2.8976099043117127E-
5, 3.202361974130835E-5, 3.4004056981189707E-5, 3.4691514894765907E-5, 3.40058814422099E-5, 3.202861598955599E-5,
2.8987689992291176E-5, 2.5215957022883225E-5, 2.1094143149778572E-5, 1.6992898780686174E-5, 1.3227107788990799E-5,
1.0030714712602 44E-5, 7.553660574207728E-6, 5.876338873562149E-6, 5.032979477664334E-6, 5.034266574988208E-6, 5.881119896924367E-
6, 7.565168705186001E-6, 1.0056054784103049E-5, 1.3280300501506392E-5, 1.7100042897405133E-5, 2.130145344788887E-
5, 2.5601334617322804E-5, 2.9675989973960396E-5, 3.3209743704762216E-5, 3.595323189057 2946E-5, 3.7776262095628674E-
5, 3.869892363371609E-5, 3.888885089771584E-5, 3.862138901519911E-5, 3.8210365576818446E-5, 3.792668185465751E-5,
3.792667819787416E-5, 3.8210349954183526E-5, 3.862134255348167E-5, 3.8888721662002744E-5, 3.869857918498365E-5, 3.777538028556294
5, 3.595106298082031E-5, 3.320461822080147E-5, 2.9664352563068908E-5, 2.557594792813349 8E-5, 2.1248244584402848E-
5, 1.6992893404494874E-5, 1.3072988483987798E-5, 9.670676661325847E-6, 6.876868567535571 5E-6, 4.6999921374734445E-
6, 3.0869161145670867E-6, 1.9482322736442074E-6, 1.181472220241448E-6, 6.884294176082348E-7, 3.8542474047586757E-
7, 2.0732816849124334E-7, 1.0715486910243656E-7, 5.321058289653724E-8, 2.5387217536896533E-8, 1.1637566917286378E-8,
5.1255283974406975E-9, 2.168922605048087E-9, 8.818163804436313E-10, 3.444615848480421E-10, 1.292801498021096 7E-10,
4.661774323870662E-11, 1.6150982510537246E-11, 5.376193872192289E-12, 1.7194088224796311E-12, 5.283377260574924E-
13, 1.559812259193252E-13, 4.424470112122897E-14, 1.205808743585125E-14, 3.1573579170343686E-15, 7.943235282542087E-
16, 1.9199912948398143E-16, 4.45891623963287E-17, 9.949186919592497E-18, 2.132917646426976E-18, 4.393279211951477E-
19, 8.694242401310261E-20, 1.653114382174869E-20, 3.0199676090808537E-21, 5.300659245140352E-22, 8.938933561007253E-
23, 1.448337582658515E-23, 2.254678810828237E-24, 3.3734025161728695E-25, 4.9348461555464614E-26, 1.3381910655307568E-
26, 4.934846155361237E-26, 3.373402515971216E-25, 2.2546788106188837E-24, 1.448337582449702E-23, 8.938933559006182E-
23, 5.3006592432979E-22, 3.0199676074509648E-21, 1.6531143807895547E-20, 8.69424238999755E-20, 4.393279203075559E-
19, 2.1329176397360208E-18, 9.949186871131627E-18, 4.458916205910157 6E-17, 1.9199912722931645E-16, 7.943235137708344E-
16, 3.157357827645033E-15, 1.2058086905785324E-14, 4.424469810126137E-14, 1.5598120938818142E-13, 5.283376391150685E-
13, 1.7194083831517118E-12, 5.3761917392746556E-12, 1.6150972561350425E-11, 4.66176986495449E-11, 1.292799578029847E-
10, 3.444607905245428E-10, 8.818132230858929E-10, 2.168910546961711E-9, 5.125484152745616E-9, 1.163741093609352E-
8, 2.538668919934436E-8, 5.320886348859341 6E-8, 1.0714949291283019E-7, 2.0731201752863115E-7, 3.8537812282180724E-
7, 6.883001378424308E-7, 1.181127760245247E-6, 1.9473504635784796E-6, 3.0847472160782125E-6, 4.694866697565399E-6,
6.865231312580701E-6, 9.645290500464211E-6, 1.301978133990794E-5, 1.6885749287775477E-5, 2.1040948717854597E-5, 2.519061642304498
5, 2.8976181705331688E-5, 3.202383492194908E-5, 3.400459433282795E-5, 3.4692801988932417E-5, 3.4008838003757603E-
5, 3.203512787209463E-5, 2.9001439114573142E-5, 2.524377866883448E-5, 2.1148082059792732E-5, 1.709305592910242E-5,

1.3405157743276777E-5, 1.033359010977702E-5, 8.046358916050503E-6, 6.642217113742261E-6, 6.169494408042384E-6, 6.642217113751772E-
6, 8.046358916095004E-6, 1.0333590109969003E-5, 1.3405157744071099E-5, 1.7093055932259777E-5, 2.114808207185082E-
5, 2.524377871307918E-5, 2.9001439270554354E-5, 3.203512840043227E-5, 3.4008839723165985E-5, 3.469280736512416E-5,
3.400461048380051E-5, 3.2023881539647725E-5, 2.897631098528949E-5, 2.5190960883835503E-5, 2.104183053107768E-5, 1.688791819832243-
5, 1.3024906824060683E-5, 9.656927911400288E-6, 6.890618001779959E-6, 4.748075561053553E-6, 3.1918967089889095E-6,
2.154662481097162E-6, 1.566505883022465E-6, 1.3766002754928628E-6, 1.5665058822727307E-6, 2.154662477949753E-6, 3.191896696932955-
6, 4.748075516809294E-6, 6.890617845798836E-6, 9.656927383062666E-6, 1.3024905104652303E-5, 1.68879128221307E-5, 2.104181438010511-
5, 2.5190914266136852E-5, 2.8976181705331688E-5, 3.20235370788572E-5, 3.400372867057743E-5, 3.469063845457719E-5,
3.400371423901324E-5, 3.2023490989496186E-5, 2.8976052581355097E-5, 2.519056984959103E-5, 2.1040932578940123E-5, 1.688574391474105,
1.3019779621293885E-5, 9.645289972318572E-6, 6.86523115664408E-6, 4.69486665333065E-6, 3.0847472040222583E-6, 1.947350460421561E-
6, 1.1811277594510105E-6, 6.883001376504483E-7, 3.8537812277722106E-7, 2.0731201751868248E-7, 1.0714949291069737E-
7, 5.32088634881541E-8, 2.538668919925742E-8, 1.1637410936076988E-8, 5.125484152742597E-9, 2.1689105469611807E-9,
8.818132230858036E-10, 3.4446079052452833E-10, 1.2927995780298244E-10, 4.661769864954456E-11, 1.6150972561350377E-
11, 5.376191739274649E-12, 1.719408383151711E-12, 5.283376391150684E-13, 1.559812093881814E-13, 4.424469810126137E-
14, 1.2058086905785324E-14, 3.157357827645033E-15, 7.943235137708344E-16, 1.9199912722931645E-16, 4.4589162059101576E-
17, 9.949186871131627E-18, 2.1329176397360208E-18, 4.393279203075559E-19, 8.694242389997548E-20, 1.653114380789534E-
20, 3.0199676074488767E-21, 5.300659243097793E-22, 8.938933540581667E-23, 1.4483374194607976E-23, 2.2546649574785994E-
24, 3.3722712449973575E-25, 4.846086976278897E-26, 6.6909553275708E-27, 8.875917908092199E-28, 1.131270973846889E-
28, 1.3853140284356237E-29, 1.629889045514653E-30, 1.8424515247958063E-31, 2.0010701823026274E-32, 2.0881265265827577E-
33, 2.093543058006342E-34, 2.0180672939051796E-35, 2.0323857818493385E-36, 2.0323857818493385E-36, 2.0180672939051796E-
35, 2.093543058006342E-34, 2.0881265265827577E-33, 2.0010701823026274E-32, 1.8424515247958063E-31, 1.629889045514653E-
30, 1.3853140284356237E-29, 1.131270973846889E-28, 8.875917908092199E-28, 6.6909553275708E-27, 4.846086976278897E-
26, 3.3722712449973575E-25, 2.2546649574785994E-24, 1.4483374194607976E-23, 8.938933540581667E-23, 5.300659243097793E-
22, 3.0199676074488767E-21, 1.653114380789534E-20, 8.694242389997548E-20, 4.393279203075559E-19, 2.1329176397360208E-
18, 9.949186871131627E-18, 4.4589162059101576E-17, 1.9199912722931645E-16, 7.943235137708344E-16, 3.157357827645033E-
15, 1.2058086905785324E-14, 4.424469810126137E-14, 1.559812093881814E-13, 5.283376391150684E-13, 1.719408383151711E-
12, 5.376191739274649E-12, 1.6150972561350377E-11, 4.661769864954456E-11, 1.2927995780298244E-10, 3.4446079052452833E-
10, 8.818132230858036E-10, 2.1689105469611807E-9, 5.125484152742597E-9, 1.1637410936076988E-8, 2.5386689199257418E-
8, 5.320886348815409E-8, 1.0714949291069728E-7, 2.0731201751868198E-7, 3.8537812277721805E-7, 6.883001376504317E-
7, 1.1811277594509235E-6, 1.947350460421122E-6, 3.0847472040201255E-6, 4.694866653320701E-6, 6.865231156599492E-6,
9.645289972126572E-6, 1.3019779620499557E-5, 1.688574391158374E-5, 2.1040932566882033E-5, 2.519056980534633E-5, 2.897605242537388-
5, 3.202349046115855E-5, 3.400371251960486E-5, 3.469063307838546E-5, 3.400371251960486E-5, 3.202349046115855E-5, 2.897605242537388-
5, 2.519056980534633E-5, 2.1040932566882033E-5, 1.688574391158374E-5, 1.3019779620499557E-5, 9.645289972126572E-6,
6.865231156599492E-6, 4.694866653320701E-6, 3.0847472040201255E-6, 1.947350460421122E-6, 1.1811277594509235E-6, 6.883001376504317-
7, 3.8537812277721805E-7, 2.0731201751868198E-7, 1.0714949291069728E-7, 5.320886348815409E-8, 2.5386689199257418E-8,
1.1637410936076988E-8, 5.125484152742597E-9, 2.1689105469611807E-9, 8.818132230858036E-10, 3.4446079052452833E-10,
1.2927995780298244E-10, 4.661769864954456E-11, 1.6150972561350377E-11, 5.376191739274649E-12, 1.719408383151711E-
12, 5.283376391150684E-13, 1.559812093881814E-13, 4.424469810126137E-14, 1.2058086905785324E-14, 3.157357827645033E-
15, 7.943235137708344E-16, 1.9199912722931645E-16, 4.4589162059101576E-17, 9.949186871131627E-18, 2.1329176397360208E-
18, 4.393279203075559E-19, 8.694242389997548E-20, 1.653114380789534E-20, 3.0199676074488767E-21, 5.300659243097793E-
22, 8.938933540581667E-23, 1.4483374194607976E-23, 2.2546649574785994E-24, 3.3722712449973575E-25, 4.846086976278897E-
26, 6.6909553275708E-27, 8.875917908092199E-28, 1.131270973846889E-28, 1.3853140284356234E-29, 1.629889045514617E-
30, 1.8424515247907424E-31, 2.0010701816070773E-32, 2.0881264347900316E-33, 2.0935314189906968E-34, 2.016649371527058E-
35, 1.8664206397070093E-36, 1.6596514214232952E-37, 1.4179223781214507E-38, 1.163901564568946E-39, 9.1792726108286E-
41, 6.955502397397228E-42, 5.063804869336075E-43, 3.5420414587198654E-44, 2.380447123787191E-45, 1.537062965437849E-
46, 9.535709212078012E-48, 5.6838493698423115E-49, 3.2550700694652257E-50, 1.791044367312114E-51, 9.46848656806728E-
53, 4.8093126520386236E-54, 2.3470031690332943E-55, 1.1004556926998E-56, 4.9574651225575716E-58, 2.1457296100570693E-
59, 8.923157154712205E-61, 3.5652526398549732E-62, 1.3686433737188218E-63, 5.0479902665468615E-65, 1.7888541765466197E-
66, 6.090593160277028E-68, 1.9923813249560384E-69, 6.262007089320793E-71, 1.8909622906949667E-72, 5.486310744003409E-
74, 1.5293474283435687E-75, 4.096001741817376E-77, 1.054004169485085E-78, 2.605869998411288E-80, 6.190010468903476E-
82, 1.4127269884987897E-83, 3.0977996121751494E-85, 6.526444075178308E-87, 1.3210770302575327E-88, 2.569258977964455E-
90, 4.800824662747621E-92, 8.61890368510078E-94, 1.4866763441763142E-95, 2.463820447951128E-97, 3.923104712650582E-
99, 6.001764713154106E-101, 8.82178059207161E-103, 1.245838531735167E-104, 1.6904233657038514E-106, 2.203725188274832E-
108, 2.7602448285437066E-110, 3.321742742940293E-112, 3.8407196341387565E-114, 4.2666541120006704E-116, 4.553973396064257E-
118, 4.670052491322254E-120, 4.601307479203744E-122, 4.355810425059214E-124, 3.961730281836719E-126, 3.4620159215246154E-
128, 2.9067082102560503E-130, 2.3447796407608932E-132, 1.8173176833982343E-134, 1.3532807095685038E-136, 9.682179961963562E-
139, 6.655591075147299E-141, 4.39570290220884E-143, 2.7893194367033434E-145, 1.7005773695846732E-147, 9.961454564547626E-
150, 5.606313018606829E-152, 3.031517967745701E-154, 1.574965960266907E-156, 7.86159049085964 8E-159, 3.770317157145745E-
161, 1.737295007972417E-163, 7.691259770839384E-166, 3.2715202183119997E-168, 1.3369955388555208E-170, 5.249748866010984E-
173, 1.980502155423184E-175, 7.178609528942127E-178, 2.499962890205022E-180, 8.364789505081965E-183, 2.6890859930557118E-
185, 8.305821340825077E-188, 2.4648398142664035E-190, 7.027858949244228E-193, 1.9252431785542454E-195, 5.067297131365684E-
198, 1.2814314216335197E-200, 3.1134549632521633E-203, 7.268052029497823E-206, 1.630127928050357E-208, 3.5128011730023157E-
211, 7.273001396396924E-214, 1.4467783242945037E-216, 2.7651489909671062E-219, 5.07765630130979E-222, 8.958520370276534E-

225, 1.5185793537032843E-227, 2.473243732429713E-230, 3.870121183265132E-233, 5.818491868493537E-236, 8.404744585683771E-239, 1.1664518846516838E-241, 1.555383039563573E-244, 1.9926733356940975E-247]
arg0[] elements >= 0.0
==========
Models.getLenThreshold(double[]):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)

this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg0[] == orig(arg0[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.TCP_lengthModel_out has only one value
this.UDP_lengthModel_out has only one value
this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

110, 1.98733255779926E-108, 1.5884677715892435E-106, 1.2199295119449431E-104, 9.002027941111209E-103, 6.38257779680831E-101, 4.3481338612578574E-99, 2.8461752954003564E-97, 1.790083513379461E-95, 1.081779748857572E-93, 6.281441864364563E-92, 3.504579075702084E-90, 1.8787522992744526E-88, 9.67746244377759E-87, 4.7897515198294255E-85, 2.2778506875201948E-83, 1.0408774761200865E-81, 4.57022328964554E-80, 1.928146784993247 8E-78, 7.816441854366254E-77, 3.0447102843313616E-75, 1.1395996266774018E-73, 4.0985451512357487E-72, 1.4163805844422793E-70, 4.703322971239878E-69, 1.500742168071292E-67, 4.601354949412003E-66, 1.3556442894777107E-64, 3.837851761151291E-63, 1.044035407920269E-61, 2.729160503714384E-60, 6.85538836152681E-59, 1.6547278974442446E-57, 3.8380899722227855E-56, 8.554612443191246E-55, 1.8322504544563194E-53, 3.7711293863925446E-52, 7.458697810692291E-51, 1.4176289321406552E-49, 2.5892453410679414E-48, 4.544625830577 38E-47, 7.66550347630047E-46, 1.242522633748804E-44, 1.935497314930632E-43, 2.8974047864657866E-42, 4.168290045856985E-41, 5.7629227861461015E-40, 7.657162601484798E-39, 9.777714729267097E-38, 1.1999292607814969E-36, 1.4152306378920357E-35, 1.6041934887833055E-34, 1.7476281311984218E-33, 1.8298226779912348E-32, 1.8413730788580957E-31, 1.7809537264347663E-30, 1.655567749900291 3E-29, 1.4792116794103138E-28, 1.2703063618674378E-27, 1.0485455081500603E-26, 8.319032263289717E-26, 6.344118035551926E-25, 4.650385864655815E-24, 3.276668760523394E-23, 2.219262072562808E-22, 1.4448540562459613E-21, 9.04244147393558E-21, 5.440025214570166E-20, 3.1461316783789997E-19, 1.7491260077087374E-18, 9.348482131378308E-18, 4.803354434177367E-17, 2.372682084186244E-16, 1.1267662839825846E-15, 5.14441063822216E-15, 2.2581518609276873E-14, 9.530040710553601E-14, 3.8669534043945895E-13, 1.5086353811076154E-12, 5.659127976132414E-12, 2.0411430643408098E-11, 7.07888778458995E-11, 2.36065943651551E-10, 7.569855476719277E-10, 2.33419506112745E-9, 6.921361791802057E-9, 1.9736064072978912E-8, 5.411954200392378E-8, 1.42719237772584E-7, 3.6195649720885026E-7, 8.828496132441961E-7, 2.0710240645411293 6, 4.672656227407467E-6, 1.013994968792008E-5, 2.116484349786022E-5, 4.249294778666767E-5, 8.206500128933713E-5, 1.524604272237740 4, 2.724804982484997E-4, 4.6850796130801063E-4, 7.750499222254525E-4, 0.0012336903764099878, 0.0018896663352045513, 0.0027855617321086512, 0.003952256419525375, 0.005398218297181328, 0.007099258979757903, 0.008991612935086298, 0.0109712354799931 0.012901278270128687, 0.014627843171027067, 0.016001782867529474, 0.0169022541417285, 0.017256718073602836, 0.017052606486148647, 0.016337883993674005, 0.015210615966186754, 0.013800453197510856, 0.012246754283354094, 0.010678375577960378, 0.009199025268342 16 0.007880052229383216, 0.006760390611773864, 0.005851775735521873, 0.00514663108228486, 0.004626173301901328, 0.004267000431968937 0.004045341056670655, 0.003938944888417 4375, 0.003927133883680862, 0.003989791960875527, 0.004106110760860398, 0.0042537976784365 0.004409233893271828, 0.004548767852501136, 0.004650976767760023, 0.004699392177273306, 0.004684962433424812, 0.0046075065367958 0 0.004475636868206238, 0.004305041979997472, 0.004115491768913182, 0.003927289696519069, 0.003758020154952 1743, 0.0036202865723674 0.003520778629728057, 0.003460590617261 6796, 0.003436392106770937, 0.003441925751356038, 0.003469384249376831, 0.0035104270794641 0.003556826418150 1533, 0.003600883756093 4275, 0.003635789991912 2246, 0.003656030183018 3954, 0.003657820238571 5085, 0.003639474626813 4234, 0.003601584331367741, 0.003546934326602 3067, 0.003480175594733 0153, 0.003407339785501 91, 0.003335307837032 0.003271308712959 9607, 0.003222453535044 6662, 0.003195245505040058, 0.003194987316677 5057, 0.003225054174571 27, 0.003286099019834 0.003375365311612505, 0.003486347729006416, 0.003609020113257 2136, 0.003730733431237 8345, 0.003837706321908 5395, 0.003916849921 76 0.003957555885795163, 0.003953076383831 83, 0.003901237912008196, 0.003804413770834 8226, 0.003668866000307968, 0.003503694948719 97 0.003319671453586125, 0.003128178126462 0016, 0.002940385890414 8926, 0.002766682732551 8897, 0.002616288934884314, 0.002496955580 50 0.002414653031616 8464, 0.002373203480892 8984, 0.002373879402375 4656, 0.002415056371816 9336, 0.002492051734984 7062, 0.002597281373244049, 0.002720817684716092, 0.002851341884163 6653, 0.002977378783370665, 0.003088617772950 4112, 0.003177091315442 0.003238015796767208, 0.003270187908018 4603, 0.003275939594436779, 0.003260743697579438, 0.003232597309324 3073, 0.003201281341315 0.003177523646061676, 0.003172022770908 3537, 0.003194268134325256, 0.003251149749021688, 0.003345479917463 8624, 0.003474703558360 0.003630179894004 3587, 0.003797406579274 4814, 0.003957395796152812, 0.004089128129254101, 0.004172690106278791, 0.004192459885414 0.004139639676951161, 0.004013579069713708, 0.003821646787138 9193, 0.003577782363340 2197, 0.003300167460976971, 0.003008604841950 0.002722153696949497, 0.002457385334626 5442, 0.002227379147171 6905, 0.002041366321134 7522, 0.001904808687494 9358, 0.001819689333046 2984, 0.001784864902599 3778, 0.001796438793703 9236, 0.001848207707216 4672, 0.001932272716921083, 0.002039874959744697, 0.002162426652551311, 0.002292594965978552, 0.002425206768578813, 0.002557720908795 4285, 0.002690085633652 1 0.002823952944492 7807, 0.002961415588192438, 0.003103600447801 9403, 0.003249530185497 4326, 0.003395615541993 6304, 0.003535969189392 85, 0.003663487463630 3216, 0.003771405287381 6955, 0.003854870529845 2096, 0.003912061102273386, 0.003944493699888 0.003956413568755276, 0.003953439021886528, 0.003940873967431521, 0.003922216162696538, 0.003898333200711099, 0.003867561978576 42 0.003826678103446334, 0.003772386165491 8103, 0.003702808444164 4298, 0.003618464792621 9456, 0.003522434349812 3823, 0.003419691664537255, 0.003315896111482 8976, 0.003216074707717 654, 0.003123622063654 2368, 0.003039871359859 7336, 0.002964252238 29 0.002894849022305 9, 0.002829081408348 8255, 0.002764266714177 591, 0.002697947168202328, 0.002628005129401238, 0.002552677248779314 0.002470584919811 1422, 0.002380837009392 964, 0.002283176939550 5537, 0.002178089332953 146, 0.002066780686217427, 0.001951000608306 0.001832744965926 4075, 0.001713940748513 7325, 0.001596226429875 275, 0.001480905864892 7798, 0.001369085792082721, 0.001261936355 51 0.001160968072644044, 0.001068211821775332, 9.862191361712134E-4, 9.178547892777621E-4, 8.659141958801867E-4, 8.326484091934924E-4, 8.193085684633738E-4, 8.258240449821245E-4, 8.507033450924647E-4, 8.911978979917742E-4, 9.437058892951416E-4, 0.001004332040502 0.001069478393361 4367, 0.001136335716445 0654, 0.001203178681266639, 0.001269428657933 5804, 0.001335512134239 4994, 0.001402585290064 5599, 0.001472202358996 8244, 0.001545981935167 2575, 0.001625291937090 4465, 0.001710954562447 1477, 0.001802979950232 2949, 0.001900364525853 3093, 0.002001014515177 2686, 0.002101852612605 4173, 0.002199126408120 4285, 0.002288873463827 7053, 0.002367439451539 7898, 0.002431922964290 1057, 0.002480447198331 0545, 0.002512223014084513, 0.002527438687964703, 0.002527055279823 8263, 0.002512585544555 181, 0.002485896944260 0933, 0.002449033931256 0725, 0.002404032430 70 0.002352715282861119, 0.002296500563910 7375, 0.002236294715972692, 0.002172547207810 4437, 0.002105499205325 9696, 0.002035579836 79 0.001963826633794 8123, 0.001892171458035 7749, 0.001823461616656 799, 0.001761170071915 2665, 0.001708854758875 5697, 0.001669511855443561, 0.001644999359789177, 0.001635678005211 4613, 0.001640343825184581, 0.001656441903810 0815, 0.001680483789690 0.001708558311938 7927, 0.001736827452097 2867, 0.001761924496660 4142, 0.001781207022965 4142, 0.0017928522628012, 0.001795812713976 0.001789674554529 4926, 0.001774478908787376, 0.001750572249500 7765, 0.001718541582543 3602, 0.001679260610019 2234,

0.0016340290371534589, 0.001584741984509583, 0.0015339973851998842, 0.00148505124606448, 0.001441568341547723, 0.0014071798659337
0.0013849286954020254, 0.0013767324156264786, 0.0013830056738507025, 0.0014025518471646604, 0.0014327694927948804,
0.0014701415480928204, 0.0015109065920763936, 0.001551768069277439, 0.001590486792654025, 0.0016262237506090382, 0.00165954843415
0.0016920939999398749, 0.0017259141956695043, 0.001762665854391443, 0.0018027911472055652, 0.0018448926665899407, 0.00188547255141
0.0019191942919780943, 0.0019393173804197613, 0.001939281113043441, 0.0019134360584124824, 0.0018584559929278276, 0.00177408543333
0.0016633839779022365, 0.0015323418917196677, 0.001388942692495575, 0.001241882529989876, 0.0010992351288427577, 9.67352131585662
4, 8.502136144527702E-4, 7.493178741883239E-4, 6.640628417020584E-4, 5.92462964332234E-4, 5.319913114274598E-4, 4.8034341552616533
4, 4.359746874202063E-4, 3.9834331249848613E-4, 3.678694809788672E-4, 3.456798158984995E-4, 3.332335133009281E-4,
3.319250851923093E-4, 3.427363444218923E-4, 3.659793193504541E-4, 4.011432808703266E-4, 4.468404385101648 6E-4, 5.008383811631753E
4, 5.601701700382261E-4, 6.21318744701705E-4, 6.804737534106662E-4, 7.338512357057908E-4, 7.780499013216787E-4, 8.10397826267403E-
4, 8.292296858742524E-4, 8.340362510555869E-4, 8.254489121082E-4, 8.05058732834543E-4, 7.751108650717634E-4, 7.381466085205525E-
4, 6.966751691489361E-4, 6.529413351886163E-4, 6.088198043552862E-4, 5.658252124142802E-4, 5.251942937062946E-4, 4.87983723403251E
4, 4.551359905631671E-4, 4.2748928203338674E-4, 4.05734049713763E-4, 3.9033774478827 91E-4, 3.814647819347341E-4, 3.789129377090753
4, 3.820768761669764E-4, 3.899416899459897E-4, 4.0110801057248154E-4, 4.138538425705775E-4, 4.26241457566334E-4, 4.362748697048391
4, 4.421023718277864E-4, 4.422420465087339E-4, 4.3579259688748763E-4, 4.225845385470117E-4, 4.032324034681412E-4,
3.790670539062601E-4, 3.51953624019864E-4, 3.2402713897421555E-4, 2.97396450727 0503E-4, 2.738723068573093E-4, 2.547660814060243E-
4, 2.4078559108901699E-4, 2.3203042289756854E-4, 2.2806885591161106E-4, 2.2806707167366127E-4, 2.3094016254909211E-
4, 2.3550076743524475E-4, 2.4059024987979023E-4, 2.451850134775436E-4, 2.4847504600246975E-4, 2.4991381846486586E-
4, 2.4924003793847047E-4, 2.464735665456684E-4, 2.4188965129442368E-4, 2.3597614700773952E-4, 2.2937693976407406E-
4, 2.2282226000214301E-4, 2.1704540682716467E-4, 2.1268784194192469E-4, 2.1020093030655094E-4, 2.0976030679904705E-
4, 2.112135900170844E-4, 2.1408001934467523E-4, 2.176103914141795E-4, 2.2090006838372523E-4, 2.2303240267330772E-
4, 2.2322072743956217E-4, 2.2091784896823715E-4, 2.158724714156793E-4, 2.0812806322863427E-4, 1.9797528945985553E-
4, 1.8587896581901062E-4, 1.7240203506359221E-4, 1.5814325361435412E-4, 1.436955438803984E-4, 1.296224797642684E-
4, 1.1644434104516672E-4, 1.0462396380697229E-4, 9.454576268412863E-5, 8.648709191809737E-5, 8.058730759125847E-5,
7.682434777283488E-5, 7.500979591734337E-5, 7.481064482156735E-5, 7.57999492551063E-5, 7.75309719643596E-5, 7.96226032028747E-
5, 8.183983441181913E-5, 8.415292332765869E-5, 8.676225856943825E-5, 9.008142217789898E-5, 9.467694915033153E-5, 1.011688642157731
4, 1.1010138730529323E-4, 1.2179903697575837E-4, 1.3623004689240696E-4, 1.529052295993704E-4, 1.7084291005750902E-
4, 1.886252238570469E-4, 2.0455538703978944E-4, 2.1690094891310894E-4, 2.2418095776744593E-4, 2.2543518494073442E-
4, 2.204102619873582E-4, 2.0961483098164371E-4, 1.942296963856645E-4, 1.7589842880364956E-4, 1.5645494706864382E-
4, 1.3765635640053853E-4, 1.2097852054289518E-4, 1.0750419283079451E-4, 9.79008666503951E-5, 9.246048047760164E-5,
9.11638641897513E-5, 9.374003100838916E-5, 9.970788901513955E-5, 1.0840601452063162E-4, 1.1902630099206842E-4, 1.3066585320474567
4, 1.424006475436901E-4, 1.533707580573033E-4, 1.6285777422369655E-4, 1.7033513688846942E-4, 1.7548143577442114E-
4, 1.7816067665896642E-4, 1.7838543517963165E-4, 1.7628308612273265E-4, 1.7208004872142486E-4, 1.6610677835223215E-
4, 1.5881274951535512E-4, 1.507720856095657E-4, 1.4266074291933704E-4, 1.351953652605042E-4, 1.290384774111004E-4,
1.2468886494641657E-4, 1.2238436050256604E-4, 1.2204377933565853E-4, 1.2326578097439164E-4, 1.253883231363422E-4,
1.2759788661807433E-4, 1.29067036213183E-4, 1.2909442874226392E-4, 1.2722301708156689E-4, 1.233181700186848E-4, 1.175954643628004
4, 1.1059621377361445E-4, 1.0311637065562352E-4, 9.610073264772225E-5, 9.051883245669764E-5, 8.72407241643605E-5,
8.692950280767974E-5, 8.996298595811762E-5, 9.639082619521176E-5, 1.0592743882610716E-4, 1.179773348977724E-4, 1.3168833123492157
4, 1.4602863663789714E-4, 1.5988394176416295E-4, 1.7216863454795064E-4, 1.819409598970065E-4, 1.8850711990036555E-
4, 1.9149684307848234E-4, 1.9089518917379195E-4, 1.8702269597261535E-4, 1.8046649052565946E-4, 1.7197534064353787E-
4, 1.6233867170609487E-4, 1.522717049551095E-4, 1.4232634336375642E-4, 1.328417088974457E-4, 1.2394096230403876E-4,
1.1557335544309476E-4, 1.075930507385809E-4, 9.98597817347044E-5, 9.234198611182886E-5, 8.520187668174743E-5, 7.884482702137558E-
5, 7.392211813241322E-5, 7.128493158601542E-5, 7.189615978775581E-5, 7.67130317840026E-5, 8.655667532283662E-5, 1.0198482673475137
4, 1.2318215189857605E-4, 1.498803665827443E-4, 1.8131824523854318E-4, 2.1624948888229005E-4, 2.5300344970058294E-
4, 2.895990233093128E-4, 3.239053758754754E-4, 3.538353030722803E-4, 3.7754955628055106E-4, 3.936455009312178E-4,
4.013028444108105E-4, 4.0036383305046605E-4, 3.913348579019873E-4, 3.753090059530185E-4, 3.538219488113172E-4, 3.286637381867607E
4, 3.016744698263001E-4, 2.7455179582009184E-4, 2.486938952107895E-4, 2.2509467185841985E-4, 2.0430045926815476E-
4, 1.8643027136002168E-4, 1.712544874916508 3E-4, 1.5831925955245798E-4, 1.4709601483961648E-4, 1.3712873131811832E-
4, 1.2814889811978925E-4, 1.2013194994011998E-4, 1.1328054958373598E-4, 1.0793765743662255E-4, 1.0445136059850663E-
4, 1.0302800885834253E-4, 1.0361514647955107E-4, 1.0584868727176373E-4, 1.0908131029508668E-4, 1.1248626906686755E-
4, 1.1520973633316042E-4, 1.165320362477250 8E-4, 1.1599753966711648E-4, 1.1348443914147104E-4, 1.0920484210427414E-
4, 1.0364580892192361E-4, 9.747622076766626E-5, 9.144831757619173E-5, 8.631619061123978E-5, 8.278033101074056E-5,
8.145368986142538E-5, 8.283630144436787E-5, 8.728521093226921E-5, 9.497329230556289E-5, 1.058406795139103E-4, 1.1955127087493498E
4, 1.3547073824082107E-4, 1.526807892292912E-4, 1.7003840244179336E-4, 1.8628061152025881E-4, 2.0016696048233336E-
4, 2.1064381815886856E-4, 2.170076477023705E-4, 2.1903890078311226E-4, 2.1707639582460323E-4, 2.1200648298821723E-
4, 2.0515382469716144E-4, 1.980804978474183E-4, 1.9232319347056186E-4, 1.891177683998561E-4, 1.8916925592502018E-
4, 1.9251931048984517E-4, 1.985423552720431E-4, 2.060718143571245E-4, 2.1362732163127276E-4, 2.1969145037416474E-
4, 2.229762457803613E-4, 2.2262703628268157E-4, 2.1833037508891265E-4, 2.1031805644783117E-4, 1.992825871639209E-
4, 1.8623526571220233E-4, 1.723430912481157E-4, 1.58775583228599E-4, 1.4658072911393993E-4, 1.3659579542530593E-4,
1.2938854454182298E-4, 1.2522054206076517E-4, 1.2402692376515222E-4, 1.2541365891057606E-4, 1.2867980723757068E-4,
1.3287448196013923E-4, 1.3689404894455268E-4, 1.3961522592049704E-4, 1.4004756859585258E-4, 1.3747906677644007E-4,
1.3158540814163516E-4, 1.2247884622161313E-4, 1.1068540918369363E-4, 9.705558166410483E-5, 8.26284687701035E-5, 6.847830541325716
5, 5.557274353110707E-5, 4.4665336875862844E-5, 3.623325507637488E-5, 3.045978767741286E-5, 2.7253263734594942E-5,

2.6291216670733722E-5, 2.7080179186720412E-5, 2.902491353560309E-5, 3.1503372157146584E-5, 3.3943538938830565E-5,
3.589567933661055E-5, 3.709025230645374E-5, 3.747028630125971E-5, 3.718922163225235E-5, 3.657129063320398E-5, 3.603988716617866E-
5, 3.602738557655897E-5, 3.688481788398481E-5, 3.881006597841922E-5, 4.180873205096693E-5, 4.56940674182416E-5, 5.0123579657842716
5, 5.4662526400953405E-5, 5.886005284818947E-5, 6.232276821616348E-5, 6.47726070763469E-5, 6.607983448390429E-5, 6.626695483616886
5, 6.548434958651268E-5, 6.39633267323978E-5, 6.195656959398477E-5, 5.967903785971012E-5, 5.726313366243597E-5, 5.4739407245335015
5, 5.204807225347627E-5, 4.907833655422737E-5, 4.572449210743998E-5, 4.1942668554332804E-5, 3.779205152588941E-5,
3.344924478259995E-5, 2.9192421938699655E-5, 2.53600169229732E-5, 2.229429755089271E-5, 2.0282026367115217E-5, 1.95030175022278E-
5, 1.9994329339697405E-5, 2.1634701770757594E-5, 2.4151412640536916E-5, 2.714969233308966E-5, 3.016236843845304E-
5, 3.2714022281186647E-5, 3.439010712381323E-5, 3.489863578113836E-5, 3.411187379001532E-5, 3.2078915537211116E-5,
2.900658251335153E-5, 2.521385539166005E-5, 2.1071462613772338E-5, 1.6941168832449468E-5, 1.3127940362764351E-5, 9.85329095551559
6, 7.251620384625635E-6, 5.385382322376234E-6, 4.271016599391116E-6, 3.906343708162147E-6, 4.2912633721232945E-6,
5.436376182806878E-6, 7.357758928270751E-6, 1.0059914051409074E-5, 1.3512307248245645E-5, 1.76272529158797E-5, 2.2247447016326538
5, 2.7149557676895525E-5, 3.206592511359135E-5, 3.672052602503616E-5, 4.086982548942958E-5, 4.4336267511302996E-5,
4.7023621420062306E-5, 4.8909280990440966E-5, 5.00170011432285E-5, 5.038114498688449E-5, 5.0016986711664743E-5, 4.890923490108206
5, 4.702349229609 56E-5, 4.4335923094801404E-5, 4.0868943688454876E-5, 3.671835711843306E-5, 3.2060799633355646E-5,
2.7137920278062124E-5, 2.2222060370814245E-5, 1.757404420601965E-5, 1.3405158274773905E-5, 9.852603721112165E-6, 6.97238606994671
6, 4.748091823762641E-6, 3.1101810392556746E-6, 1.9591188707233857E-6, 1.1866030805859301E-6, 6.913670123930401E-
7, 3.884754624591 46E-7, 2.1291124241975207E-7, 1.1926064459510177E-7, 7.952398360914685E-8, 8.078061420693403E-8,
1.2391776419125612E-7, 2.2407663201588472E-7, 4.1293425086107574E-7, 7.42390970342881E-7, 1.2886217573968435E-6, 2.15479176995569
6, 3.470171947653351E-6, 5.383182942738018E-6, 8.046364292434154E-6, 1.1592642152000667E-5, 1.610452735286727E-5,
2.158061072088778E-5, 2.790616376772666E-5, 3.4835859789531076E-5, 4.199583204903081E-5, 4.890923437353662E-5, 5.504464508667889E
5, 5.988120288377637E-5, 6.29797649449887E-5, 6.404698092231923E-5, 6.297976494497918E-5, 5.988120288373188E-5, 5.504464508648691E
5, 4.8909234372742285E-5, 4.1995832045873444E-5, 3.48358597774729E-5, 2.790616372348153E-5, 2.1580610564904442E-5,
1.610452682451968E-5, 1.1592640432547694E-5, 8.046358916050416E-6, 5.383166790971133E-6, 3.4701253267973437E-6, 2.154662477939804
6, 1.2882772523616239E-6, 7.415090011386023E-7, 4.107648119765624E-7, 2.1894942845519829E-7, 1.1227497706557278E-
7, 5.537777404506445E-8, 2.626850246693238E-8, 1.1981871918600643E-8, 5.254764904869093E-9, 2.215531402968553E-9,
8.979762537340598E-10, 3.498812269619042E-10, 1.3115534739552233E-10, 4.76743739277747E-11, 1.8026362153890266E-11,
1.0796628176650562E-11, 1.7882439031407876E-11, 4.714919364648731E-11, 1.2943673333588474E-10, 3.4450522722175906E-
10, 8.81825325761886E-10, 2.1689137142682 85E-9, 5.125484949199558E-9, 1.1637411128518465E-8, 2.5386689243950053E-
8, 5.320886349820675E-8, 1.0714949291327256E-7, 2.0731201752125474E-7, 3.853781227872543E-7, 6.883001376950374E-7,
1.1811277596429258E-6, 1.9473504612154458E-6, 3.084747207177483E-6, 4.694866665378789E-6, 6.865231200844193E-6, 9.645290128107799
6, 1.3019780148837283E-5, 1.6885745630992562E-5, 2.1040937943075904E-5, 2.519058595632884E-5, 2.8976099043117127E-
5, 3.2023619741308 35E-5, 3.4004056981189707E-5, 3.4691514894765907E-5, 3.40058814422099E-5, 3.202861598955599E-5,
2.8987689992291176E-5, 2.5215957022883225E-5, 2.1094143149778572E-5, 1.6992898780686174E-5, 1.3227107788990799E-5,
1.0030714712602 44E-5, 7.553660574207728E-6, 5.876338873562149E-6, 5.032979477664334E-6, 5.034266574988208E-6, 5.881119896924367E-
6, 7.565168705186001E-6, 1.0056054784103049E-5, 1.3280300501506392E-5, 1.7100042897405133E-5, 2.130145344788887E-
5, 2.5601334617322804E-5, 2.9675989973960396E-5, 3.3209743704762216E-5, 3.5953231890572946E-5, 3.7776262095628674E-
5, 3.8698923633716 09E-5, 3.888885089771584E-5, 3.862138901519911E-5, 3.8210365576818446E-5, 3.792668185465751E-5,
3.7926678197874 16E-5, 3.8210349954183526E-5, 3.862134255348167E-5, 3.8888721662002744E-5, 3.869857918498365E-5, 3.777538028556294
5, 3.5951062980820 31E-5, 3.320461822080147E-5, 2.9664352563068908E-5, 2.5575947928133498E-5, 2.1248244584402848E-
5, 1.6992893404494874E-5, 1.3072988483987798E-5, 9.670676661325847E-6, 6.8768685675355715E-6, 4.6999921374734445E-
6, 3.0869161145670867E-6, 1.9482322736442074E-6, 1.181472220241448E-6, 6.884294176082348E-7, 3.8542474047586757E-
7, 2.0732816849124334E-7, 1.0715486910243656E-7, 5.321058289653724E-8, 2.5387217536896533E-8, 1.1637566917286378E-8,
5.1255283974406975E-9, 2.168922605048087E-9, 8.818163804436313E-10, 3.444615848480421E-10, 1.2928014980210967E-10,
4.6617743238706 62E-11, 1.6150982510537246E-11, 5.376193872192289E-12, 1.7194088224796311E-12, 5.283377260574924E-
13, 1.5598122591932 52E-13, 4.424470112122897E-14, 1.205808743585125E-14, 3.1573579170343686E-15, 7.943235282542087E-
16, 1.9199912948398143E-16, 4.45891623963287E-17, 9.949186919592497E-18, 2.132917646426976E-18, 4.393279211951477E-
19, 8.694242401310261E-20, 1.653114382174869E-20, 3.0199676090808537E-21, 5.300659245140352E-22, 8.938933561007253E-
23, 1.448337582658515E-23, 2.254678810828237E-24, 3.3734025161728695E-25, 4.9348461555464614E-26, 1.3381910655307568E-
26, 4.9348461553612 37E-26, 3.373402515971216E-25, 2.2546788106188837E-24, 1.448337582449702E-23, 8.938933559006182E-
23, 5.3006592432979E-22, 3.0199676074509648E-21, 1.6531143807895547E-20, 8.69424238999755E-20, 4.393279203075559E-
19, 2.1329176397360208E-18, 9.949186871131627E-18, 4.4589162059101576E-17, 1.9199912722931645E-16, 7.943235137708344E-
16, 3.157357827645033E-15, 1.2058086905785324E-14, 4.424469810126137E-14, 1.5598120938818142E-13, 5.283376391150685E-
13, 1.7194083831517118E-12, 5.3761917392746556E-12, 1.6150972561350425E-11, 4.66176986495449E-11, 1.292799578029847E-
10, 3.444607905245428E-10, 8.818132230858929E-10, 2.168910546961711E-9, 5.125484152745616E-9, 1.163741093609352E-
8, 2.538668919934436E-8, 5.320886348859341 6E-8, 1.0714949291283019E-7, 2.0731201752863115E-7, 3.8537812282180724E-
7, 6.883001378424308E-7, 1.181127760245247E-6, 1.9473504635784796E-6, 3.0847472160782125E-6, 4.694866697565399E-6,
6.865231312580701E-6, 9.645290500464211E-6, 1.301978133990794E-5, 1.6885749287775477E-5, 2.1040948717854597E-5, 2.519061642304498
5, 2.8976181705331688E-5, 3.202383492194908E-5, 3.400459433282795E-5, 3.4692801988932417E-5, 3.4008838003757603E-
5, 3.2035127872094 63E-5, 2.9001439114573142E-5, 2.524377866883448E-5, 2.1148082059792732E-5, 1.709305592910242E-5,
1.3405157743276777E-5, 1.033359010977702E-5, 8.046358916050503E-6, 6.642217113742261E-6, 6.169494408042384E-6, 6.642217113751772E
6, 8.046358916095004E-6, 1.0333590109969003E-5, 1.3405157744071099E-5, 1.7093055932259777E-5, 2.114808207185082E-
5, 2.5243778713079 18E-5, 2.9001439270554354E-5, 3.203512840043227E-5, 3.4008839723165985E-5, 3.4692807365124 16E-5,

3.400461048380051E-5, 3.2023881539647725E-5, 2.897631098528949E-5, 2.5190960883835503E-5, 2.104183053107768E-5, 1.688791819832243
5, 1.3024906824060683E-5, 9.656927911400288E-6, 6.890618001779959E-6, 4.748075561053553E-6, 3.1918967089889095E-6,
2.154662481097162E-6, 1.566505883022465E-6, 1.3766002754928628E-6, 1.5665058822727307E-6, 2.154662477949753E-6, 3.191896696932955
6, 4.748075516809294E-6, 6.890617845798836E-6, 9.656927383062666E-6, 1.3024905104652303E-5, 1.68879128221307E-5, 2.104181438010511
5, 2.5190914266136852E-5, 2.8976181705331688E-5, 3.20235370788572E-5, 3.400372867057743E-5, 3.469063845457719E-5,
3.400371423901324E-5, 3.2023490989496186E-5, 2.8976052581355097E-5, 2.519056984959103E-5, 2.1040932578940123E-5, 1.68857439147410
5, 1.3019779621293888E-5, 9.645289972318572E-6, 6.86523115664408E-6, 4.69486665333065E-6, 3.0847472040222583E-6, 1.947350460421561E
6, 1.1811277594510105E-6, 6.883001376504483E-7, 3.8537812277722106E-7, 2.0731201751868248E-7, 1.0714949291069737E-
7, 5.32088634881541E-8, 2.538668919925742E-8, 1.1637410936076988E-8, 5.125484152742597E-9, 2.1689105469611807E-9,
8.818132230858036E-10, 3.4446079052452833E-10, 1.2927995780298244E-10, 4.661769864954456E-11, 1.6150972561350377E-
11, 5.376191739274649E-12, 1.719408383151711E-12, 5.283376391150684E-13, 1.559812093881814E-13, 4.424469810126137E-
14, 1.2058086905785324E-14, 3.157357827645033E-15, 7.943235137708344E-16, 1.9199912722931645E-16, 4.4589162059101576E-
17, 9.949186871131627E-18, 2.1329176397360208E-18, 4.393279203075559E-19, 8.694242389997548E-20, 1.653114380789534E-
20, 3.0199676074488767E-21, 5.300659243097793E-22, 8.938933540581667E-23, 1.4483374194607976E-23, 2.2546649574785994E-
24, 3.3722712449973575E-25, 4.846086976278897E-26, 6.6909553275708E-27, 8.875917908092199E-28, 1.131270973846889E-
28, 1.3853140284356237E-29, 1.629889045514653E-30, 1.8424515247958063E-31, 2.0010701823026274E-32, 2.0881265265827577E-
33, 2.093543058006342E-34, 2.0180672939051796E-35, 2.0323857818493385E-36, 2.0323857818493385E-36, 2.0180672939051796E-
35, 2.093543058006342E-34, 2.0881265265827577E-33, 2.0010701823026274E-32, 1.8424515247958063E-31, 1.629889045514653E-
30, 1.3853140284356237E-29, 1.131270973846889E-28, 8.875917908092199E-28, 6.6909553275708E-27, 4.846086976278897E-
26, 3.3722712449973575E-25, 2.2546649574785994E-24, 1.4483374194607976E-23, 8.938933540581667E-23, 5.300659243097793E-
22, 3.0199676074488767E-21, 1.653114380789534E-20, 8.694242389997548E-20, 4.393279203075559E-19, 2.1329176397360208E-
18, 9.949186871131627E-18, 4.4589162059101576E-17, 1.9199912722931645E-16, 7.943235137708344E-16, 3.157357827645033E-
15, 1.2058086905785324E-14, 4.424469810126137E-14, 1.559812093881814E-13, 5.283376391150684E-13, 1.719408383151711E-
12, 5.376191739274649E-12, 1.6150972561350377E-11, 4.661769864954456E-11, 1.2927995780298244E-10, 3.4446079052452833E-
10, 8.818132230858036E-10, 2.1689105469611807E-9, 5.125484152742597E-9, 1.1637410936076988E-8, 2.5386689199257418E-
8, 5.320886348815409E-8, 1.0714949291069728E-7, 2.0731201751868198E-7, 3.8537812277721805E-7, 6.883001376504317E-
7, 1.1811277594509235E-6, 1.947350460421122E-6, 3.0847472040201255E-6, 4.694866653320701E-6, 6.865231156599492E-6,
9.645289972126572E-6, 1.3019779620499557E-5, 1.688574391158374E-5, 2.1040932566882033E-5, 2.519056980534633E-5, 2.897605242537388
5, 3.202349046115855E-5, 3.400371251960486E-5, 3.469063307838546E-5, 3.400371251960486E-5, 3.202349046115855E-5, 2.897605242537388
5, 2.519056980534633E-5, 2.1040932566882033E-5, 1.688574391158374E-5, 1.3019779620499557E-5, 9.645289972126572E-6,
6.865231156599492E-6, 4.694866653320701E-6, 3.0847472040201255E-6, 1.947350460421122E-6, 1.1811277594509235E-6, 6.883001376504317
7, 3.8537812277721805E-7, 2.0731201751868198E-7, 1.0714949291069728E-7, 5.320886348815409E-8, 2.5386689199257418E-8,
1.1637410936076988E-8, 5.125484152742597E-9, 2.1689105469611807E-9, 8.818132230858036E-10, 3.4446079052452833E-10,
1.2927995780298244E-10, 4.661769864954456E-11, 1.6150972561350377E-11, 5.376191739274649E-12, 1.719408383151711E-
12, 5.283376391150684E-13, 1.559812093881814E-13, 4.424469810126137E-14, 1.2058086905785324E-14, 3.157357827645033E-
15, 7.943235137708344E-16, 1.9199912722931645E-16, 4.4589162059101576E-17, 9.949186871131627E-18, 2.1329176397360208E-
18, 4.393279203075559E-19, 8.694242389997548E-20, 1.653114380789534E-20, 3.0199676074488767E-21, 5.300659243097793E-
22, 8.938933540581667E-23, 1.4483374194607976E-23, 2.2546649574785994E-24, 3.3722712449973575E-25, 4.846086976278897E-
26, 6.6909553275708E-27, 8.875917908092199E-28, 1.131270973846889E-28, 1.3853140284356234E-29, 1.629889045514617E-
30, 1.8424515247907424E-31, 2.0010701816070773E-32, 2.0881264347900316E-33, 2.0935314189906968E-34, 2.016649371527058E-
35, 1.8664206397070093E-36, 1.6596514214232952E-37, 1.4179223781214507E-38, 1.163901564568946E-39, 9.1792726108286E-
41, 6.955502397397228E-42, 5.063804869336075E-43, 3.5420414587198654E-44, 2.380447123787191E-45, 1.537062965437849E-
46, 9.535709212078012E-48, 5.6838493698423115E-49, 3.2550700694652257E-50, 1.791044367312114E-51, 9.46848656806728E-
53, 4.8093126520386236E-54, 2.3470031690332943E-55, 1.1004556926998E-56, 4.9574651225575716E-58, 2.1457296100570693E-
59, 8.923157154712205E-61, 3.5652526398549732E-62, 1.3686433737188218E-63, 5.0479902665468615E-65, 1.7888541765466197E-
66, 6.090593160277028E-68, 1.9923813249560384E-69, 6.262007089320793E-71, 1.8909622906949667E-72, 5.486310744003409E-
74, 1.5293474283435687E-75, 4.096001741817376E-77, 1.054004169485085E-78, 2.605869998411288E-80, 6.190010468903476E-
82, 1.4127269884987897E-83, 3.0977996121751494E-85, 6.526444075178308E-87, 1.3210770302575327E-88, 2.569258977964455E-
90, 4.800824662747621E-92, 8.61890368510078E-94, 1.4866763441763142E-95, 2.463820447951128E-97, 3.923104712650582E-
99, 6.001764713154106E-101, 8.82178059207161E-103, 1.2458385331735167E-104, 1.6904233657038514E-106, 2.203725188274832E-
108, 2.7602448285437066E-110, 3.321742742940293E-112, 3.8407196341387565E-114, 4.2666541120006704E-116, 4.553973396064257E-
118, 4.670052491322254E-120, 4.601307479203744E-122, 4.355810425059214E-124, 3.961730281836719E-126, 3.4620159215246154E-
128, 2.9067082102560503E-130, 2.3447796407608932E-132, 1.8173176833982343E-134, 1.3532807095685038E-136, 9.682179961963562E-
139, 6.655591075147299E-141, 4.39570290220884E-143, 2.7893194367033434E-145, 1.7005773695846732E-147, 9.961454564547626E-
150, 5.606313018606829E-152, 3.031517967745701E-154, 1.574965960266907E-156, 7.861590490859648E-159, 3.770317157145745E-
161, 1.737295007972417E-163, 7.691259770839384E-166, 3.2715202183119997E-168, 1.3369955388555208E-170, 5.249748866010984E-
173, 1.980502155423184E-175, 7.178609528942127E-178, 2.499962890205022E-180, 8.364789505081965E-183, 2.6890859930557118E-
185, 8.305821340825077E-188, 2.4648398142664035E-190, 7.027858949244228E-193, 1.9252431785542454E-195, 5.067297131365684E-
198, 1.2814314216335197E-200, 3.1134549632521633E-203, 7.268052029497823E-206, 1.630127928050357E-208, 3.5128011730023157E-
211, 7.273001396396924E-214, 1.4467783242945037E-216, 2.7651489909671062E-219, 5.07765630130979E-222, 8.958520370276534E-
225, 1.5185793537032843E-227, 2.473243732429713E-230, 3.870121183265132E-233, 5.818491868493537E-236, 8.404744585683771E-
239, 1.1664518846516838E-241, 1.555383039563573E-244, 1.9926733356940975E-247]
  arg0[] elements >= 0.0

return == 1.76272529158797E-5
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
return in arg0[]
==========
Models.getTCPSmoothLenProb(double, java.util.HashMap[], java.util.Vector[], double[][], double[], java.lang.String):::ENTER
this.lenProb_out[] == arg3[]
this.anomalyLengthThreshold_out[] == arg4[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null,
null, null,
null, null,
null, null,
null, null,
null, null,
null, null,
null, null,
null, null,
null, null,
null, null,
null, null]
this.TCP_lengthModel has only one value
this.TCP_lengthModel[] has only one value, of length 1024
this.UDP_lengthModel has only one value
this.TCP_lengthModel_in has only one value
this.UDP_lengthModel_in has only one value
this.TCP_lengthModel_out has only one value
this.UDP_lengthModel_out has only one value
this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.UDPNormalLength == null
this.normal_length_perc == 0.999
arg0 == 5.0
this.TCP_model.getClass() == arg1.getClass()
this.TCP_lengthModel.getClass() == arg2.getClass()
this.lenProb.getClass() == arg3.getClass()
==========
Models.getTCPSmoothLenProb(double, java.util.HashMap[], java.util.Vector[], double[][], double[], java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)

this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.lenProb_out[] == orig(arg3[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.anomalyLengthThreshold_out[] == orig(arg4[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.UDPNormalLength == null
this.normal_length_perc == 0.999
arg4[] elements one of 0.0, 1.76272529158797E-5
arg4[] one of [0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(arg1.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(arg2.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.lenProb.getClass() == orig(arg3.getClass())
arg5.toString == orig(arg5.toString)
==========
Models.postProcess():::ENTER
this.TCP_lengthModel == this.UDP_lengthModel
this.TCP_lengthModel == this.TCP_lengthModel_in
this.TCP_lengthModel == this.UDP_lengthModel_in
this.TCP_lengthModel == this.TCP_lengthModel_out
this.TCP_lengthModel == this.UDP_lengthModel_out
this.TCP_lengthModel == this.TCP_lengthModel_outsrc
this.TCP_lengthModel == this.UDP_lengthModel_outsrc
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null,
null, null,
null, java.util.HashMap, null, null, null, null, null, null,
null, null,
null, null,
null, null,
null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.anomalyLengthThreshold[] elements == 0.0
this.UDPNormalLength == null
this.normal_length_perc == 0.0
this.anomalyLengthThreshold[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold[]
==========
Models.postProcess():::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.lenProb[] == orig(this.lenProb[])
this.lenProb[] == orig(this.lenProb_out[])
this.lenProb[] == orig(this.lenProb_in[])
this.lenProb[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_out == orig(this.lenProb_out)

this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.TCP_lengthModel_out has only one value
this.UDP_lengthModel_out has only one value
this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.lenProb[] contains only nulls and has only one value, of length 1024
this.lenProb[] elements == null
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

java.lang.Integer, java.lang.Integer, java.lang.Integer,

this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg0[] == orig(arg0[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

java.lang.Integer, java.lang.Integer, java.lang.Integer,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
arg0 == 0.2
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == arg1.getClass()
this.TCP_lengthModel.getClass() == arg2.getClass()
this.host_ip.toString > arg3.toString
this.anomalyLengthThreshold[] elements < this.normal_length_perc
this.anomalyLengthThreshold[] elements < arg0

==========

Models.reduceModel_onedirection(double, java.util.HashMap[], java.util.Vector[], java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])

this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
arg0 has only one value
arg0.toString == ”text-model-p3.dat”
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
==========
Models.saveTextModel(java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])

this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
==========
Models.saveText_anomalyLength(java.io.PrintWriter, double[], java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])

this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg1[] == orig(arg1[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
arg0 has only one value
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.lenProb.getClass() == arg1.getClass()
==========
Models.saveText_lenprob(java.io.PrintWriter, double[][], java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])

this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg1[] == orig(arg1[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
arg0 has only one value
arg4.toString one of " dst ", " src "
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == arg1.getClass()
this.TCP_lengthModel.getClass() == arg2.getClass()
this.host_ip.toString > arg3.toString
this.host_ip.toString > arg4.toString
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg3.toString != arg4.toString

==========

Models.saveText_oneDirection(java.io.PrintWriter, java.util.HashMap[], java.util.Vector[], java.lang.String, java.lang.String):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])
this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])

this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.UDPNormalLength has only one value
this.UDPNormalLength_out has only one value
this.UDPNormalLength_in has only one value
this.UDPNormalLength_outsrc has only one value
this.normal_length_perc == 0.999
arg4.toString one of " dst ", " src "
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_model.getClass() == orig(arg1.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(arg2.getClass())
this.host_ip.toString > arg3.toString
this.host_ip.toString > arg4.toString
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg3.toString != arg4.toString
arg3.toString == orig(arg3.toString)
arg4.toString == orig(arg4.toString)
==========
Models.saveText_onemodel(java.io.PrintWriter, OneCentroid):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.min_sample == arg1.resettimes
arg1.num == arg1.count
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null,
null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, java.util.HashMap, null, null, null, null, null, null,
null, null,
null, null,
null, null,

null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.trunc_length < arg1.num
this.trunc_length < arg1.start
this.trunc_length < arg1.end
this.min_sample < arg1.num
this.min_sample < arg1.start
this.min_sample < arg1.end
this.max_sample > arg1.num
this.max_sample != arg1.start
this.max_sample != arg1.end
arg1.x[] elements >= this.sample_rate
this.sample_rate in arg1.x[]
arg1.xsqr[] elements >= this.sample_rate
this.sample_rate in arg1.xsqr[]
arg1.avg[] elements >= this.sample_rate
this.sample_rate in arg1.avg[]
arg1.std[] elements > this.sample_rate
this.sample_rate < arg1.centroidAnomalyThreshold
this.host_ip.toString > arg1.filetype.toString
this.stablesize <= arg1.num
this.stablesize < arg1.start
this.stablesize < arg1.end
arg1.std[] elements >= this.smooth_factor
this.smooth_factor in arg1.std[]
this.smooth_factor < arg1.centroidAnomalyThreshold
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg1.num != arg1.start
arg1.num != arg1.end
arg1.num != arg1.maxSample
arg1.x[] >= arg1.xsqr[] (elementwise)
arg1.x[] >= arg1.avg[] (elementwise)
arg1.x[] elements < arg1.centroidAnomalyThreshold
arg1.xsqr[] elements < arg1.centroidAnomalyThreshold
arg1.avg[] elements < arg1.samplePerc
arg1.avg[] elements < arg1.centroidAnomalyThreshold
arg1.std[] elements < arg1.samplePerc
arg1.std[] elements < arg1.centroidAnomalyThreshold
arg1.start <= arg1.end
arg1.start > arg1.minSample
arg1.start > arg1.maxSample
arg1.end > arg1.minSample
arg1.end > arg1.maxSample
arg1.centroidAnomalyThreshold
arg1.samplePerc < arg1.centroidAnomalyThreshold
==========
Models.saveText_onemodel(java.io.PrintWriter, OneCentroid):::EXIT
this.TCP_model == orig(this.TCP_model)
this.TCP_model[] == orig(this.TCP_model[])
this.UDP_model == orig(this.UDP_model)
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == orig(this.UDP_model[])
this.UDP_model[] == orig(this.TCP_model_in[])
this.UDP_model[] == orig(this.UDP_model_in[])
this.UDP_model[] == orig(this.TCP_model_out[])
this.UDP_model[] == orig(this.UDP_model_out[])
this.UDP_model[] == orig(this.TCP_model_outsrc[])
this.UDP_model[] == orig(this.UDP_model_outsrc[])
this.UDP_model[] == orig(this.UDPNormalLength[])
this.UDP_model[] == orig(this.UDPNormalLength_out[])
this.UDP_model[] == orig(this.UDPNormalLength_in[])

this.UDP_model[] == orig(this.UDPNormalLength_outsrc[])
this.TCP_model_in == orig(this.TCP_model_in)
this.UDP_model_in == orig(this.UDP_model_in)
this.TCP_model_out == orig(this.TCP_model_out)
this.UDP_model_out == orig(this.UDP_model_out)
this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel[] == orig(this.TCP_lengthModel[])
this.UDP_lengthModel == orig(this.UDP_lengthModel)
this.UDP_lengthModel[] == orig(this.UDP_lengthModel[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_in[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_out[])
this.UDP_lengthModel[] == orig(this.TCP_lengthModel_outsrc[])
this.UDP_lengthModel[] == orig(this.UDP_lengthModel_outsrc[])
this.TCP_lengthModel_in == orig(this.TCP_lengthModel_in)
this.UDP_lengthModel_in == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel_out == orig(this.TCP_lengthModel_out)
this.UDP_lengthModel_out == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel_outsrc == orig(this.TCP_lengthModel_outsrc)
this.UDP_lengthModel_outsrc == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == arg1.resettimes
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.min_sample == orig(arg1.resettimes)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == orig(this.lenProb[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.lenProb_out == orig(this.lenProb_out)
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg1.num == arg1.count
arg1.num == orig(arg1.num)
arg1.num == orig(arg1.count)
arg1.x == orig(arg1.x)
arg1.x[] == orig(arg1.x[])
arg1.xsqr == orig(arg1.xsqr)

arg1.xsqr[] == orig(arg1.xsqr[])
arg1.avg == orig(arg1.avg)
arg1.avg[] == orig(arg1.avg[])
arg1.std == orig(arg1.std)
arg1.std[] == orig(arg1.std[])
arg1.start == orig(arg1.start)
arg1.end == orig(arg1.end)
arg1.minSample == orig(arg1.minSample)
arg1.maxSample == orig(arg1.maxSample)
arg1.keepSample == orig(arg1.keepSample)
arg1.samplePerc == orig(arg1.samplePerc)
arg1.samples == orig(arg1.samples)
arg1.samples[] == orig(arg1.samples[])
arg1.centroidAnomalyThreshold == orig(arg1.centroidAnomalyThreshold)
arg1.merged == orig(arg1.merged)
arg1.filetype == orig(arg1.filetype)
arg1.sampleDistance == orig(arg1.sampleDistance)
arg1.sampleDistance[] == orig(arg1.sampleDistance[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.TCP_lengthModel has only one value
this.TCP_lengthModel != null
this.TCP_lengthModel[] has only one value, of length 1024
this.UDP_lengthModel has only one value
this.UDP_lengthModel != null
this.TCP_lengthModel_in has only one value
this.TCP_lengthModel_in != null
this.UDP_lengthModel_in has only one value
this.UDP_lengthModel_in != null
this.TCP_lengthModel_out has only one value
this.TCP_lengthModel_out != null
this.UDP_lengthModel_out has only one value
this.UDP_lengthModel_out != null
this.TCP_lengthModel_outsrc has only one value
this.TCP_lengthModel_outsrc != null
this.UDP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc != null
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

arg1.num >= 1
arg1.x != null
arg1.x[] elements >= 0.0
arg1.xsqr != null
arg1.xsqr[] elements >= 0.0
arg1.avg != null
arg1.avg[] elements >= 0.0
arg1.std != null
arg1.std[] elements >= 0.0010
arg1.minSample == 5
arg1.maxSample == 20
arg1.samplePerc == 0.5
arg1.samples != null
arg1.samples[] elements != null
arg1.samples[].getClass() elements == short[].class
arg1.filetype has only one value
arg1.filetype != null
arg1.filetype.toString == ""
arg1.sampleDistance != null
arg1.sampleDistance[] elements != null
arg1.sampleDistance[].getClass() elements == java.lang.Double.class
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.trunc_length < arg1.num
this.trunc_length < arg1.start
this.trunc_length < arg1.end
this.min_sample < arg1.num
this.min_sample < arg1.start
this.min_sample < arg1.end
this.max_sample > arg1.num
this.max_sample != arg1.start
this.max_sample != arg1.end
arg1.x[] elements >= this.sample_rate
this.sample_rate in arg1.x[]
arg1.xsqr[] elements >= this.sample_rate
this.sample_rate in arg1.xsqr[]
arg1.avg[] elements >= this.sample_rate
this.sample_rate in arg1.avg[]
arg1.std[] elements > this.sample_rate
this.sample_rate < arg1.centroidAnomalyThreshold
this.host_ip.toString > arg1.filetype.toString
this.host_ip.toString == orig(this.host_ip.toString)

this.stablesize <= arg1.num
this.stablesize < arg1.start
this.stablesize < arg1.end
arg1.std[] elements >= this.smooth_factor
this.smooth_factor in arg1.std[]
this.smooth_factor < arg1.centroidAnomalyThreshold
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements < this.normal_length_perc
arg1.num != arg1.start
arg1.num != arg1.end
arg1.num != arg1.maxSample
arg1.x[] >= arg1.xsqr[] (elementwise)
arg1.x[] >= arg1.avg[] (elementwise)
arg1.x[] elements < arg1.centroidAnomalyThreshold
arg1.xsqr[] elements < arg1.centroidAnomalyThreshold
arg1.avg[] elements < arg1.samplePerc
arg1.avg[] elements < arg1.centroidAnomalyThreshold
arg1.std[] elements < arg1.samplePerc
arg1.std[] elements < arg1.centroidAnomalyThreshold
arg1.start <= arg1.end
arg1.start > arg1.minSample
arg1.start > arg1.maxSample
arg1.end > arg1.minSample
arg1.end > arg1.maxSample
arg1.centroidAnomalyThreshold
arg1.samplePerc < arg1.centroidAnomalyThreshold
arg1.filetype.toString == orig(arg1.filetype.toString)
==========
Models.saveText_udplength(java.io.PrintWriter, java.util.HashMap[], java.lang.String):::ENTER
this.UDP_model[] == this.UDPNormalLength[]
this.UDP_model[] == this.UDPNormalLength_out[]
this.UDP_model[] == this.UDPNormalLength_in[]
this.UDP_model[] == this.UDPNormalLength_outsrc[]
this.UDP_model[] == arg1[]
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0,

this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength[] == arg1[]
this.UDPNormalLength_out == orig(this.UDPNormalLength_out)
this.UDPNormalLength_out[] == arg1[]
this.UDPNormalLength_in == orig(this.UDPNormalLength_in)
this.UDPNormalLength_in[] == arg1[]
this.UDPNormalLength_outsrc == orig(this.UDPNormalLength_outsrc)
this.UDPNormalLength_outsrc[] == arg1[]
this.normal_length_perc == orig(this.normal_length_perc)
arg1[] == orig(this.UDP_model[])
arg1[] == orig(this.TCP_model_in[])
arg1[] == orig(this.UDP_model_in[])
arg1[] == orig(this.TCP_model_out[])
arg1[] == orig(this.UDP_model_out[])
arg1[] == orig(this.TCP_model_outsrc[])
arg1[] == orig(this.UDP_model_outsrc[])
arg1[] == orig(this.UDPNormalLength[])
arg1[] == orig(this.UDPNormalLength_out[])
arg1[] == orig(this.UDPNormalLength_in[])
arg1[] == orig(this.UDPNormalLength_outsrc[])
arg1[] == orig(arg1[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.normal_length_perc == 0.999
arg1[] contains only nulls and has only one value, of length 1024
arg1[] elements == null
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_out.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_in.getClass())
this.TCP_model.getClass() == orig(this.UDPNormalLength_outsrc.getClass())
this.TCP_model.getClass() == orig(arg1.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_in.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_out.getClass())
this.TCP_lengthModel.getClass() == orig(this.TCP_lengthModel_outsrc.getClass())
this.TCP_lengthModel.getClass() == orig(this.UDP_lengthModel_outsrc.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
arg2.toString == orig(arg2.toString)
==========
Models.trainModel(java.lang.String, java.lang.String, int, byte[]):::ENTER
this.TCP_lengthModel == this.UDP_lengthModel
this.TCP_lengthModel == this.TCP_lengthModel_in
this.TCP_lengthModel == this.UDP_lengthModel_in
this.TCP_lengthModel == this.TCP_lengthModel_out
this.TCP_lengthModel == this.UDP_lengthModel_out
this.TCP_lengthModel == this.TCP_lengthModel_outsrc
this.TCP_lengthModel == this.UDP_lengthModel_outsrc
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_lengthModel == null
this.lenProb[] contains only nulls and has only one value, of length 1024
this.lenProb[] elements == null
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.TCP_model_outsrc == orig(this.TCP_model_outsrc)
this.UDP_model_outsrc == orig(this.UDP_model_outsrc)
this.TCP_lengthModel == this.UDP_lengthModel
this.TCP_lengthModel == this.TCP_lengthModel_in
this.TCP_lengthModel == this.UDP_lengthModel_in
this.TCP_lengthModel == this.TCP_lengthModel_out
this.TCP_lengthModel == this.UDP_lengthModel_out
this.TCP_lengthModel == this.TCP_lengthModel_outsrc
this.TCP_lengthModel == this.UDP_lengthModel_outsrc
this.TCP_lengthModel == orig(this.TCP_lengthModel)
this.TCP_lengthModel == orig(this.UDP_lengthModel)
this.TCP_lengthModel == orig(this.TCP_lengthModel_in)
this.TCP_lengthModel == orig(this.UDP_lengthModel_in)
this.TCP_lengthModel == orig(this.TCP_lengthModel_out)
this.TCP_lengthModel == orig(this.UDP_lengthModel_out)
this.TCP_lengthModel == orig(this.TCP_lengthModel_outsrc)
this.TCP_lengthModel == orig(this.UDP_lengthModel_outsrc)
this.trunc_length == orig(this.trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb == orig(this.lenProb)
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.lenProb[] == orig(this.lenProb[])
this.lenProb[] == orig(this.lenProb_out[])
this.lenProb[] == orig(this.lenProb_in[])
this.lenProb[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold == orig(this.anomalyLengthThreshold)
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold[] == orig(this.anomalyLengthThreshold_outsrc[])
this.lenProb_out == orig(this.lenProb_out)
this.anomalyLengthThreshold_out == orig(this.anomalyLengthThreshold_out)
this.lenProb_in == orig(this.lenProb_in)
this.anomalyLengthThreshold_in == orig(this.anomalyLengthThreshold_in)
this.lenProb_outsrc == orig(this.lenProb_outsrc)
this.anomalyLengthThreshold_outsrc == orig(this.anomalyLengthThreshold_outsrc)
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.UDPNormalLength == orig(this.UDPNormalLength)
this.UDPNormalLength == orig(this.UDPNormalLength_out)
this.UDPNormalLength == orig(this.UDPNormalLength_in)
this.UDPNormalLength == orig(this.UDPNormalLength_outsrc)
this.normal_length_perc == orig(this.normal_length_perc)
arg3[] == orig(arg3[])
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

null, null, null, null, null, null, null, null, null, null, null, null, null, java.util.HashMap, null, null, null, null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.anomalyLengthThreshold[] elements == 0.0
this.UDPNormalLength == null
this.normal_length_perc == 0.0
arg0.toString == "TCP_model"
arg1.toString == "tcp"
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
arg3[] elements > this.trunc_length
arg3[] elements > this.min_sample
arg3[] elements < this.max_sample
this.host_ip.toString < arg0.toString
this.host_ip.toString < arg1.toString
this.host_ip.toString == orig(this.host_ip.toString)
arg3[] elements > this.stablesize
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
this.anomalyLengthThreshold[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold[]
arg0.toString < arg1.toString
arg0.toString == orig(arg0.toString)
arg1.toString == orig(arg1.toString)
orig(arg2) in arg3[]

==========

OneCentroid:::OBJECT
this.num == this.count
this.num >= 1
this.x != null
this.x[] elements >= 0.0
this.xsqr != null
this.xsqr[] elements >= 0.0
this.avg != null
this.avg[] elements >= 0.0
this.std != null
this.std[] elements >= 0.0
this.start >= 0
this.end >= 0
this.resettimes == 0
this.minSample == 5

this.maxSample == 20
this.samplePerc == 0.5
this.samples != null
this.samples[] elements != null
this.samples[].getClass() elements == short[].class
this.filetype has only one value
this.filetype != null
this.filetype.toString == ""
this.sampleDistance != null
this.sampleDistance[] elements != null
this.sampleDistance[].getClass() elements == java.lang.Double.class
this.num != this.start
this.num != this.end
this.num > this.resettimes
this.x[] >= this.xsqr[] (elementwise)
this.x[] >= this.avg[] (elementwise)
this.x[] elements < this.centroidAnomalyThreshold
this.xsqr[] elements < this.centroidAnomalyThreshold
this.avg[] elements < this.samplePerc
this.avg[] elements < this.centroidAnomalyThreshold
this.std[] elements < this.samplePerc
this.std[] elements < this.centroidAnomalyThreshold
this.start <= this.end
this.start >= this.resettimes
this.start != this.minSample
this.start != this.maxSample
this.end >= this.resettimes
this.end != this.minSample
this.end != this.maxSample
this.centroidAnomalyThreshold
this.samplePerc < this.centroidAnomalyThreshold
==========
OneCentroid.MahaDis(float[], float[], float[]):::ENTER
arg0 != null
arg0[] elements >= 0.0
arg1 != null
arg1[] elements >= 0.0
arg2 != null
arg2[] elements >= 0.0010
==========
OneCentroid.MahaDis(float[], float[], float[]):::EXIT
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
arg0[] elements >= 0.0
arg1[] elements >= 0.0
arg2[] elements >= 0.0010
return >= 0.0
==========
OneCentroid.OneCentroid(float[], short[]):::ENTER
arg0 != null
arg0[] elements >= 0.0
arg1 != null
arg1[] elements >= 0
==========
OneCentroid.OneCentroid(float[], short[]):::EXIT
this.x[] == arg0[]
this.avg[] == arg0[]
this.start == this.end
this.start == this.resettimes
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
this.num == 1

this.std[] == [0.0, 0.0, 0.0,

OneCentroid.add(float[], short[]):::EXIT
this.x == orig(this.x)
this.xsqr == orig(this.xsqr)
this.avg == orig(this.avg)
this.std == orig(this.std)
this.std[] == orig(this.std[])
this.start == this.end
this.start == orig(this.start)
this.start == orig(this.end)
this.resettimes == orig(this.resettimes)
this.minSample == orig(this.minSample)
this.maxSample == orig(this.maxSample)
this.samplePerc == orig(this.samplePerc)
this.samples == orig(this.samples)
this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.merged == orig(this.merged)
this.filetype == orig(this.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == orig(this.sampleDistance[])
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
this.num >= 2
this.std[] == [0.0, 0.0, 0.0,

orig(this.x[]) elements < this.centroidAnomalyThreshold
orig(this.xsqr[]) elements < this.centroidAnomalyThreshold
orig(this.avg[]) elements < this.centroidAnomalyThreshold
this.filetype.toString == orig(this.filetype.toString)
===
OneCentroid.addSample(short[]):::ENTER
this.start == this.end
this.std[] == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.std[] elements == 0.0
this.keepSample == true
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0 != null
arg0[] elements >= 0
this.num < this.maxSample
this.x[] >= this.std[] (elementwise)
this.xsqr[] <= this.avg[] (elementwise)
this.xsqr[] >= this.std[] (elementwise)
this.xsqr[] elements < this.samplePerc
this.avg[] >= this.std[] (elementwise)
arg0[] elements >= this.resettimes
this.resettimes in arg0[]
===
OneCentroid.addSample(short[]):::EXIT441
===
OneCentroid.addSample(short[]):::EXIT
this.num == orig(this.num)
this.num == orig(this.count)
this.x == orig(this.x)
this.x[] == orig(this.x[])
this.xsqr == orig(this.xsqr)
this.xsqr[] == orig(this.xsqr[])
this.avg == orig(this.avg)
this.avg[] == orig(this.avg[])
this.std == orig(this.std)
this.std[] == orig(this.std[])
this.start == this.end
this.start == orig(this.start)
this.start == orig(this.end)
this.resettimes == orig(this.resettimes)
this.minSample == orig(this.minSample)
this.maxSample == orig(this.maxSample)
this.samplePerc == orig(this.samplePerc)
this.samples == orig(this.samples)
this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.merged == orig(this.merged)
this.filetype == orig(this.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == orig(this.sampleDistance[])
arg0[] == orig(arg0[])
this.std[] == [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.std[] elements == 0.0
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0[] elements >= 0
this.num < this.maxSample
this.x[] >= this.std[] (elementwise)
this.xsqr[] <= this.avg[] (elementwise)
this.xsqr[] >= this.std[] (elementwise)
this.xsqr[] elements < this.samplePerc
this.avg[] >= this.std[] (elementwise)
arg0[] elements >= this.resettimes
this.resettimes in arg0[]
this.filetype.toString == orig(this.filetype.toString)
==========
OneCentroid.addSampleDistance(double):::ENTER
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
arg0 >= 0.0
this.num != this.maxSample
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
==========
OneCentroid.addSampleDistance(double):::EXIT
this.num == orig(this.num)
this.num == orig(this.count)
this.x == orig(this.x)
this.x[] == orig(this.x[])
this.xsqr == orig(this.xsqr)
this.xsqr[] == orig(this.xsqr[])
this.avg == orig(this.avg)
this.avg[] == orig(this.avg[])
this.std == orig(this.std)
this.std[] == orig(this.std[])
this.start == orig(this.start)
this.end == orig(this.end)
this.resettimes == orig(this.resettimes)
this.minSample == orig(this.minSample)
this.maxSample == orig(this.maxSample)
this.keepSample == orig(this.keepSample)
this.samplePerc == orig(this.samplePerc)
this.samples == orig(this.samples)
this.samples[] == orig(this.samples[])
this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.merged == orig(this.merged)
this.filetype == orig(this.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.num != this.maxSample
this.start > this.resettimes
this.start > this.minSample

this.start > this.maxSample
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
this.filetype.toString == orig(this.filetype.toString)
===
OneCentroid.compareTo(java.lang.Object):::ENTER
this.start == this.end
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0 != null
arg0.getClass() == OneCentroid.class
this.num < this.start
this.num != this.maxSample
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
===
OneCentroid.compareTo(java.lang.Object):::EXIT406
return == 1
this.num < this.maxSample
this.xsqr[] elements < this.samplePerc
===
OneCentroid.compareTo(java.lang.Object):::EXIT408
this.num >= 2
return == -1
this.num > return
===
OneCentroid.compareTo(java.lang.Object):::EXIT409
this.resettimes == return
return == orig(this.resettimes)
this.keepSample == true
return == 0
this.num < this.minSample
this.num < this.maxSample
this.num > return
this.xsqr[] <= this.avg[] (elementwise)
this.xsqr[] elements < this.samplePerc
===
OneCentroid.compareTo(java.lang.Object):::EXIT
this.num == orig(this.num)
this.num == orig(this.count)
this.x == orig(this.x)
this.x[] == orig(this.x[])
this.xsqr == orig(this.xsqr)
this.xsqr[] == orig(this.xsqr[])
this.avg == orig(this.avg)
this.avg[] == orig(this.avg[])
this.std == orig(this.std)
this.std[] == orig(this.std[])
this.start == this.end
this.start == orig(this.start)
this.start == orig(this.end)
this.resettimes == orig(this.resettimes)
this.minSample == orig(this.minSample)
this.maxSample == orig(this.maxSample)
this.keepSample == orig(this.keepSample)
this.samplePerc == orig(this.samplePerc)
this.samples == orig(this.samples)
this.samples[] == orig(this.samples[])

this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.merged == orig(this.merged)
this.filetype == orig(this.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == orig(this.sampleDistance[])
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
return one of -1, 0, 1
this.num < this.start
this.num != this.maxSample
this.num >= return
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.start > return
this.minSample > return
this.maxSample > return
this.filetype.toString == orig(this.filetype.toString)
==========
OneCentroid.compute(float):::ENTER
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0 == 0.0010
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
==========
OneCentroid.compute(float):::EXIT
this.num == orig(this.num)
this.num == orig(this.count)
this.x == orig(this.x)
this.x[] == orig(this.x[])
this.xsqr == orig(this.xsqr)
this.xsqr[] == orig(this.xsqr[])
this.avg == orig(this.avg)
this.std == orig(this.std)
this.start == orig(this.start)
this.end == orig(this.end)
this.resettimes == orig(this.resettimes)
this.minSample == orig(this.minSample)
this.maxSample == orig(this.maxSample)
this.keepSample == orig(this.keepSample)
this.samplePerc == orig(this.samplePerc)
this.samples == orig(this.samples)
this.samples[] == orig(this.samples[])
this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.merged == orig(this.merged)
this.filetype == orig(this.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == orig(this.sampleDistance[])
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []

this.x[] >= orig(this.avg[]) (elementwise)
this.avg[] == orig(this.avg[]) (elementwise)
orig(arg0) in this.std[]
this.std[] elements >= orig(arg0)
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
orig(this.avg[]) elements < this.samplePerc
orig(this.std[]) elements < this.samplePerc
orig(this.avg[]) elements < this.centroidAnomalyThreshold
orig(this.std[]) elements < this.centroidAnomalyThreshold
this.filetype.toString == orig(this.filetype.toString)
==========
OneCentroid.computeThreshold():::ENTER
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.num != this.maxSample
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
==========
OneCentroid.computeThreshold():::EXIT
this.num == orig(this.num)
this.num == orig(this.count)
this.x == orig(this.x)
this.x[] == orig(this.x[])
this.xsqr == orig(this.xsqr)
this.xsqr[] == orig(this.xsqr[])
this.avg == orig(this.avg)
this.avg[] == orig(this.avg[])
this.std == orig(this.std)
this.std[] == orig(this.std[])
this.start == orig(this.start)
this.end == orig(this.end)
this.resettimes == orig(this.resettimes)
this.minSample == orig(this.minSample)
this.maxSample == orig(this.maxSample)
this.keepSample == orig(this.keepSample)
this.samplePerc == orig(this.samplePerc)
this.samples == orig(this.samples)
this.samples[] == orig(this.samples[])
this.merged == orig(this.merged)
this.filetype == orig(this.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == orig(this.sampleDistance[])
this.std[] elements >= 0.0010
this.num != this.maxSample
this.x[] elements < orig(this.centroidAnomalyThreshold)
this.xsqr[] elements < orig(this.centroidAnomalyThreshold)
this.avg[] elements < orig(this.centroidAnomalyThreshold)
this.std[] elements < orig(this.centroidAnomalyThreshold)
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample

this.filetype.toString == orig(this.filetype.toString)
==========
OneCentroid.indexSort(float[]):::ENTER
arg0 != null
arg0[] elements >= 0.0
==========
OneCentroid.indexSort(float[]):::EXIT
arg0[] == orig(arg0[])
arg0[] elements >= 0.0
return != null
return[] elements >= 0
==========
OneCentroid.merge(OneCentroid):::ENTER
this.resettimes == arg0.resettimes
this.minSample == arg0.minSample
this.maxSample == arg0.maxSample
this.samplePerc == arg0.samplePerc
this.centroidAnomalyThreshold == arg0.centroidAnomalyThreshold
this.merged == arg0.merged
this.filetype == arg0.filetype
this.sampleDistance[] == arg0.sampleDistance[]
arg0.num == arg0.count
arg0.start == arg0.end
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0 != null
arg0.num >= 1
arg0.x != null
arg0.x[] elements >= 0.0
arg0.xsqr != null
arg0.xsqr[] elements >= 0.0
arg0.avg != null
arg0.avg[] elements >= 0.0
arg0.std != null
arg0.std[] elements >= 0.0010
arg0.samples != null
arg0.samples[] elements != null
arg0.samples[].getClass() elements == short[].class
arg0.sampleDistance != null
this.num < this.start
this.num < this.end
this.num < arg0.start
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.start > arg0.num
this.start < arg0.start
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
this.end > arg0.num
this.end < arg0.start
this.resettimes < arg0.num
this.resettimes < arg0.start
this.minSample < arg0.start
this.maxSample != arg0.num
this.maxSample < arg0.start
arg0.avg[] elements < this.samplePerc
arg0.std[] elements < this.samplePerc
arg0.x[] elements < this.centroidAnomalyThreshold

arg0.xsqr[] elements < this.centroidAnomalyThreshold
arg0.avg[] elements < this.centroidAnomalyThreshold
arg0.std[] elements < this.centroidAnomalyThreshold
this.filetype.toString == arg0.filetype.toString
arg0.num < arg0.start
arg0.x[] >= arg0.xsqr[] (elementwise)
arg0.x[] >= arg0.avg[] (elementwise)
==========
OneCentroid.merge(OneCentroid):::EXIT
this.x == orig(this.x)
this.xsqr == orig(this.xsqr)
this.avg == orig(this.avg)
this.std == orig(this.std)
this.std[] == orig(this.std[])
this.start == orig(this.start)
this.end == arg0.start
this.end == arg0.end
this.end == orig(arg0.start)
this.end == orig(arg0.end)
this.resettimes == arg0.resettimes
this.resettimes == orig(this.resettimes)
this.resettimes == orig(arg0.resettimes)
this.minSample == arg0.minSample
this.minSample == orig(this.minSample)
this.minSample == orig(arg0.minSample)
this.maxSample == arg0.maxSample
this.maxSample == orig(this.maxSample)
this.maxSample == orig(arg0.maxSample)
this.samplePerc == arg0.samplePerc
this.samplePerc == orig(this.samplePerc)
this.samplePerc == orig(arg0.samplePerc)
this.samples == orig(this.samples)
this.centroidAnomalyThreshold == arg0.centroidAnomalyThreshold
this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.centroidAnomalyThreshold == orig(arg0.centroidAnomalyThreshold)
this.merged == arg0.merged
this.merged == orig(this.merged)
this.merged == orig(arg0.merged)
this.filetype == arg0.filetype
this.filetype == orig(this.filetype)
this.filetype == orig(arg0.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == arg0.sampleDistance[]
this.sampleDistance[] == orig(this.sampleDistance[])
this.sampleDistance[] == orig(arg0.sampleDistance[])
arg0.num == arg0.count
arg0.num == orig(arg0.num)
arg0.num == orig(arg0.count)
arg0.x == orig(arg0.x)
arg0.x[] == orig(arg0.x[])
arg0.xsqr == orig(arg0.xsqr)
arg0.xsqr[] == orig(arg0.xsqr[])
arg0.avg == orig(arg0.avg)
arg0.avg[] == orig(arg0.avg[])
arg0.std == orig(arg0.std)
arg0.std[] == orig(arg0.std[])
arg0.keepSample == orig(arg0.keepSample)
arg0.samples == orig(arg0.samples)
arg0.samples[] == orig(arg0.samples[])
arg0.sampleDistance == orig(arg0.sampleDistance)
this.num >= 2
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0

this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0.num >= 1
arg0.x != null
arg0.x[] elements >= 0.0
arg0.xsqr != null
arg0.xsqr[] elements >= 0.0
arg0.avg != null
arg0.avg[] elements >= 0.0
arg0.std != null
arg0.std[] elements >= 0.0010
arg0.samples != null
arg0.samples[] elements != null
arg0.samples[].getClass() elements == short[].class
arg0.sampleDistance != null
this.num > arg0.num
this.num > orig(this.num)
this.num != orig(this.end)
this.x[] >= arg0.x[] (elementwise)
this.x[] >= arg0.xsqr[] (elementwise)
this.x[] >= arg0.avg[] (elementwise)
this.x[] >= orig(this.x[]) (elementwise)
this.x[] >= orig(this.xsqr[]) (elementwise)
this.x[] >= orig(this.avg[]) (elementwise)
this.xsqr[] >= arg0.xsqr[] (elementwise)
this.xsqr[] >= orig(this.xsqr[]) (elementwise)
this.start < this.end
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.start > arg0.num
this.start > orig(this.num)
this.start <= orig(this.end)
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
this.end > arg0.num
this.end > orig(this.num)
this.end > orig(this.end)
this.resettimes < arg0.num
this.resettimes < orig(this.num)
this.resettimes < orig(this.end)
this.minSample < orig(this.end)
this.maxSample != arg0.num
this.maxSample < orig(this.end)
arg0.avg[] elements < this.samplePerc
arg0.std[] elements < this.samplePerc
orig(this.avg[]) elements < this.samplePerc
arg0.x[] elements < this.centroidAnomalyThreshold
arg0.xsqr[] elements < this.centroidAnomalyThreshold
arg0.avg[] elements < this.centroidAnomalyThreshold
arg0.std[] elements < this.centroidAnomalyThreshold
orig(this.x[]) elements < this.centroidAnomalyThreshold
orig(this.xsqr[]) elements < this.centroidAnomalyThreshold
orig(this.avg[]) elements < this.centroidAnomalyThreshold
this.filetype.toString == arg0.filetype.toString
this.filetype.toString == orig(this.filetype.toString)
this.filetype.toString == orig(arg0.filetype.toString)
arg0.num < orig(this.end)
arg0.x[] >= arg0.xsqr[] (elementwise)
arg0.x[] >= arg0.avg[] (elementwise)
this.num - arg0.num - orig(this.num) == 0

```
===========================================================================
OneCentroid.merge_std(OneCentroid):::ENTER
this.resettimes == arg0.resettimes
this.minSample == arg0.minSample
this.maxSample == arg0.maxSample
this.samplePerc == arg0.samplePerc
this.centroidAnomalyThreshold == arg0.centroidAnomalyThreshold
this.merged == arg0.merged
this.filetype == arg0.filetype
this.sampleDistance[] == arg0.sampleDistance[]
arg0.num == arg0.count
arg0.start == arg0.end
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0 != null
arg0.num >= 1
arg0.x != null
arg0.x[] elements >= 0.0
arg0.xsqr != null
arg0.xsqr[] elements >= 0.0
arg0.avg != null
arg0.avg[] elements >= 0.0
arg0.std != null
arg0.std[] elements >= 0.0010
arg0.samples != null
arg0.samples[] elements != null
arg0.samples[].getClass() elements == short[].class
arg0.sampleDistance != null
this.num < this.start
this.num < this.end
this.num < arg0.start
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.start > arg0.num
this.start < arg0.start
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
this.end > arg0.num
this.end < arg0.start
this.resettimes < arg0.num
this.resettimes < arg0.start
this.minSample < arg0.start
this.maxSample != arg0.num
this.maxSample < arg0.start
arg0.avg[] elements < this.samplePerc
arg0.std[] elements < this.samplePerc
arg0.x[] elements < this.centroidAnomalyThreshold
arg0.xsqr[] elements < this.centroidAnomalyThreshold
arg0.avg[] elements < this.centroidAnomalyThreshold
arg0.std[] elements < this.centroidAnomalyThreshold
this.filetype.toString == arg0.filetype.toString
arg0.num < arg0.start
arg0.x[] >= arg0.xsqr[] (elementwise)
arg0.x[] >= arg0.avg[] (elementwise)
===========================================================================
OneCentroid.merge_std(OneCentroid):::EXIT
this.x == orig(this.x)
this.xsqr == orig(this.xsqr)
```

this.avg == orig(this.avg)
this.std == orig(this.std)
this.start == orig(this.start)
this.end == arg0.start
this.end == arg0.end
this.end == orig(arg0.start)
this.end == orig(arg0.end)
this.resettimes == arg0.resettimes
this.resettimes == orig(this.resettimes)
this.resettimes == orig(arg0.resettimes)
this.minSample == arg0.minSample
this.minSample == orig(this.minSample)
this.minSample == orig(arg0.minSample)
this.maxSample == arg0.maxSample
this.maxSample == orig(this.maxSample)
this.maxSample == orig(arg0.maxSample)
this.samplePerc == arg0.samplePerc
this.samplePerc == orig(this.samplePerc)
this.samplePerc == orig(arg0.samplePerc)
this.samples == orig(this.samples)
this.centroidAnomalyThreshold == arg0.centroidAnomalyThreshold
this.centroidAnomalyThreshold == orig(this.centroidAnomalyThreshold)
this.centroidAnomalyThreshold == orig(arg0.centroidAnomalyThreshold)
this.merged == arg0.merged
this.merged == orig(this.merged)
this.merged == orig(arg0.merged)
this.filetype == arg0.filetype
this.filetype == orig(this.filetype)
this.filetype == orig(arg0.filetype)
this.sampleDistance == orig(this.sampleDistance)
this.sampleDistance[] == arg0.sampleDistance[]
this.sampleDistance[] == orig(this.sampleDistance[])
this.sampleDistance[] == orig(arg0.sampleDistance[])
arg0.num == arg0.count
arg0.num == orig(arg0.num)
arg0.num == orig(arg0.count)
arg0.x == orig(arg0.x)
arg0.x[] == orig(arg0.x[])
arg0.xsqr == orig(arg0.xsqr)
arg0.xsqr[] == orig(arg0.xsqr[])
arg0.avg == orig(arg0.avg)
arg0.avg[] == orig(arg0.avg[])
arg0.std == orig(arg0.std)
arg0.std[] == orig(arg0.std[])
arg0.keepSample == orig(arg0.keepSample)
arg0.samples == orig(arg0.samples)
arg0.samples[] == orig(arg0.samples[])
arg0.sampleDistance == orig(arg0.sampleDistance)
this.num >= 2
this.std[] elements >= 0.0010
this.centroidAnomalyThreshold == 64.0
this.merged == false
this.sampleDistance[] == []
this.sampleDistance[].getClass() == []
arg0.num >= 1
arg0.x != null
arg0.x[] elements >= 0.0
arg0.xsqr != null
arg0.xsqr[] elements >= 0.0
arg0.avg != null
arg0.avg[] elements >= 0.0
arg0.std != null
arg0.std[] elements >= 0.0010

arg0.samples != null
arg0.samples[] elements != null
arg0.samples[].getClass() elements == short[].class
arg0.sampleDistance != null
this.num > arg0.num
this.num > orig(this.num)
this.num != orig(this.end)
this.x[] >= arg0.x[] (elementwise)
this.x[] >= arg0.xsqr[] (elementwise)
this.x[] >= arg0.avg[] (elementwise)
this.x[] >= orig(this.x[]) (elementwise)
this.x[] >= orig(this.xsqr[]) (elementwise)
this.x[] >= orig(this.avg[]) (elementwise)
this.xsqr[] >= arg0.xsqr[] (elementwise)
this.xsqr[] >= orig(this.xsqr[]) (elementwise)
this.start < this.end
this.start > this.resettimes
this.start > this.minSample
this.start > this.maxSample
this.start > arg0.num
this.start > orig(this.num)
this.start <= orig(this.end)
this.end > this.resettimes
this.end > this.minSample
this.end > this.maxSample
this.end > arg0.num
this.end > orig(this.num)
this.end > orig(this.end)
this.resettimes < arg0.num
this.resettimes < orig(this.num)
this.resettimes < orig(this.end)
this.minSample < orig(this.end)
this.maxSample != arg0.num
this.maxSample < orig(this.end)
arg0.avg[] elements < this.samplePerc
arg0.std[] elements < this.samplePerc
orig(this.avg[]) elements < this.samplePerc
orig(this.std[]) elements < this.samplePerc
arg0.x[] elements < this.centroidAnomalyThreshold
arg0.xsqr[] elements < this.centroidAnomalyThreshold
arg0.avg[] elements < this.centroidAnomalyThreshold
arg0.std[] elements < this.centroidAnomalyThreshold
orig(this.x[]) elements < this.centroidAnomalyThreshold
orig(this.xsqr[]) elements < this.centroidAnomalyThreshold
orig(this.avg[]) elements < this.centroidAnomalyThreshold
orig(this.std[]) elements < this.centroidAnomalyThreshold
this.filetype.toString == arg0.filetype.toString
this.filetype.toString == orig(this.filetype.toString)
this.filetype.toString == orig(arg0.filetype.toString)
arg0.num < orig(this.end)
arg0.x[] >= arg0.xsqr[] (elementwise)
arg0.x[] >= arg0.avg[] (elementwise)
this.num - arg0.num - orig(this.num) == 0
==========
OneModel:::OBJECT
this.start == this.end
this.b_packets[] == this.b_nums[]
this.centroids != null
this.centroids[] elements != null
this.centroids[].getClass() elements == OneCentroid.class
this.NMax one of 10, 20
this.start >= 0
this.min_sample == 0

this.max_sample one of 0, 1000
this.sample_rate == 0.0
this.totalPacketNum >= 0
this.b_packets != null
this.b_packets[] == []
this.b_packets[].getClass() == []
this.b_nums != null
this.batch == false
this.reduceN >= 0
this.NMax != this.start
this.NMax > this.min_sample
this.NMax != this.max_sample
this.NMax > this.reduceN
this.start >= this.min_sample
this.start >= this.totalPacketNum
this.start >= this.reduceN
this.min_sample <= this.max_sample
this.min_sample <= this.totalPacketNum
this.min_sample <= this.reduceN
this.max_sample >= this.totalPacketNum
this.max_sample >= this.reduceN
this.totalPacketNum >= this.reduceN
==
OneModel.OneModel():::EXIT
this.centroids[] == this.b_packets[]
this.centroids[] == this.b_nums[]
this.start == this.min_sample
this.start == this.max_sample
this.start == this.totalPacketNum
this.start == this.reduceN
this.centroids[] == []
this.centroids[].getClass() == []
this.NMax == 10
this.start == 0
==
OneModel.OneModel(int, int, int, float):::ENTER
arg0 == 20
arg1 == 0
arg2 == 1000
arg3 == 0.0
==
OneModel.OneModel(int, int, int, float):::EXIT
this.centroids[] == this.b_packets[]
this.centroids[] == this.b_nums[]
this.NMax == orig(arg0)
this.start == this.min_sample
this.start == this.totalPacketNum
this.start == this.reduceN
this.start == orig(arg1)
this.max_sample == orig(arg2)
this.sample_rate == orig(arg3)
this.centroids[] == []
this.centroids[].getClass() == []
this.NMax == 20
this.start == 0
this.max_sample == 1000
==
OneModel.addPacket(float[], short[]):::ENTER
this.min_sample == this.reduceN
this.NMax == 20
this.max_sample == 1000
arg0 != null
arg0[] elements >= 0.0

arg1 != null
arg1[] elements >= 0
this.NMax < this.start
this.start > this.min_sample
this.start != this.max_sample
this.start > this.totalPacketNum
arg1[] elements < this.start
arg1[] elements >= this.min_sample
this.min_sample in arg1[]
this.max_sample > this.totalPacketNum
arg1[] elements < this.max_sample
arg0[] elements >= this.sample_rate
this.sample_rate in arg0[]
==========
OneModel.addPacket(float[], short[]):::EXIT
this.centroids == orig(this.centroids)
this.NMax == orig(this.NMax)
this.start == orig(this.start)
this.start == orig(this.end)
this.min_sample == this.reduceN
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.reduceN)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.b_packets == orig(this.b_packets)
this.b_packets[] == orig(this.b_packets[])
this.b_packets[] == orig(this.b_nums[])
this.b_nums == orig(this.b_nums)
this.batch == orig(this.batch)
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
this.NMax == 20
this.max_sample == 1000
this.totalPacketNum >= 1
arg0[] elements >= 0.0
arg1[] elements >= 0
this.NMax < this.start
this.start > this.min_sample
this.start != this.max_sample
this.start > this.totalPacketNum
arg1[] elements < this.start
this.start > orig(this.totalPacketNum)
this.min_sample < this.totalPacketNum
arg1[] elements >= this.min_sample
this.min_sample in arg1[]
this.min_sample <= orig(this.totalPacketNum)
this.max_sample > this.totalPacketNum
arg1[] elements < this.max_sample
this.max_sample > orig(this.totalPacketNum)
arg0[] elements >= this.sample_rate
this.sample_rate in arg0[]
this.totalPacketNum - orig(this.totalPacketNum) - 1 == 0
==========
OneModel.distance(float[], float[]):::ENTER
arg0 != null
arg0[] elements >= 0.0
arg1 != null
arg1[] elements >= 0.0
==========
OneModel.distance(float[], float[]):::EXIT
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
arg0[] elements >= 0.0

arg1[] elements >= 0.0
return >= 0.0
===
OneModel.postprocess():::ENTER
this.min_sample == this.reduceN
this.NMax == 20
this.max_sample == 1000
this.totalPacketNum >= 1
this.NMax < this.start
this.start > this.min_sample
this.start != this.max_sample
this.start > this.totalPacketNum
this.min_sample < this.totalPacketNum
this.max_sample > this.totalPacketNum
===
OneModel.postprocess():::EXIT
this.centroids == orig(this.centroids)
this.NMax == orig(this.NMax)
this.start == orig(this.start)
this.start == orig(this.end)
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.reduceN)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.totalPacketNum == orig(this.totalPacketNum)
this.b_packets == orig(this.b_packets)
this.b_packets[] == orig(this.b_packets[])
this.b_packets[] == orig(this.b_nums[])
this.b_nums == orig(this.b_nums)
this.batch == orig(this.batch)
this.NMax == 20
this.max_sample == 1000
this.totalPacketNum >= 1
this.NMax < this.start
this.start > this.min_sample
this.start != this.max_sample
this.start > this.totalPacketNum
this.start > this.reduceN
this.min_sample < this.totalPacketNum
this.max_sample > this.totalPacketNum
this.max_sample > this.reduceN
this.totalPacketNum > this.reduceN
===
OneModel.updateWinner(float[], double, short[]):::ENTER
this.min_sample == this.reduceN
this.NMax == 20
this.max_sample == 1000
this.totalPacketNum >= 1
arg0 != null
arg0[] elements >= 0.0
arg1 == 0.1
arg2 != null
arg2[] elements >= 0
this.NMax < this.start
this.start > this.min_sample
this.start != this.max_sample
this.start > this.totalPacketNum
arg2[] elements < this.start
this.min_sample < this.totalPacketNum
arg2[] elements >= this.min_sample
this.min_sample in arg2[]
this.max_sample > this.totalPacketNum
arg2[] elements < this.max_sample

arg0[] elements >= this.sample_rate
this.sample_rate in arg0[]
==========
OneModel.updateWinner(float[], double, short[]):::EXIT
this.centroids == orig(this.centroids)
this.NMax == orig(this.NMax)
this.start == orig(this.start)
this.start == orig(this.end)
this.min_sample == this.reduceN
this.min_sample == orig(this.min_sample)
this.min_sample == orig(this.reduceN)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.totalPacketNum == orig(this.totalPacketNum)
this.b_packets == orig(this.b_packets)
this.b_packets[] == orig(this.b_packets[])
this.b_packets[] == orig(this.b_nums[])
this.b_nums == orig(this.b_nums)
this.batch == orig(this.batch)
arg0[] == orig(arg0[])
arg2[] == orig(arg2[])
this.NMax == 20
this.max_sample == 1000
this.totalPacketNum >= 1
arg0[] elements >= 0.0
arg2[] elements >= 0
this.NMax < this.start
this.start > this.min_sample
this.start != this.max_sample
this.start > this.totalPacketNum
arg2[] elements < this.start
this.min_sample < this.totalPacketNum
arg2[] elements >= this.min_sample
this.min_sample in arg2[]
this.max_sample > this.totalPacketNum
arg2[] elements < this.max_sample
arg0[] elements >= this.sample_rate
this.sample_rate in arg0[]
==========
Payload:::OBJECT
this.TEST_PACKET_COUNT == this.MM.trunc_length
this.TEST_PACKET_COUNT == this.trainPH.MM.trunc_length
this.MM == this.trainPH.MM
this.MM.TCP_model == this.trainPH.MM.TCP_model
this.MM.TCP_model[].getClass() == this.trainPH.MM.TCP_model[].getClass()
this.MM.UDP_model == this.trainPH.MM.UDP_model
this.MM.UDP_model[] == this.MM.TCP_model_in[]
this.MM.UDP_model[] == this.MM.UDP_model_in[]
this.MM.UDP_model[] == this.MM.TCP_model_out[]
this.MM.UDP_model[] == this.MM.UDP_model_out[]
this.MM.UDP_model[] == this.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDP_model_outsrc[]
this.MM.UDP_model[].getClass() == this.MM.UDP_lengthModel[].getClass()
this.MM.UDP_model[].getClass() == this.trainPH.MM.UDP_model[].getClass()
this.MM.TCP_model_in == this.trainPH.MM.TCP_model_in
this.MM.UDP_model_in == this.trainPH.MM.UDP_model_in
this.MM.TCP_model_out == this.trainPH.MM.TCP_model_out
this.MM.UDP_model_out == this.trainPH.MM.UDP_model_out
this.MM.TCP_model_outsrc == this.trainPH.MM.TCP_model_outsrc
this.MM.UDP_model_outsrc == this.trainPH.MM.UDP_model_outsrc
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel
this.MM.TCP_lengthModel[] == this.trainPH.MM.TCP_lengthModel[]
this.MM.UDP_lengthModel == this.trainPH.MM.UDP_lengthModel

this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.UDP_lengthModel[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.trainPH.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[].getClass() == this.trainPH.MM.UDP_model[].getClass()
this.MM.TCP_lengthModel_in == this.trainPH.MM.TCP_lengthModel_in
this.MM.UDP_lengthModel_in == this.trainPH.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel_out == this.trainPH.MM.TCP_lengthModel_out
this.MM.UDP_lengthModel_out == this.trainPH.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel_outsrc == this.trainPH.MM.TCP_lengthModel_outsrc
this.MM.UDP_lengthModel_outsrc == this.trainPH.MM.UDP_lengthModel_outsrc
this.MM.trunc_length == this.trunc_length
this.MM.trunc_length == this.trainPH.MM.trunc_length
this.MM.port_trunc_length == this.port_trunc_length
this.MM.port_trunc_length == this.trainPH.MM.port_trunc_length
this.MM.min_sample == this.MM.modification_time
this.MM.min_sample == this.min_sample_num
this.MM.min_sample == this.trainPH.MM.min_sample
this.MM.max_sample == this.max_sample_num
this.MM.max_sample == this.trainPH.MM.max_sample
this.MM.sample_rate == this.sample_rate
this.MM.sample_rate == this.trainPH.MM.sample_rate
this.MM.host_ip == this.host_ip
this.MM.host_ip == this.trainPH.host_ip
this.MM.stablesize == this.stablesize
this.MM.stablesize == this.trainPH.MM.stablesize
this.MM.stabledis == this.stabledis
this.MM.stabledis == this.trainPH.MM.stabledis
this.MM.smooth_factor == this.smooth_factor
this.MM.smooth_factor == this.trainPH.MM.smooth_factor
this.MM.aggregation_done == this.protocol_udp
this.MM.aggregation_done == this.file_filtering
this.MM.aggregation_done == this.trainPH.MM.aggregation_done
this.MM.lenProb == this.trainPH.MM.lenProb
this.MM.anomalyLengthThreshold == this.trainPH.MM.anomalyLengthThreshold
this.MM.lenProb_out == this.trainPH.MM.lenProb_out
this.MM.lenProb_out[] == this.MM.lenProb_in[]
this.MM.lenProb_out[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold_out == this.trainPH.MM.anomalyLengthThreshold_out
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.lenProb_in == this.trainPH.MM.lenProb_in
this.MM.anomalyLengthThreshold_in == this.trainPH.MM.anomalyLengthThreshold_in
this.MM.lenProb_outsrc == this.trainPH.MM.lenProb_outsrc
this.MM.anomalyLengthThreshold_outsrc == this.trainPH.MM.anomalyLengthThreshold_outsrc
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength
this.MM.UDPNormalLength[] == this.MM.UDPNormalLength_out[]
this.MM.UDPNormalLength[] == this.MM.UDPNormalLength_in[]
this.MM.UDPNormalLength[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDPNormalLength[] == this.trainPH.MM.UDPNormalLength[]
this.MM.UDPNormalLength[] == this.trainPH.MM.UDPNormalLength_out[]
this.MM.UDPNormalLength[] == this.trainPH.MM.UDPNormalLength_in[]
this.MM.UDPNormalLength[] == this.trainPH.MM.UDPNormalLength_outsrc[]
this.MM.UDPNormalLength_out == this.trainPH.MM.UDPNormalLength_out

this.MM.UDPNormalLength_in == this.trainPH.MM.UDPNormalLength_in
this.MM.UDPNormalLength_outsrc == this.trainPH.MM.UDPNormalLength_outsrc
this.MM.normal_length_perc == this.trainPH.MM.normal_length_perc
this.host_ip == this.trainPH.host_ip
this.in_src_ports[] == this.out_src_ports[]
this.protocol_tcp == this.print
this.protocol_tcp == this.auto_correlation
this.protocol_udp == this.trainPH.MM.aggregation_done
this.trunc_length == this.trainPH.MM.trunc_length
this.port_trunc_length == this.trainPH.MM.port_trunc_length
this.port_keywords == this.trainPH.port_keywords
this.port_keywords_range == this.trainPH.port_keywords_range
this.file_filtering == this.EPOCH_TRAIN_FLAG
this.file_filtering == this.readtextmodel
this.file_filtering == this.communication_on
this.file_filtering == this.trainPH.MM.aggregation_done
this.smooth_factor == this.trainPH.MM.smooth_factor
this.min_sample_num == this.trainPH.MM.min_sample
this.max_sample_num == this.trainPH.MM.max_sample
this.sample_rate == this.trainPH.MM.sample_rate
this.auto_adjust == this.normal_alert
this.auto_adjust == this.OFFLINE_TRAIN
this.auto_adjust == this.OFFLINE_TEST
this.stablesize == this.trainPH.MM.stablesize
this.stabledis == this.trainPH.MM.stabledis
this.in_dst_ports_set == this.trainPH.in_dst_ports_set
this.in_src_ports_set == this.trainPH.in_src_ports_set
this.out_dst_ports_set == this.trainPH.out_dst_ports_set
this.out_src_ports_set == this.trainPH.out_src_ports_set
this.trainPH.packetNum == this.trainPH.content_packetNum
this.trainPH.host_ip == this.trainPH.MM.host_ip
this.trainPH.MM.UDP_model[] == this.trainPH.MM.TCP_model_in[]
this.trainPH.MM.UDP_model[] == this.trainPH.MM.UDP_model_in[]
this.trainPH.MM.UDP_model[] == this.trainPH.MM.TCP_model_out[]
this.trainPH.MM.UDP_model[] == this.trainPH.MM.UDP_model_out[]
this.trainPH.MM.UDP_model[] == this.trainPH.MM.TCP_model_outsrc[]
this.trainPH.MM.UDP_model[] == this.trainPH.MM.UDP_model_outsrc[]
this.trainPH.MM.min_sample == this.trainPH.MM.modification_time
this.trainPH.MM.lenProb_out[] == this.trainPH.MM.lenProb_in[]
this.trainPH.MM.lenProb_out[] == this.trainPH.MM.lenProb_outsrc[]
this.trainPH.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_in[]
this.trainPH.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_outsrc[]
this has only one value
this.PACKET_COUNT one of 1000, 5000
this.TEST_PACKET_COUNT one of -1, 1000
this.FILTER != null
this.FILTER.toString one of "", "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM.TCP_model has only one value
this.MM.TCP_model != null
this.MM.TCP_model.getClass() == java.util.HashMap[].class
this.MM.TCP_model[].getClass() one of [null, null, null,

null, null,

null, null, null,

this.MM.TCP_model_out has only one value
this.MM.TCP_model_out != null
this.MM.UDP_model_out has only one value
this.MM.UDP_model_out != null
this.MM.TCP_model_outsrc has only one value
this.MM.TCP_model_outsrc != null
this.MM.UDP_model_outsrc has only one value
this.MM.UDP_model_outsrc != null
this.MM.TCP_lengthModel.getClass() == java.util.Vector[].class
this.MM.TCP_lengthModel[] has only one value, of length 1024
this.MM.TCP_lengthModel[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0,

this.in_dst_ports has only one value
this.in_dst_ports != null
this.in_dst_ports[] elements has only one value
this.in_dst_ports[] elements != null
this.in_dst_ports[].getClass() elements == java.lang.String.class
this.in_dst_ports[].getClass() one of [], [java.lang.String]
this.in_src_ports has only one value
this.in_src_ports != null
this.in_src_ports[] == []
this.in_src_ports[].getClass() == []
this.out_dst_ports has only one value
this.out_dst_ports != null
this.out_dst_ports[] elements has only one value
this.out_dst_ports[] elements != null
this.out_src_ports has only one value
this.out_src_ports != null
this.protocol_tcp == true
this.trunc_length == -1
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.file_filtering == false
this.smooth_factor == 0.0010
this.min_sample_num one of 0, 20
this.max_sample_num one of 200, 1000
this.sample_rate one of 0.0, 0.05
this.kernel_h == 5.0
this.normal_length_perc == 0.999
this.threshold_step one of 0.1, 0.2
this.fpRate one of 0.0010, 0.01
this.stablesize == 1
this.stabledis == 0.05
this.cluster_threshold one of 0.2, 0.3
this.modelfile != null
this.modelfile.toString == ”model.dat”
this.verbose one of 1, 2
this.offlineTrainFile != null
this.offlineTrainFile.toString one of ””, ”train-p2.dmp”
this.TRAIN_HOUR one of 0.01, 0.02
this.STABLE_PORTS_PERC == 0.9
this.offlineTestFile != null
this.offlineTestFile.toString one of ””, ”test-p3.dmp”
this.string_compare_method != null
this.string_compare_method.toString == ”string”
this.string_compare_threshold == 0.5
this.textmodelfile has only one value
this.textmodelfile != null
this.textmodelfile.toString == ””
this.communication_time one of 0, 1800
this.comm_ips has only one value
this.comm_ips != null
this.comm_ips[] elements != null
this.comm_ips[].getClass() elements == java.lang.String.class
this.comm_ips[].getClass() one of [], [java.lang.String, java.lang.String]
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null

this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.trainPH.packetNum == 2300
this.trainPH.port_keywords has only one value
this.trainPH.port_keywords_range has only one value
this.trainPH.host_ip has only one value
this.trainPH.host_ip.toString == "128.59.16.101"
this.trainPH.in_dst_ports_set has only one value
this.trainPH.in_src_ports_set has only one value
this.trainPH.out_dst_ports_set has only one value
this.trainPH.out_src_ports_set has only one value
this.trainPH.MM has only one value
this.trainPH.MM.TCP_model has only one value
this.trainPH.MM.TCP_model.getClass() == java.util.HashMap[].class
this.trainPH.MM.TCP_model[] has only one value, of length 1024
this.trainPH.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.trainPH.MM.anomalyLengthThreshold[] one of [0.0, 0.0, 0.0,

0.0, 0.0, 0.0,

this.PACKET_COUNT > this.MM.max_sample
this.PACKET_COUNT > this.MM.stablesize
this.PACKET_COUNT > this.trunc_length
this.PACKET_COUNT > this.min_sample_num
this.PACKET_COUNT > this.max_sample_num
this.PACKET_COUNT > this.stablesize
this.PACKET_COUNT > this.verbose
this.PACKET_COUNT > this.communication_time
this.TEST_PACKET_COUNT < this.MM.min_sample
this.TEST_PACKET_COUNT < this.MM.max_sample
this.TEST_PACKET_COUNT < this.MM.stablesize
this.TEST_PACKET_COUNT >= this.trunc_length
this.TEST_PACKET_COUNT != this.min_sample_num
this.TEST_PACKET_COUNT != this.max_sample_num
this.TEST_PACKET_COUNT != this.stablesize
this.TEST_PACKET_COUNT != this.verbose
this.TEST_PACKET_COUNT != this.communication_time
this.FILTER.toString > this.MM.host_ip.toString
this.FILTER.toString != this.config_device.toString
this.FILTER.toString != this.host_ip.toString
this.FILTER.toString < this.modelfile.toString
this.FILTER.toString <= this.offlineTrainFile.toString
this.FILTER.toString <= this.offlineTestFile.toString
this.FILTER.toString < this.string_compare_method.toString
this.FILTER.toString >= this.textmodelfile.toString
this.MM.TCP_model.getClass() == this.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() != this.MM.lenProb.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model.getClass()
this.MM.TCP_model[] == this.trainPH.MM.TCP_model[] (elementwise)
this.MM.UDP_model[] == this.MM.UDPNormalLength[] (elementwise)
this.MM.UDP_model[] is the reverse of this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model[] (elementwise)
this.MM.UDP_model[] is the reverse of this.trainPH.MM.UDP_model[]
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.trainPH.MM.UDP_lengthModel_outsrc.getClass()
this.MM.trunc_length < this.min_sample_num
this.MM.trunc_length < this.max_sample_num
this.MM.trunc_length < this.verbose
this.MM.trunc_length < this.communication_time
this.MM.min_sample < this.max_sample_num
this.MM.min_sample < this.verbose
this.MM.min_sample < this.communication_time
this.MM.max_sample > this.min_sample_num

this.MM.max_sample > this.verbose
this.MM.max_sample < this.communication_time
this.MM.host_ip.toString < this.config_device.toString
this.MM.host_ip.toString == this.host_ip.toString
this.MM.host_ip.toString < this.modelfile.toString
this.MM.host_ip.toString < this.offlineTrainFile.toString
this.MM.host_ip.toString < this.offlineTestFile.toString
this.MM.host_ip.toString < this.string_compare_method.toString
this.MM.host_ip.toString > this.textmodelfile.toString
this.MM.host_ip.toString == this.trainPH.host_ip.toString
this.MM.stablesize > this.min_sample_num
this.MM.stablesize < this.max_sample_num
this.MM.stablesize < this.verbose
this.MM.stablesize < this.communication_time
this.MM.anomalyLengthThreshold[] elements < this.MM.stabledis
this.MM.stabledis != this.MM.normal_length_perc
this.MM.stabledis < this.threshold_step
this.MM.stabledis > this.fpRate
this.MM.stabledis < this.cluster_threshold
this.MM.stabledis > this.TRAIN_HOUR
this.MM.smooth_factor > this.sample_rate
this.MM.lenProb.getClass() == this.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb.getClass()
this.MM.lenProb[] == this.trainPH.MM.lenProb[] (elementwise)
this.MM.anomalyLengthThreshold[] >= this.MM.anomalyLengthThreshold_out[] (elementwise)
this.MM.anomalyLengthThreshold[] elements <= this.MM.normal_length_perc
this.MM.anomalyLengthThreshold[] elements < this.kernel_h
this.MM.anomalyLengthThreshold[] elements < this.normal_length_perc
this.MM.anomalyLengthThreshold[] elements < this.threshold_step
this.MM.anomalyLengthThreshold[] elements < this.fpRate
this.MM.anomalyLengthThreshold[] elements < this.stabledis
this.MM.anomalyLengthThreshold[] elements < this.cluster_threshold
this.MM.anomalyLengthThreshold[] elements < this.TRAIN_HOUR
this.MM.anomalyLengthThreshold[] elements < this.STABLE_PORTS_PERC
this.MM.anomalyLengthThreshold[] elements < this.string_compare_threshold
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold[] (elementwise)
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_out[] (elementwise)
this.MM.lenProb_out[] is the reverse of this.trainPH.MM.lenProb_out[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_out[] (elementwise)
this.MM.anomalyLengthThreshold_out[] is the reverse of this.trainPH.MM.anomalyLengthThreshold_out[]
this.MM.UDPNormalLength.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.UDPNormalLength.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.UDPNormalLength.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.UDPNormalLength.getClass() == this.trainPH.MM.TCP_model.getClass()
this.MM.UDPNormalLength.getClass() == this.trainPH.MM.UDPNormalLength.getClass()
this.MM.UDPNormalLength.getClass() == this.trainPH.MM.UDPNormalLength_out.getClass()
this.MM.UDPNormalLength.getClass() == this.trainPH.MM.UDPNormalLength_in.getClass()
this.MM.UDPNormalLength.getClass() == this.trainPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.UDPNormalLength[] == this.trainPH.MM.UDP_model[] (elementwise)
this.MM.UDPNormalLength[] is the reverse of this.trainPH.MM.UDP_model[]
this.MM.normal_length_perc < this.kernel_h
this.MM.normal_length_perc <= this.normal_length_perc
this.MM.normal_length_perc != this.threshold_step
this.MM.normal_length_perc != this.fpRate
this.MM.normal_length_perc != this.stabledis
this.MM.normal_length_perc != this.cluster_threshold
this.MM.normal_length_perc != this.TRAIN_HOUR
this.MM.normal_length_perc != this.STABLE_PORTS_PERC
this.MM.normal_length_perc != this.string_compare_threshold
this.config_device.toString != this.host_ip.toString
this.config_device.toString < this.modelfile.toString

this.config_device.toString != this.offlineTrainFile.toString
this.config_device.toString != this.offlineTestFile.toString
this.config_device.toString < this.string_compare_method.toString
this.config_device.toString > this.textmodelfile.toString
this.host_ip.toString < this.modelfile.toString
this.host_ip.toString != this.offlineTrainFile.toString
this.host_ip.toString != this.offlineTestFile.toString
this.host_ip.toString < this.string_compare_method.toString
this.host_ip.toString > this.textmodelfile.toString
this.host_ip.toString == this.trainPH.host_ip.toString
this.trunc_length < this.min_sample_num
this.trunc_length < this.max_sample_num
this.trunc_length < this.verbose
this.trunc_length < this.communication_time
this.smooth_factor != this.sample_rate
this.min_sample_num < this.max_sample_num
this.min_sample_num != this.stablesize
this.min_sample_num != this.verbose
this.min_sample_num != this.communication_time
this.max_sample_num > this.stablesize
this.max_sample_num > this.verbose
this.max_sample_num != this.communication_time
this.kernel_h > this.threshold_step
this.kernel_h > this.fpRate
this.kernel_h > this.cluster_threshold
this.kernel_h > this.TRAIN_HOUR
this.normal_length_perc > this.threshold_step
this.normal_length_perc > this.fpRate
this.normal_length_perc > this.cluster_threshold
this.normal_length_perc > this.TRAIN_HOUR
this.threshold_step > this.fpRate
this.threshold_step > this.stabledis
this.threshold_step <= this.cluster_threshold
this.threshold_step > this.TRAIN_HOUR
this.threshold_step < this.STABLE_PORTS_PERC
this.threshold_step < this.string_compare_threshold
this.fpRate < this.stabledis
this.fpRate < this.cluster_threshold
this.fpRate <= this.TRAIN_HOUR
this.fpRate < this.STABLE_PORTS_PERC
this.fpRate < this.string_compare_threshold
this.stablesize <= this.verbose
this.stablesize != this.communication_time
this.stabledis < this.cluster_threshold
this.stabledis > this.TRAIN_HOUR
this.cluster_threshold > this.TRAIN_HOUR
this.cluster_threshold < this.STABLE_PORTS_PERC
this.cluster_threshold < this.string_compare_threshold
this.modelfile.toString != this.offlineTrainFile.toString
this.modelfile.toString != this.offlineTestFile.toString
this.modelfile.toString < this.string_compare_method.toString
this.modelfile.toString > this.textmodelfile.toString
this.verbose != this.communication_time
this.offlineTrainFile.toString >= this.offlineTestFile.toString
this.offlineTrainFile.toString != this.string_compare_method.toString
this.offlineTrainFile.toString >= this.textmodelfile.toString
this.TRAIN_HOUR < this.STABLE_PORTS_PERC
this.TRAIN_HOUR < this.string_compare_threshold
this.offlineTestFile.toString != this.string_compare_method.toString
this.offlineTestFile.toString >= this.textmodelfile.toString
this.string_compare_method.toString > this.textmodelfile.toString
this.trainPH.host_ip.toString == this.trainPH.MM.host_ip.toString
this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model.getClass()

this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_in.getClass()
this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_in.getClass()
this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_out.getClass()
this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_out.getClass()
this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_outsrc.getClass()
this.trainPH.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_outsrc.getClass()
this.trainPH.MM.lenProb.getClass() == this.trainPH.MM.lenProb_out.getClass()
this.trainPH.MM.lenProb.getClass() == this.trainPH.MM.lenProb_in.getClass()
this.trainPH.MM.lenProb.getClass() == this.trainPH.MM.lenProb_outsrc.getClass()
==========
Payload.Payload():::EXIT
this.PACKET_COUNT == this.TEST_PACKET_COUNT
this.FILTER == this.offlineTrainFile
this.FILTER == this.offlineTestFile
this.FILTER == this.textmodelfile
this.in_dst_ports[] == this.in_src_ports[]
this.in_dst_ports[] == this.out_dst_ports[]
this.in_dst_ports[] == this.out_src_ports[]
this.in_dst_ports[] == this.comm_ips[]
this.stablesize == this.verbose
this.PACKET_COUNT == 1000
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "localhost"
this.in_dst_ports[] == []
this.in_dst_ports[].getClass() == []
this.min_sample_num == 20
this.max_sample_num == 200
this.sample_rate == 0.05
this.threshold_step == 0.1
this.fpRate == 0.01
this.cluster_threshold == 0.3
this.modelfile has only one value
this.string_compare_method has only one value
this.communication_time == 0
this.trainPH == null
this.FILTER.toString == this.offlineTrainFile.toString
this.FILTER.toString == this.offlineTestFile.toString
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.buildPortModel():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]

this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
==========
Payload.buildPortModel():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.MM.UDP_model[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_in[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_in[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_out[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_out[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.MM.lenProb_out[])
this.MM.lenProb[] == orig(this.MM.lenProb_in[])
this.MM.lenProb[] == orig(this.MM.lenProb_outsrc[])

this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)

this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.MM.UDPNormalLength == null
this.MM.normal_length_perc == 0.0
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.buildport(java.util.BitSet, java.util.Vector):::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out

this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
arg1[].getClass() elements == java.lang.String.class
arg1[].getClass() one of [], [java.lang.String]
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
===
Payload.buildport(java.util.BitSet, java.util.Vector):::EXIT422
this.in_src_ports[] == arg1[]
this.out_src_ports[] == arg1[]
arg1[] == orig(this.in_src_ports[])
arg1[] == orig(this.out_src_ports[])
arg1[] == []
arg1[].getClass() == []
===
Payload.buildport(java.util.BitSet, java.util.Vector):::EXIT439
arg1[].getClass() == [java.lang.String]
===
Payload.buildport(java.util.BitSet, java.util.Vector):::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.MM.UDP_model[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_in[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_in[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_out[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_out[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out

this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.MM.lenProb_out[])
this.MM.lenProb[] == orig(this.MM.lenProb_in[])
this.MM.lenProb[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)

this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
arg1[] == orig(arg1[])
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null

this.MM.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.MM.anomalyLengthThreshold[] elements == 0.0
this.MM.UDPNormalLength == null
this.MM.normal_length_perc == 0.0
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
arg1[].getClass() elements == java.lang.String.class
arg1[].getClass() one of [], [java.lang.String]
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]

this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.init():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.FILTER == this.textmodelfile
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.init():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]

this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.config_device == orig(this.config_device)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)

this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.FILTER)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null,

this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == ”train-p2.dmp”
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == ”test-p3.dmp”
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.FILTER.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.main(java.lang.String[]):::ENTER
arg0 has only one value
arg0.getClass() == java.lang.String[].class
arg0[] contains no nulls and has only one value, of length 3
arg0[].toString == [train, +g, globalFile.txt]
arg0[].toString elements one of ”+g”, ”globalFile.txt”, ”train”
==========
Payload.main(java.lang.String[]):::EXIT
arg0[] == orig(arg0[])
arg0[] contains no nulls and has only one value, of length 3
arg0[].toString == [train, +g, globalFile.txt]
arg0[].toString elements one of ”+g”, ”globalFile.txt”, ”train”
==========
Payload.portsString(java.util.Vector, java.lang.String):::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.FILTER == this.textmodelfile
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ””
this.MM == null
this.host_ip has only one value
this.host_ip.toString == ”128.59.16.101”
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010

this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
arg1 has only one value
arg1.toString == "dst"
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.portsString(java.util.Vector, java.lang.String):::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == this.textmodelfile
this.FILTER == orig(this.FILTER)
this.FILTER == orig(this.textmodelfile)
this.MM == orig(this.MM)
this.config_device == orig(this.config_device)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)

this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
arg0[] == orig(arg0[])
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
arg0[].getClass() == [java.lang.String]
arg0[].getClass() elements == java.lang.String.class
arg1.toString == "dst"
return.toString == "(dst port 80)"
this.FILTER.toString == this.textmodelfile.toString
this.FILTER.toString == orig(this.FILTER.toString)
this.FILTER.toString == orig(this.textmodelfile.toString)
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.host_ip.toString)

this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
arg1.toString == orig(arg1.toString)
==========
Payload.processModel():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.TCP_model[] == this.trainPH.MM.TCP_model[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.lenProb[] == this.trainPH.MM.lenProb[]
this.MM.lenProb[] == this.trainPH.MM.lenProb_out[]
this.MM.lenProb[] == this.trainPH.MM.lenProb_in[]
this.MM.lenProb[] == this.trainPH.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength_outsrc
this.host_ip == this.trainPH.MM.host_ip
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.min_sample_num == this.trainPH.MM.modification_time
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000

this.TEST_PACKET_COUNT == -1

this.FILTER has only one value

this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"

this.MM has only one value

this.MM.TCP_model[] has only one value, of length 1024

this.MM.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
this.host_ip.toString == this.trainPH.MM.host_ip.toString
==========
Payload.processModel():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trainPH.MM.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM == orig(this.trainPH.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model == orig(this.trainPH.MM.TCP_model)
this.MM.TCP_model[] == this.trainPH.MM.TCP_model[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.trainPH.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model == orig(this.trainPH.MM.UDP_model)
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.trainPH.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.trainPH.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.trainPH.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.TCP_model_in == orig(this.trainPH.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.UDP_model_in == orig(this.trainPH.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.TCP_model_out == orig(this.trainPH.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.UDP_model_out == orig(this.trainPH.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.TCP_model_outsrc == orig(this.trainPH.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.trainPH.MM.UDP_model_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)

this.MM.lenProb == orig(this.trainPH.MM.lenProb)
this.MM.lenProb[] == this.trainPH.MM.lenProb[]
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold == orig(this.trainPH.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold[]
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out == orig(this.trainPH.MM.lenProb_out)
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_outsrc[]
this.MM.lenProb_out[] == orig(this.MM.lenProb[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out == orig(this.trainPH.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.lenProb_in == orig(this.trainPH.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.anomalyLengthThreshold_in == orig(this.trainPH.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.lenProb_outsrc == orig(this.trainPH.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.trainPH.MM.anomalyLengthThreshold_outsrc)
this.MM.normal_length_perc == this.normal_length_perc
this.config_device == orig(this.config_device)
this.host_ip == this.trainPH.MM.host_ip
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.host_ip == orig(this.trainPH.host_ip)
this.host_ip == orig(this.trainPH.MM.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)

this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.protocol_udp == orig(this.trainPH.MM.aggregation_done)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.trainPH.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords == orig(this.trainPH.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.port_keywords_range == orig(this.trainPH.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.smooth_factor == orig(this.trainPH.MM.smooth_factor)
this.min_sample_num == this.trainPH.MM.modification_time
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.min_sample_num == orig(this.trainPH.MM.min_sample)
this.min_sample_num == orig(this.trainPH.MM.modification_time)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.max_sample_num == orig(this.trainPH.MM.max_sample)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.sample_rate == orig(this.trainPH.MM.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == this.trainPH.MM.normal_length_perc
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stablesize == orig(this.trainPH.MM.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.stabledis == orig(this.trainPH.MM.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_dst_ports_set == orig(this.trainPH.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)

this.in_src_ports_set == orig(this.trainPH.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_dst_ports_set == orig(this.trainPH.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.out_src_ports_set == orig(this.trainPH.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.trainPH.packetNum == orig(this.trainPH.packetNum)
this.trainPH.packetNum == orig(this.trainPH.content_packetNum)
this.trainPH.TCP == orig(this.trainPH.TCP)
this.trainPH.UDP == orig(this.trainPH.UDP)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null]
this.MM.TCP_lengthModel has only one value
this.MM.UDP_lengthModel has only one value
this.MM.TCP_lengthModel_in has only one value
this.MM.UDP_lengthModel_in has only one value
this.MM.TCP_lengthModel_out has only one value
this.MM.UDP_lengthModel_out has only one value
this.MM.TCP_lengthModel_outsrc has only one value
this.MM.UDP_lengthModel_outsrc has only one value
this.MM.lenProb[] has only one value, of length 1024
this.MM.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.offlineTrainFile has only one value
this.offlineTrainFile.toString == ”train-p2.dmp”
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == ”test-p3.dmp”
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH has only one value
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model_outsrc.getClass())
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb_outsrc.getClass())
orig(this.MM.normal_length_perc) in this.MM.anomalyLengthThreshold[]
orig(this.MM.normal_length_perc) in this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] elements == orig(this.MM.normal_length_perc)
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == this.trainPH.MM.host_ip.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.host_ip.toString == orig(this.trainPH.host_ip.toString)
this.host_ip.toString == orig(this.trainPH.MM.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)

this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.readGlobalFile(java.lang.String):::ENTER
this.PACKET_COUNT == this.TEST_PACKET_COUNT
this.FILTER == this.offlineTrainFile
this.FILTER == this.offlineTestFile
this.FILTER == this.textmodelfile
this.in_dst_ports[] == this.in_src_ports[]
this.in_dst_ports[] == this.out_dst_ports[]
this.in_dst_ports[] == this.out_src_ports[]
this.in_dst_ports[] == this.comm_ips[]
this.stablesize == this.verbose
this.PACKET_COUNT == 1000
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "localhost"
this.in_dst_ports[] == []
this.in_dst_ports[].getClass() == []
this.min_sample_num == 20
this.max_sample_num == 200
this.sample_rate == 0.05
this.threshold_step == 0.1
this.fpRate == 0.01
this.cluster_threshold == 0.3
this.modelfile has only one value
this.string_compare_method has only one value
this.communication_time == 0
this.trainPH == null
arg0 has only one value
arg0.toString == "globalFile.txt"
this.FILTER.toString == this.offlineTrainFile.toString
this.FILTER.toString == this.offlineTestFile.toString
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.readGlobalFile(java.lang.String):::EXIT
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == this.textmodelfile
this.FILTER == orig(this.FILTER)
this.FILTER == orig(this.offlineTrainFile)
this.FILTER == orig(this.offlineTestFile)
this.FILTER == orig(this.textmodelfile)
this.MM == orig(this.MM)
this.config_device == orig(this.config_device)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_dst_ports[])
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_dst_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.in_src_ports[] == orig(this.comm_ips[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_src_ports == orig(this.out_src_ports)
this.min_sample_num == orig(this.communication_time)
this.max_sample_num == orig(this.PACKET_COUNT)
this.max_sample_num == orig(this.TEST_PACKET_COUNT)
this.stablesize == orig(this.stablesize)
this.stablesize == orig(this.verbose)
this.comm_ips == orig(this.comm_ips)

this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
arg0.toString == "globalFile.txt"
this.FILTER.toString == this.textmodelfile.toString
this.FILTER.toString == orig(this.FILTER.toString)
this.FILTER.toString == orig(this.offlineTrainFile.toString)
this.FILTER.toString == orig(this.offlineTestFile.toString)
this.FILTER.toString == orig(this.textmodelfile.toString)
this.config_device.toString == orig(this.config_device.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
arg0.toString == orig(arg0.toString)
==========
Payload.startCalibrate():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.TCP_model[] == this.trainPH.MM.TCP_model[]
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_out[]

this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_outsrc[]
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.trainPH.MM.lenProb[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.normal_length_perc == this.normal_length_perc
this.host_ip == this.trainPH.MM.host_ip
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.min_sample_num == this.trainPH.MM.modification_time
this.normal_length_perc == this.trainPH.MM.normal_length_perc
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH has only one value
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_outsrc.getClass()
this.host_ip.toString == this.trainPH.MM.host_ip.toString
==========
Payload.startCalibrate():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trainPH.MM.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM == orig(this.trainPH.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model == orig(this.trainPH.MM.TCP_model)
this.MM.TCP_model[] == this.trainPH.MM.TCP_model[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.trainPH.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model == orig(this.trainPH.MM.UDP_model)
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_in[]

this.MM.UDP_model[] == this.trainPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.trainPH.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.trainPH.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.trainPH.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.trainPH.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.TCP_model_in == orig(this.trainPH.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.UDP_model_in == orig(this.trainPH.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.TCP_model_out == orig(this.trainPH.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.UDP_model_out == orig(this.trainPH.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.TCP_model_outsrc == orig(this.trainPH.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.trainPH.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.trainPH.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.TCP_lengthModel[] == orig(this.trainPH.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel == orig(this.trainPH.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.trainPH.MM.UDP_lengthModel_outsrc[])

this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel_in == orig(this.trainPH.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.trainPH.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel_out == orig(this.trainPH.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.trainPH.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel_outsrc == orig(this.trainPH.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.trainPH.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb == orig(this.trainPH.MM.lenProb)
this.MM.lenProb[] == this.trainPH.MM.lenProb[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.trainPH.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold == orig(this.trainPH.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.trainPH.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out == orig(this.trainPH.MM.lenProb_out)
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.trainPH.MM.lenProb_outsrc[]
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.trainPH.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out == orig(this.trainPH.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.trainPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.trainPH.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.lenProb_in == orig(this.trainPH.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.anomalyLengthThreshold_in == orig(this.trainPH.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.lenProb_outsrc == orig(this.trainPH.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.trainPH.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.trainPH.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_out == orig(this.trainPH.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_in == orig(this.trainPH.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.UDPNormalLength_outsrc == orig(this.trainPH.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == this.normal_length_perc
this.config_device == orig(this.config_device)

this.host_ip == this.trainPH.MM.host_ip
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.host_ip == orig(this.trainPH.host_ip)
this.host_ip == orig(this.trainPH.MM.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.protocol_udp == orig(this.trainPH.MM.aggregation_done)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.trainPH.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords == orig(this.trainPH.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.port_keywords_range == orig(this.trainPH.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.smooth_factor == orig(this.trainPH.MM.smooth_factor)
this.min_sample_num == this.trainPH.MM.modification_time
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.min_sample_num == orig(this.trainPH.MM.min_sample)
this.min_sample_num == orig(this.trainPH.MM.modification_time)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.max_sample_num == orig(this.trainPH.MM.max_sample)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.sample_rate == orig(this.trainPH.MM.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == this.trainPH.MM.normal_length_perc
this.normal_length_perc == orig(this.MM.normal_length_perc)
this.normal_length_perc == orig(this.normal_length_perc)
this.normal_length_perc == orig(this.trainPH.MM.normal_length_perc)
this.threshold_step == this.cluster_threshold

this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stablesize == orig(this.trainPH.MM.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.stabledis == orig(this.trainPH.MM.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_dst_ports_set == orig(this.trainPH.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.in_src_ports_set == orig(this.trainPH.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_dst_ports_set == orig(this.trainPH.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.out_src_ports_set == orig(this.trainPH.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.trainPH.packetNum == orig(this.trainPH.packetNum)
this.trainPH.packetNum == orig(this.trainPH.content_packetNum)
this.trainPH.TCP == orig(this.trainPH.TCP)
this.trainPH.UDP == orig(this.trainPH.UDP)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.MM.UDPNormalLength has only one value
this.MM.UDPNormalLength_out has only one value
this.MM.UDPNormalLength_in has only one value
this.MM.UDPNormalLength_outsrc has only one value
this.host_ip has only one value
this.host_ip.toString == ”128.59.16.101”
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == ”train-p2.dmp”
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == ”test-p3.dmp”
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH has only one value
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.trainPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model.getClass())

this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.trainPH.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.trainPH.MM.UDP_lengthModel_outsrc.getClass())
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.trainPH.MM.lenProb_outsrc.getClass())
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == this.trainPH.MM.host_ip.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.host_ip.toString == orig(this.trainPH.host_ip.toString)
this.host_ip.toString == orig(this.trainPH.MM.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.startTrain():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out

this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.trainPH == null
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
==========
Payload.startTrain():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.trainPH.MM.TCP_model[]
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == this.trainPH.MM.UDP_model[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.trainPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.trainPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.trainPH.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.trainPH.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel)

this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.lenProb[] == this.trainPH.MM.lenProb[]
this.MM.lenProb[] == this.trainPH.MM.lenProb_out[]
this.MM.lenProb[] == this.trainPH.MM.lenProb_in[]
this.MM.lenProb[] == this.trainPH.MM.lenProb_outsrc[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.MM.lenProb_out[])
this.MM.lenProb[] == orig(this.MM.lenProb_in[])
this.MM.lenProb[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.trainPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.trainPH.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == this.trainPH.MM.host_ip
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN

this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == this.trainPH.MM.modification_time
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1

this.FILTER has only one value

this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"

this.MM has only one value

this.MM.TCP_model[] has only one value, of length 1024

this.MM.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.trainPH.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == this.trainPH.MM.host_ip.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
TrainPacketHandler:::OBJECT
this.packetNum == this.content_packetNum
this.host_ip == this.MM.host_ip
this.MM.UDP_model[] == this.MM.TCP_model_in[]
this.MM.UDP_model[] == this.MM.UDP_model_in[]
this.MM.UDP_model[] == this.MM.TCP_model_out[]
this.MM.UDP_model[] == this.MM.UDP_model_out[]
this.MM.UDP_model[] == this.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.min_sample == this.MM.modification_time
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this has only one value
this.host_ip has only one value
this.host_ip != null
this.host_ip.toString == "128.59.16.101"
this.MM has only one value
this.MM != null
this.MM.TCP_model has only one value
this.MM.TCP_model != null
this.MM.TCP_model.getClass() == java.util.HashMap[].class
this.MM.TCP_model[].getClass() one of [null, null, null,

null, null,

null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.MM.anomalyLengthThreshold[] elements == 0.0
this.MM.lenProb_out has only one value
this.MM.lenProb_out != null
this.MM.anomalyLengthThreshold_out has only one value
this.MM.anomalyLengthThreshold_out != null
this.MM.lenProb_in has only one value
this.MM.lenProb_in != null
this.MM.anomalyLengthThreshold_in has only one value
this.MM.anomalyLengthThreshold_in != null
this.MM.lenProb_outsrc has only one value
this.MM.lenProb_outsrc != null
this.MM.anomalyLengthThreshold_outsrc has only one value
this.MM.anomalyLengthThreshold_outsrc != null
this.MM.UDPNormalLength == null
this.MM.normal_length_perc == 0.0
this.TCP == 6
this.UDP == 17
this.packetNum > this.MM.trunc_length
this.packetNum >= this.MM.min_sample
this.host_ip.toString == this.MM.host_ip.toString
this.MM.TCP_model.getClass() == this.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() != this.MM.lenProb.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_outsrc.getClass()
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
==========
TrainPacketHandler.TrainPacketHandler(java.lang.String, Models):::ENTER
arg0 == arg1.host_ip
arg1.TCP_model[] == arg1.UDP_model[]
arg1.TCP_model[] == arg1.TCP_model_in[]
arg1.TCP_model[] == arg1.UDP_model_in[]
arg1.TCP_model[] == arg1.TCP_model_out[]
arg1.TCP_model[] == arg1.UDP_model_out[]
arg1.TCP_model[] == arg1.TCP_model_outsrc[]
arg1.TCP_model[] == arg1.UDP_model_outsrc[]
arg1.TCP_lengthModel == arg1.UDP_lengthModel
arg1.TCP_lengthModel == arg1.TCP_lengthModel_in
arg1.TCP_lengthModel == arg1.UDP_lengthModel_in
arg1.TCP_lengthModel == arg1.TCP_lengthModel_out
arg1.TCP_lengthModel == arg1.UDP_lengthModel_out
arg1.TCP_lengthModel == arg1.TCP_lengthModel_outsrc
arg1.TCP_lengthModel == arg1.UDP_lengthModel_outsrc
arg1.min_sample == arg1.modification_time
arg1.lenProb[] == arg1.lenProb_out[]
arg1.lenProb[] == arg1.lenProb_in[]
arg1.lenProb[] == arg1.lenProb_outsrc[]
arg1.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold_out[]
arg1.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold_in[]
arg1.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold_outsrc[]

arg1.UDPNormalLength == arg1.UDPNormalLength_out
arg1.UDPNormalLength == arg1.UDPNormalLength_in
arg1.UDPNormalLength == arg1.UDPNormalLength_outsrc
arg0 has only one value
arg0.toString == "128.59.16.101"
arg1 has only one value
arg1.TCP_model has only one value
arg1.TCP_model.getClass() == java.util.HashMap[].class
arg1.TCP_model[] contains only nulls and has only one value, of length 1024
arg1.TCP_model[] elements == null
arg1.TCP_model[].getClass() == [null, null, null,

arg1.port_trunc_length has only one value
arg1.min_sample == 0
arg1.max_sample == 1000
arg1.sample_rate == 0.0
arg1.stablesize == 1
arg1.stabledis == 0.05
arg1.smooth_factor == 0.0010
arg1.aggregation_done == false
arg1.lenProb has only one value
arg1.lenProb.getClass() == double[][].class
arg1.lenProb[] contains only nulls and has only one value, of length 1024
arg1.lenProb[] elements == null
arg1.anomalyLengthThreshold has only one value
arg1.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

arg1.TCP_model.getClass() == arg1.UDP_model_out.getClass()
arg1.TCP_model.getClass() == arg1.TCP_model_outsrc.getClass()
arg1.TCP_model.getClass() == arg1.UDP_model_outsrc.getClass()
arg1.lenProb.getClass() == arg1.lenProb_out.getClass()
arg1.lenProb.getClass() == arg1.lenProb_in.getClass()
arg1.lenProb.getClass() == arg1.lenProb_outsrc.getClass()
arg1.anomalyLengthThreshold[] elements == arg1.normal_length_perc
arg1.normal_length_perc in arg1.anomalyLengthThreshold[]
==========
TrainPacketHandler.TrainPacketHandler(java.lang.String, Models):::EXIT
this.packetNum == this.MM.min_sample
this.packetNum == this.MM.modification_time
this.packetNum == arg1.min_sample
this.packetNum == arg1.modification_time
this.packetNum == orig(arg1.min_sample)
this.packetNum == orig(arg1.modification_time)
this.port_keywords == this.port_keywords_range
this.host_ip == arg1.host_ip
this.host_ip == orig(arg0)
this.host_ip == orig(arg1.host_ip)
this.in_dst_ports_set == this.in_src_ports_set
this.in_dst_ports_set == this.out_dst_ports_set
this.in_dst_ports_set == this.out_src_ports_set
this.MM == orig(arg1)
this.MM.TCP_model == arg1.TCP_model
this.MM.TCP_model[] == arg1.TCP_model[]
this.MM.UDP_model == arg1.UDP_model
this.MM.UDP_model[] == arg1.TCP_model[]
this.MM.TCP_model_in == arg1.TCP_model_in
this.MM.TCP_model_in[] == arg1.TCP_model[]
this.MM.UDP_model_in == arg1.UDP_model_in
this.MM.UDP_model_in[] == arg1.TCP_model[]
this.MM.TCP_model_out == arg1.TCP_model_out
this.MM.TCP_model_out[] == arg1.TCP_model[]
this.MM.UDP_model_out == arg1.UDP_model_out
this.MM.UDP_model_out[] == arg1.TCP_model[]
this.MM.TCP_model_outsrc == arg1.TCP_model_outsrc
this.MM.TCP_model_outsrc[] == arg1.TCP_model[]
this.MM.UDP_model_outsrc == arg1.UDP_model_outsrc
this.MM.UDP_model_outsrc[] == arg1.TCP_model[]
this.MM.TCP_lengthModel == arg1.TCP_lengthModel
this.MM.UDP_lengthModel == arg1.TCP_lengthModel
this.MM.TCP_lengthModel_in == arg1.TCP_lengthModel
this.MM.UDP_lengthModel_in == arg1.TCP_lengthModel
this.MM.TCP_lengthModel_out == arg1.TCP_lengthModel
this.MM.UDP_lengthModel_out == arg1.TCP_lengthModel
this.MM.TCP_lengthModel_outsrc == arg1.TCP_lengthModel
this.MM.UDP_lengthModel_outsrc == arg1.TCP_lengthModel
this.MM.trunc_length == arg1.trunc_length
this.MM.port_trunc_length == arg1.port_trunc_length
this.MM.max_sample == arg1.max_sample
this.MM.sample_rate == arg1.sample_rate
this.MM.stablesize == arg1.stablesize
this.MM.stabledis == arg1.stabledis
this.MM.smooth_factor == arg1.smooth_factor
this.MM.aggregation_done == arg1.aggregation_done
this.MM.lenProb == arg1.lenProb
this.MM.lenProb[] == arg1.lenProb[]
this.MM.anomalyLengthThreshold == arg1.anomalyLengthThreshold
this.MM.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold[]
this.MM.lenProb_out == arg1.lenProb_out
this.MM.lenProb_out[] == arg1.lenProb[]
this.MM.anomalyLengthThreshold_out == arg1.anomalyLengthThreshold_out

this.MM.anomalyLengthThreshold_out[] == arg1.anomalyLengthThreshold[]
this.MM.lenProb_in == arg1.lenProb_in
this.MM.lenProb_in[] == arg1.lenProb[]
this.MM.anomalyLengthThreshold_in == arg1.anomalyLengthThreshold_in
this.MM.anomalyLengthThreshold_in[] == arg1.anomalyLengthThreshold[]
this.MM.lenProb_outsrc == arg1.lenProb_outsrc
this.MM.lenProb_outsrc[] == arg1.lenProb[]
this.MM.anomalyLengthThreshold_outsrc == arg1.anomalyLengthThreshold_outsrc
this.MM.anomalyLengthThreshold_outsrc[] == arg1.anomalyLengthThreshold[]
this.MM.UDPNormalLength == arg1.UDPNormalLength
this.MM.UDPNormalLength_out == arg1.UDPNormalLength
this.MM.UDPNormalLength_in == arg1.UDPNormalLength
this.MM.UDPNormalLength_outsrc == arg1.UDPNormalLength
this.MM.normal_length_perc == arg1.normal_length_perc
arg1.TCP_model == orig(arg1.TCP_model)
arg1.TCP_model[] == arg1.UDP_model[]
arg1.TCP_model[] == arg1.TCP_model_in[]
arg1.TCP_model[] == arg1.UDP_model_in[]
arg1.TCP_model[] == arg1.TCP_model_out[]
arg1.TCP_model[] == arg1.UDP_model_out[]
arg1.TCP_model[] == arg1.TCP_model_outsrc[]
arg1.TCP_model[] == arg1.UDP_model_outsrc[]
arg1.TCP_model[] == orig(arg1.TCP_model[])
arg1.TCP_model[] == orig(arg1.UDP_model[])
arg1.TCP_model[] == orig(arg1.TCP_model_in[])
arg1.TCP_model[] == orig(arg1.UDP_model_in[])
arg1.TCP_model[] == orig(arg1.TCP_model_out[])
arg1.TCP_model[] == orig(arg1.UDP_model_out[])
arg1.TCP_model[] == orig(arg1.TCP_model_outsrc[])
arg1.TCP_model[] == orig(arg1.UDP_model_outsrc[])
arg1.UDP_model == orig(arg1.UDP_model)
arg1.TCP_model_in == orig(arg1.TCP_model_in)
arg1.UDP_model_in == orig(arg1.UDP_model_in)
arg1.TCP_model_out == orig(arg1.TCP_model_out)
arg1.UDP_model_out == orig(arg1.UDP_model_out)
arg1.TCP_model_outsrc == orig(arg1.TCP_model_outsrc)
arg1.UDP_model_outsrc == orig(arg1.UDP_model_outsrc)
arg1.TCP_lengthModel == arg1.UDP_lengthModel
arg1.TCP_lengthModel == arg1.TCP_lengthModel_in
arg1.TCP_lengthModel == arg1.UDP_lengthModel_in
arg1.TCP_lengthModel == arg1.TCP_lengthModel_out
arg1.TCP_lengthModel == arg1.UDP_lengthModel_out
arg1.TCP_lengthModel == arg1.TCP_lengthModel_outsrc
arg1.TCP_lengthModel == arg1.UDP_lengthModel_outsrc
arg1.TCP_lengthModel == orig(arg1.TCP_lengthModel)
arg1.TCP_lengthModel == orig(arg1.UDP_lengthModel)
arg1.TCP_lengthModel == orig(arg1.TCP_lengthModel_in)
arg1.TCP_lengthModel == orig(arg1.UDP_lengthModel_in)
arg1.TCP_lengthModel == orig(arg1.TCP_lengthModel_out)
arg1.TCP_lengthModel == orig(arg1.UDP_lengthModel_out)
arg1.TCP_lengthModel == orig(arg1.TCP_lengthModel_outsrc)
arg1.TCP_lengthModel == orig(arg1.UDP_lengthModel_outsrc)
arg1.trunc_length == orig(arg1.trunc_length)
arg1.port_trunc_length == orig(arg1.port_trunc_length)
arg1.max_sample == orig(arg1.max_sample)
arg1.sample_rate == orig(arg1.sample_rate)
arg1.stablesize == orig(arg1.stablesize)
arg1.stabledis == orig(arg1.stabledis)
arg1.smooth_factor == orig(arg1.smooth_factor)
arg1.aggregation_done == orig(arg1.aggregation_done)
arg1.lenProb == orig(arg1.lenProb)
arg1.lenProb[] == arg1.lenProb_out[]
arg1.lenProb[] == arg1.lenProb_in[]

arg1.lenProb[] == arg1.lenProb_outsrc[]
arg1.lenProb[] == orig(arg1.lenProb[])
arg1.lenProb[] == orig(arg1.lenProb_out[])
arg1.lenProb[] == orig(arg1.lenProb_in[])
arg1.lenProb[] == orig(arg1.lenProb_outsrc[])
arg1.anomalyLengthThreshold == orig(arg1.anomalyLengthThreshold)
arg1.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold_out[]
arg1.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold_in[]
arg1.anomalyLengthThreshold[] == arg1.anomalyLengthThreshold_outsrc[]
arg1.anomalyLengthThreshold[] == orig(arg1.anomalyLengthThreshold[])
arg1.anomalyLengthThreshold[] == orig(arg1.anomalyLengthThreshold_out[])
arg1.anomalyLengthThreshold[] == orig(arg1.anomalyLengthThreshold_in[])
arg1.anomalyLengthThreshold[] == orig(arg1.anomalyLengthThreshold_outsrc[])
arg1.lenProb_out == orig(arg1.lenProb_out)
arg1.anomalyLengthThreshold_out == orig(arg1.anomalyLengthThreshold_out)
arg1.lenProb_in == orig(arg1.lenProb_in)
arg1.anomalyLengthThreshold_in == orig(arg1.anomalyLengthThreshold_in)
arg1.lenProb_outsrc == orig(arg1.lenProb_outsrc)
arg1.anomalyLengthThreshold_outsrc == orig(arg1.anomalyLengthThreshold_outsrc)
arg1.UDPNormalLength == arg1.UDPNormalLength_out
arg1.UDPNormalLength == arg1.UDPNormalLength_in
arg1.UDPNormalLength == arg1.UDPNormalLength_outsrc
arg1.UDPNormalLength == orig(arg1.UDPNormalLength)
arg1.UDPNormalLength == orig(arg1.UDPNormalLength_out)
arg1.UDPNormalLength == orig(arg1.UDPNormalLength_in)
arg1.UDPNormalLength == orig(arg1.UDPNormalLength_outsrc)
arg1.normal_length_perc == orig(arg1.normal_length_perc)
this.packetNum == 0
this.port_keywords == null
this.in_dst_ports_set == null
arg0.toString == "128.59.16.101"
arg1.TCP_model has only one value
arg1.TCP_model.getClass() == java.util.HashMap[].class
arg1.TCP_model[] contains only nulls and has only one value, of length 1024
arg1.TCP_model[] elements == null
arg1.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
arg1.anomalyLengthThreshold[] elements == 0.0
arg1.lenProb_out has only one value
arg1.anomalyLengthThreshold_out has only one value
arg1.lenProb_in has only one value
arg1.anomalyLengthThreshold_in has only one value
arg1.lenProb_outsrc has only one value
arg1.anomalyLengthThreshold_outsrc has only one value
arg1.UDPNormalLength == null
arg1.normal_length_perc == 0.0
arg1.TCP_model[] elements == this.port_keywords
this.port_keywords in arg1.TCP_model[]
this.host_ip.toString == arg0.toString
this.MM.TCP_model.getClass() == arg1.TCP_model.getClass()
this.MM.UDP_model.getClass() == arg1.TCP_model.getClass()
this.MM.TCP_model_in.getClass() == arg1.TCP_model.getClass()
this.MM.UDP_model_in.getClass() == arg1.TCP_model.getClass()
this.MM.TCP_model_out.getClass() == arg1.TCP_model.getClass()
this.MM.UDP_model_out.getClass() == arg1.TCP_model.getClass()
this.MM.TCP_model_outsrc.getClass() == arg1.TCP_model.getClass()
this.MM.UDP_model_outsrc.getClass() == arg1.TCP_model.getClass()
this.MM.host_ip.toString == arg0.toString
this.MM.lenProb.getClass() == arg1.lenProb.getClass()
this.MM.lenProb_out.getClass() == arg1.lenProb.getClass()
this.MM.lenProb_in.getClass() == arg1.lenProb.getClass()
this.MM.lenProb_outsrc.getClass() == arg1.lenProb.getClass()
arg0.toString == arg1.host_ip.toString
arg0.toString == orig(arg0.toString)
arg0.toString == orig(arg1.host_ip.toString)
arg1.TCP_model.getClass() == arg1.UDP_model.getClass()
arg1.TCP_model.getClass() == arg1.TCP_model_in.getClass()
arg1.TCP_model.getClass() == arg1.UDP_model_in.getClass()
arg1.TCP_model.getClass() == arg1.TCP_model_out.getClass()
arg1.TCP_model.getClass() == arg1.UDP_model_out.getClass()
arg1.TCP_model.getClass() == arg1.TCP_model_outsrc.getClass()
arg1.TCP_model.getClass() == arg1.UDP_model_outsrc.getClass()
arg1.TCP_model.getClass() == orig(arg1.TCP_model.getClass())
arg1.TCP_model.getClass() == orig(arg1.UDP_model.getClass())
arg1.TCP_model.getClass() == orig(arg1.TCP_model_in.getClass())
arg1.TCP_model.getClass() == orig(arg1.UDP_model_in.getClass())
arg1.TCP_model.getClass() == orig(arg1.TCP_model_out.getClass())
arg1.TCP_model.getClass() == orig(arg1.UDP_model_out.getClass())
arg1.TCP_model.getClass() == orig(arg1.TCP_model_outsrc.getClass())
arg1.TCP_model.getClass() == orig(arg1.UDP_model_outsrc.getClass())
arg1.lenProb.getClass() == arg1.lenProb_out.getClass()
arg1.lenProb.getClass() == arg1.lenProb_in.getClass()
arg1.lenProb.getClass() == arg1.lenProb_outsrc.getClass()
arg1.lenProb.getClass() == orig(arg1.lenProb.getClass())
arg1.lenProb.getClass() == orig(arg1.lenProb_out.getClass())
arg1.lenProb.getClass() == orig(arg1.lenProb_in.getClass())
arg1.lenProb.getClass() == orig(arg1.lenProb_outsrc.getClass())
arg1.anomalyLengthThreshold[] elements == arg1.normal_length_perc
arg1.normal_length_perc in arg1.anomalyLengthThreshold[]
==========

TrainPacketHandler.byteToFloat(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
TrainPacketHandler.byteToFloat(byte[], int):::EXIT
arg0[] == orig(arg0[])
return != null
return[] elements >= 0.0
arg0[] elements > orig(arg1)
==========
TrainPacketHandler.byteToShort(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
TrainPacketHandler.byteToShort(byte[], int):::EXIT
arg0[] > return[] (lexically)
arg0[] >= return[] (lexically)
arg0[] == orig(arg0[])
return != null
return[] elements >= 0
arg0[] elements > orig(arg1)
return[] elements > orig(arg1)
==========
TrainPacketHandler.packetArrived(net.sourceforge.jpcap.net.Packet):::ENTER
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
arg0 != null
arg0._rcsid has only one value
arg0._rcsid != null
arg0._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this.host_ip.toString > arg0._rcsid.toString
==========
TrainPacketHandler.packetArrived(net.sourceforge.jpcap.net.Packet):::EXIT117
==========
TrainPacketHandler.packetArrived(net.sourceforge.jpcap.net.Packet):::EXIT
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.host_ip == orig(this.host_ip)
this.host_ip == orig(this.MM.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])

this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.trunc_length == orig(this.MM.trunc_length)
this.MM.port_trunc_length == orig(this.MM.port_trunc_length)
this.MM.min_sample == orig(this.MM.min_sample)
this.MM.min_sample == orig(this.MM.modification_time)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.stablesize == orig(this.MM.stablesize)
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.aggregation_done == orig(this.MM.aggregation_done)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.MM.lenProb_out[])
this.MM.lenProb[] == orig(this.MM.lenProb_in[])
this.MM.lenProb[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
arg0._rcsid == orig(arg0._rcsid)
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.MM.TCP_model[] has only one value, of length 1024

this.MM.TCP_model[].getClass() == [null, null, null,

this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
arg0._rcsid.toString == orig(arg0._rcsid.toString)
===
net.sourceforge.jpcap.capture.PacketCapture:::CLASS
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER != null
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == "jpcap"
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 0, 1, 2
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == 10
net.sourceforge.jpcap.capture.PacketCapture.nextInstance < net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
===
net.sourceforge.jpcap.capture.PacketCapture:::OBJECT
this.instanceNum one of 0, 1
this._rcsid.toString == "$Id: PacketCapture.java,v 1.14 2003/06/24 16:54:27 pcharles Exp $"
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString > this._rcsid.toString
net.sourceforge.jpcap.capture.PacketCapture.nextInstance >= this.instanceNum
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX > this.instanceNum
===
net.sourceforge.jpcap.capture.PacketCapture.PacketCapture():::EXIT
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
===
net.sourceforge.jpcap.capture.PacketCapture.capture(int):::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
count one of -1, 2300
===
net.sourceforge.jpcap.capture.PacketCapture.capture(int):::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
===
net.sourceforge.jpcap.capture.PacketCapture.close():::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
===
net.sourceforge.jpcap.capture.PacketCapture.close():::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
===
net.sourceforge.jpcap.capture.PacketCapture.openOffline(java.lang.String):::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
fileName has only one value
fileName.toString == "train-p2.dmp"
===
net.sourceforge.jpcap.capture.PacketCapture.openOffline(java.lang.String):::EXIT

net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
fileName.toString == "train-p2.dmp"
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
fileName.toString == orig(fileName.toString)
==========
net.sourceforge.jpcap.capture.PacketCapture.setFilter(java.lang.String, boolean):::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
filterExpression has only one value
filterExpression.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst
port 80))))"
optimize == true
==========
net.sourceforge.jpcap.capture.PacketCapture.setFilter(java.lang.String, boolean):::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 1, 2
this._rcsid has only one value
filterExpression.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst
port 80))))"
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
filterExpression.toString == orig(filterExpression.toString)
==========
net.sourceforge.jpcap.capture.PacketCapture.this():::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 0, 1
this.instanceNum == 0
this._rcsid == null
==========
net.sourceforge.jpcap.capture.PacketCapture.this():::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 0, 1
this.instanceNum == 0
this._rcsid has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
==========
net.sourceforge.jpcap.capture.PacketCaptureBase:::OBJECT
this.receivedCount == this.droppedCount
this.getClass() == net.sourceforge.jpcap.capture.PacketCapture.class
this.linkType one of 0, 1
this.receivedCount == 0
this._rcsid.toString == "$Id: PacketCaptureBase.java,v 1.4 2001/07/29 20:13:08 pcharles Exp $"
this.linkType >= this.receivedCount
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.PacketCaptureBase():::EXIT
this.linkType == this.receivedCount
this.linkType == this.droppedCount
this.linkType == 0
this._rcsid has only one value
==========

net.sourceforge.jpcap.capture.PacketCaptureBase.handlePacket(int, int, int, int, byte[]):::ENTER
length == caplen
this.linkType == 1
this._rcsid has only one value
this._rcsid != null
data != null
this.linkType < length
this.linkType < seconds
this.linkType < useconds
this.receivedCount < length
this.receivedCount < seconds
this.receivedCount < useconds
this.receivedCount in data[]
length < seconds
length != useconds
data[] elements < length
seconds > useconds
data[] elements < seconds
201419 * length - 11933014 * seconds - useconds + 1.317986122174792E16 == 0
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.handlePacket(int, int, int, int, byte[]):::EXIT
this.linkType == orig(this.linkType)
this.receivedCount == orig(this.receivedCount)
this.receivedCount == orig(this.droppedCount)
this._rcsid == orig(this._rcsid)
data[] == orig(data[])
this.linkType == 1
this._rcsid has only one value
this._rcsid != null
this.linkType < orig(length)
this.linkType < orig(seconds)
this.linkType < orig(useconds)
this.receivedCount in data[]
this.receivedCount < orig(length)
this.receivedCount < orig(seconds)
this.receivedCount < orig(useconds)
this._rcsid.toString == orig(this._rcsid.toString)
data[] elements < orig(length)
data[] elements < orig(seconds)
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.this():::ENTER
this.linkType == this.receivedCount
this.linkType == this.droppedCount
this.linkType == 0
this._rcsid == null
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.this():::EXIT
this.linkType == this.receivedCount
this.linkType == this.droppedCount
this.linkType == orig(this.linkType)
this.linkType == orig(this.receivedCount)
this.linkType == orig(this.droppedCount)
this.linkType == 0
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketDispatcher:::OBJECT
this.getClass() == net.sourceforge.jpcap.capture.PacketCapture.class
this._rcsid.toString == "$Id: PacketDispatcher.java,v 1.2 2001/06/18 05:02:26 pcharles Exp $"
==========
net.sourceforge.jpcap.capture.PacketDispatcher.PacketDispatcher():::EXIT
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketDispatcher.addPacketListener(net.sourceforge.jpcap.capture.PacketListener):::ENTER

this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketDispatcher.addPacketListener(net.sourceforge.jpcap.capture.PacketListener):::EXIT
this.rawListeners == orig(this.rawListeners)
this.objListeners == orig(this.objListeners)
this._rcsid == orig(this._rcsid)
this._rcsid has only one value
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchPacket(net.sourceforge.jpcap.net.Packet):::ENTER
this.rawListeners != null
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
packet != null
packet._rcsid has only one value
packet._rcsid != null
packet._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString > packet._rcsid.toString
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchPacket(net.sourceforge.jpcap.net.Packet):::EXIT
this.rawListeners == orig(this.rawListeners)
this.objListeners == orig(this.objListeners)
this._rcsid == orig(this._rcsid)
packet._rcsid == orig(packet._rcsid)
this.rawListeners != null
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
packet._rcsid has only one value
packet._rcsid != null
packet._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString > packet._rcsid.toString
this._rcsid.toString == orig(this._rcsid.toString)
packet._rcsid.toString == orig(packet._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchRawPacket(net.sourceforge.jpcap.net.RawPacket):::ENTER
this.rawListeners != null
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
rawPacket != null
rawPacket.timeval != null
rawPacket.timeval._rcsid has only one value
rawPacket.timeval._rcsid != null
rawPacket.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
rawPacket.bytes != null
rawPacket.droplen == 0
rawPacket._rcsid has only one value
rawPacket._rcsid != null
rawPacket._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString < rawPacket.timeval._rcsid.toString
this._rcsid.toString < rawPacket._rcsid.toString
rawPacket.timeval.seconds > rawPacket.timeval.microseconds
rawPacket.bytes[] elements < rawPacket.timeval.seconds
rawPacket.timeval.seconds > rawPacket.droplen
rawPacket.timeval.microseconds > rawPacket.droplen
rawPacket.timeval._rcsid.toString > rawPacket._rcsid.toString
rawPacket.droplen in rawPacket.bytes[]
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchRawPacket(net.sourceforge.jpcap.net.RawPacket):::EXIT
this.rawListeners == orig(this.rawListeners)
this.objListeners == orig(this.objListeners)

this._rcsid == orig(this._rcsid)
rawPacket.timeval == orig(rawPacket.timeval)
rawPacket.timeval.seconds == orig(rawPacket.timeval.seconds)
rawPacket.timeval.microseconds == orig(rawPacket.timeval.microseconds)
rawPacket.timeval._rcsid == orig(rawPacket.timeval._rcsid)
rawPacket.bytes == orig(rawPacket.bytes)
rawPacket.bytes[] == orig(rawPacket.bytes[])
rawPacket.droplen == orig(rawPacket.droplen)
rawPacket._rcsid == orig(rawPacket._rcsid)
this.rawListeners != null
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
rawPacket.timeval != null
rawPacket.timeval._rcsid has only one value
rawPacket.timeval._rcsid != null
rawPacket.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
rawPacket.bytes != null
rawPacket.droplen == 0
rawPacket._rcsid has only one value
rawPacket._rcsid != null
rawPacket._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString < rawPacket.timeval._rcsid.toString
this._rcsid.toString < rawPacket._rcsid.toString
this._rcsid.toString == orig(this._rcsid.toString)
rawPacket.timeval.seconds > rawPacket.timeval.microseconds
rawPacket.bytes[] elements < rawPacket.timeval.seconds
rawPacket.timeval.seconds > rawPacket.droplen
rawPacket.timeval.microseconds > rawPacket.droplen
rawPacket.timeval._rcsid.toString > rawPacket._rcsid.toString
rawPacket.timeval._rcsid.toString == orig(rawPacket.timeval._rcsid.toString)
rawPacket.droplen in rawPacket.bytes[]
rawPacket._rcsid.toString == orig(rawPacket._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketDispatcher.this():::ENTER
this.rawListeners == this.objListeners
this.rawListeners == null
this._rcsid == null
==========
net.sourceforge.jpcap.capture.PacketDispatcher.this():::EXIT
this._rcsid has only one value
==========
net.sourceforge.jpcap.net.EthernetPacket:::OBJECT
this._ethOffset == this._ethernetHeaderLength
this._ethernetHeaderBytes == this._ethernetDataBytes
this._sourceHwAddress == this._destinationHwAddress
this._ethOffset one of 0, 14
this._ethernetHeaderBytes == null
this._sourceHwAddress == null
this._etherProtocol == 0
this._etherProtocolSet == false
this._rcsid.toString == "$Id: EthernetPacket.java,v 1.17 2003/06/24 23:09:49 pcharles Exp $"
this._etherProtocol in this._bytes[]
this._ethOffset >= this._etherProtocol
==========
net.sourceforge.jpcap.net.EthernetPacket.EthernetPacket(int, byte[]):::ENTER
lLen == 14
bytes != null
==========
net.sourceforge.jpcap.net.EthernetPacket.EthernetPacket(int, byte[]):::EXIT
this._bytes == orig(bytes)
this._bytes[] == bytes[]
this._ethOffset == orig(lLen)

bytes[] == orig(bytes[])
this._bytes != null
this._ethOffset == 14
this._rcsid has only one value
this._rcsid != null
this._etherProtocol in bytes[]
===
net.sourceforge.jpcap.net.EthernetPacket.this():::ENTER
this._bytes == this._ethernetHeaderBytes
this._bytes == this._ethernetDataBytes
this._ethOffset == this._etherProtocol
this._sourceHwAddress == this._rcsid
this._bytes == null
this._ethOffset == 0
===
net.sourceforge.jpcap.net.EthernetPacket.this():::EXIT
this._bytes == this._ethernetHeaderBytes
this._bytes == this._ethernetDataBytes
this._bytes == orig(this._bytes)
this._bytes == orig(this._ethernetHeaderBytes)
this._bytes == orig(this._ethernetDataBytes)
this._ethOffset == this._etherProtocol
this._ethOffset == orig(this._ethOffset)
this._ethOffset == orig(this._ethernetHeaderLength)
this._ethOffset == orig(this._etherProtocol)
this._sourceHwAddress == orig(this._sourceHwAddress)
this._sourceHwAddress == orig(this._destinationHwAddress)
this._sourceHwAddress == orig(this._rcsid)
this._etherProtocolSet == orig(this._etherProtocolSet)
this._bytes == null
this._ethOffset == 0
this._rcsid has only one value
this._rcsid != null
===
net.sourceforge.jpcap.net.IPAddress:::CLASS
===
net.sourceforge.jpcap.net.IPAddress.extract(int, byte[]):::ENTER
offset one of 26, 30
bytes != null
net.sourceforge.jpcap.net.IPAddress.WIDTH < offset
===
net.sourceforge.jpcap.net.IPAddress.extract(int, byte[]):::EXIT
bytes[] == orig(bytes[])
return != null
===
net.sourceforge.jpcap.net.IPAddress.toString(int):::ENTER
address != 0
net.sourceforge.jpcap.net.IPAddress.WIDTH != address
===
net.sourceforge.jpcap.net.IPAddress.toString(int):::EXIT
return != null
===
net.sourceforge.jpcap.net.IPPacket:::OBJECT
this._version == this._typeOfService
this._version == this._id
this._version == this._fragmentFlags
this._version == this._fragmentOffset
this._version == this._timeToLive
this._version == this._ipChecksum
this._version == this._sourceAddressAsLong
this._version == this._destinationAddressAsLong
this._versionSet == this._typeOfServiceSet
this._versionSet == this._idSet

this._versionSet == this._fragmentFlagsSet
this._versionSet == this._fragmentOffsetSet
this._versionSet == this._timeToLiveSet
this._versionSet == this._ipChecksumSet
this._versionSet == this._sourceAddressAsLongSet
this._versionSet == this._destinationAddressAsLongSet
this._versionSet == this._isValidChecksum
this._versionSet == this._isValidChecksumSet
this._sourceAddressBytes == this._destinationAddressBytes
this._sourceAddressBytes == this._ipHeaderBytes
this._sourceAddressBytes == this._ipDataBytes
this._ipOffset one of 0, 34
this._version == 0
this._versionSet == false
this._ipHeaderLength one of 0, 20
this._length >= 0
this._ipProtocol one of 0, 6
this._sourceAddressBytes == null
this._destinationAddress.toString == "128.59.16.101"
this._rcsid.toString == "$Id: IPPacket.java,v 1.20 2003/07/02 18:41:10 pcharles Exp $"
this._ipOffset >= this._version
this._ipOffset >= this._ipHeaderLength
this._ipOffset >= this._ipProtocol
this._version <= this._ipHeaderLength
this._version <= this._length
this._version <= this._ipProtocol
this._ipHeaderLength >= this._ipProtocol
this._length >= this._ipProtocol
this._sourceAddress.toString != this._destinationAddress.toString
this._sourceAddress.toString > this._rcsid.toString
this._destinationAddress.toString > this._rcsid.toString
==========
net.sourceforge.jpcap.net.IPPacket.IPPacket(int, byte[]):::ENTER
lLen == 14
bytes != null
==========
net.sourceforge.jpcap.net.IPPacket.IPPacket(int, byte[]):::EXIT
this._version == this._length
this._version == this._ipProtocol
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
bytes[] == orig(bytes[])
this._ipOffset == 34
this._ipHeaderLength == 20
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._version in bytes[]
==========
net.sourceforge.jpcap.net.IPPacket.getDestinationAddress():::ENTER
this._lengthSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length

this._length > this._ipProtocol
==========
net.sourceforge.jpcap.net.IPPacket.getDestinationAddress():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._lengthSet == orig(this._ipProtocolSet)
this._ipProtocol == orig(this._ipProtocol)
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._destinationAddress == return
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return != null
return.toString == "128.59.16.101"
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
this._destinationAddress.toString == return.toString
this._rcsid.toString < return.toString
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getIPHeaderLength():::ENTER
this._version == this._ipProtocol
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null

this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
==========
net.sourceforge.jpcap.net.IPPacket.getIPHeaderLength():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == this._ipProtocol
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == return
this._length == orig(this._length)
this._lengthSet == orig(this._lengthSet)
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._ipHeaderLength)
this._ipOffset == 34
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return == 20
this._ipOffset < this._length
this._version < this._length
this._length > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getIPProtocol():::ENTER
this._version == this._ipProtocol
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null

this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
==========
net.sourceforge.jpcap.net.IPPacket.getIPProtocol():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._ipProtocol == return
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return == 6
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getIpHeaderLength():::ENTER
this._version == this._ipProtocol
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true

this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
===
net.sourceforge.jpcap.net.IPPacket.getIpHeaderLength():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == this._ipProtocol
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == return
this._length == orig(this._length)
this._lengthSet == orig(this._lengthSet)
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._ipHeaderLength)
this._ipOffset == 34
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return == 20
this._ipOffset < this._length
this._version < this._length
this._length > return
this._rcsid.toString == orig(this._rcsid.toString)
===
net.sourceforge.jpcap.net.IPPacket.getLength():::ENTER
this._version == this._length
this._version == this._ipProtocol
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34

this._ipHeaderLength == 20
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.IPPacket.getLength():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == this._ipProtocol
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._length)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._lengthSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == return
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < return
this._version < return
this._ipHeaderLength < return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getProtocol():::ENTER
this._version == this._ipProtocol
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value

this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
==========
net.sourceforge.jpcap.net.IPPacket.getProtocol():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._ipProtocol == return
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return == 6
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getSourceAddress():::ENTER
this._lengthSet == this._ipProtocolSet
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._sourceAddress == null
this._destinationAddress != null

this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
==========
net.sourceforge.jpcap.net.IPPacket.getSourceAddress():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._lengthSet == orig(this._ipProtocolSet)
this._ipProtocol == orig(this._ipProtocol)
this._sourceAddress == return
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._destinationAddress == orig(this._destinationAddress)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._destinationAddress != null
this._rcsid has only one value
this._rcsid != null
return != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
this._sourceAddress.toString == return.toString
this._destinationAddress.toString != return.toString
this._destinationAddress.toString == orig(this._destinationAddress.toString)
this._rcsid.toString < return.toString
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.this():::ENTER
this._ipOffset == this._version
this._ipOffset == this._ipHeaderLength

this._ipOffset == this._typeOfService
this._ipOffset == this._length
this._ipOffset == this._id
this._ipOffset == this._fragmentFlags
this._ipOffset == this._fragmentOffset
this._ipOffset == this._timeToLive
this._ipOffset == this._ipProtocol
this._ipOffset == this._ipChecksum
this._ipOffset == this._sourceAddressAsLong
this._ipOffset == this._destinationAddressAsLong
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._sourceAddress == this._rcsid
this._ipOffset == 0
this._sourceAddress == null
==========
net.sourceforge.jpcap.net.IPPacket.this():::EXIT
this._ipOffset == this._version
this._ipOffset == this._ipHeaderLength
this._ipOffset == this._typeOfService
this._ipOffset == this._length
this._ipOffset == this._id
this._ipOffset == this._fragmentFlags
this._ipOffset == this._fragmentOffset
this._ipOffset == this._timeToLive
this._ipOffset == this._ipProtocol
this._ipOffset == this._ipChecksum
this._ipOffset == this._sourceAddressAsLong
this._ipOffset == this._destinationAddressAsLong
this._ipOffset == orig(this._ipOffset)
this._ipOffset == orig(this._version)
this._ipOffset == orig(this._ipHeaderLength)
this._ipOffset == orig(this._typeOfService)
this._ipOffset == orig(this._length)
this._ipOffset == orig(this._id)
this._ipOffset == orig(this._fragmentFlags)
this._ipOffset == orig(this._fragmentOffset)
this._ipOffset == orig(this._timeToLive)
this._ipOffset == orig(this._ipProtocol)
this._ipOffset == orig(this._ipChecksum)
this._ipOffset == orig(this._sourceAddressAsLong)
this._ipOffset == orig(this._destinationAddressAsLong)
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._lengthSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddress == orig(this._rcsid)
this._sourceAddressBytes == orig(this._sourceAddressBytes)

this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._ipOffset == 0
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.IPProtocol:::CLASS
net.sourceforge.jpcap.net.IPProtocol.messages has only one value
net.sourceforge.jpcap.net.IPProtocol.messages != null
==========
net.sourceforge.jpcap.net.IPProtocol.extractProtocol(int, byte[]):::ENTER
lLen == 14
packetBytes != null
==========
net.sourceforge.jpcap.net.IPProtocol.extractProtocol(int, byte[]):::EXIT
net.sourceforge.jpcap.net.IPProtocol.messages == orig(net.sourceforge.jpcap.net.IPProtocol.messages)
packetBytes[] == orig(packetBytes[])
return == 6
return in packetBytes[]
==========
net.sourceforge.jpcap.net.LinkLayer:::CLASS
net.sourceforge.jpcap.net.LinkLayer.descriptions has only one value
net.sourceforge.jpcap.net.LinkLayer.descriptions != null
==========
net.sourceforge.jpcap.net.LinkLayer.getLinkLayerLength(int):::ENTER
layerType == 1
==========
net.sourceforge.jpcap.net.LinkLayer.getLinkLayerLength(int):::EXIT59
==========
net.sourceforge.jpcap.net.LinkLayer.getLinkLayerLength(int):::EXIT
net.sourceforge.jpcap.net.LinkLayer.descriptions == orig(net.sourceforge.jpcap.net.LinkLayer.descriptions)
return == 14
==========
net.sourceforge.jpcap.net.LinkLayer.getProtoOffset(int):::ENTER
layerType == 1
==========
net.sourceforge.jpcap.net.LinkLayer.getProtoOffset(int):::EXIT100
==========
net.sourceforge.jpcap.net.LinkLayer.getProtoOffset(int):::EXIT
net.sourceforge.jpcap.net.LinkLayer.descriptions == orig(net.sourceforge.jpcap.net.LinkLayer.descriptions)
return == 12
==========
net.sourceforge.jpcap.net.Packet:::OBJECT
this._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
==========
net.sourceforge.jpcap.net.Packet.Packet():::EXIT
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.Packet.this():::ENTER
this._rcsid == null
==========
net.sourceforge.jpcap.net.Packet.this():::EXIT
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.PacketEncoding:::CLASS
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY has only one value
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY != null
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] == []
==========

net.sourceforge.jpcap.net.PacketEncoding.extractData(int, int, byte[], int):::ENTER
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] < bytes[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] <= bytes[] (lexically)
offset == 34
headerLen one of 20, 32
bytes != null
offset > headerLen
offset < dataLength
headerLen < dataLength
bytes[] elements < dataLength
==========
net.sourceforge.jpcap.net.PacketEncoding.extractData(int, int, byte[], int):::EXIT108
==========
net.sourceforge.jpcap.net.PacketEncoding.extractData(int, int, byte[], int):::EXIT
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY == orig(net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_A
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] < bytes[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] <= bytes[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] < return[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] <= return[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] == orig(net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE
bytes[] < return[] (lexically)
bytes[] <= return[] (lexically)
bytes[] == orig(bytes[])
return != null
bytes[] elements < orig(dataLength)
return[] elements < orig(dataLength)
==========
net.sourceforge.jpcap.net.PacketFactory:::CLASS
net.sourceforge.jpcap.net.PacketFactory.lLen one of 0, 14
==========
net.sourceforge.jpcap.net.PacketFactory.dataToPacket(int, byte[]):::ENTER
linkType == 1
bytes != null
net.sourceforge.jpcap.net.PacketFactory.lLen != linkType
==========
net.sourceforge.jpcap.net.PacketFactory.dataToPacket(int, byte[]):::EXIT58
==========
net.sourceforge.jpcap.net.PacketFactory.dataToPacket(int, byte[]):::EXIT
bytes[] == orig(bytes[])
net.sourceforge.jpcap.net.PacketFactory.lLen == 14
return != null
return._rcsid has only one value
return._rcsid != null
return._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
net.sourceforge.jpcap.net.PacketFactory.lLen >= orig(net.sourceforge.jpcap.net.PacketFactory.lLen)
==========
net.sourceforge.jpcap.net.RawPacket:::OBJECT
this.timeval._rcsid has only one value
this.timeval._rcsid != null
this.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.droplen == 0
this._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this.timeval.seconds > this.timeval.microseconds
this.bytes[] elements < this.timeval.seconds
this.timeval.seconds > this.droplen
this.timeval.microseconds > this.droplen
this.timeval._rcsid.toString > this._rcsid.toString
this.droplen in this.bytes[]
==========
net.sourceforge.jpcap.net.RawPacket.RawPacket(net.sourceforge.jpcap.util.Timeval, byte[], int):::ENTER
timeval != null
timeval._rcsid has only one value
timeval._rcsid != null

timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
bytes != null
droplen == 0
timeval.seconds > timeval.microseconds
bytes[] elements < timeval.seconds
timeval.seconds > droplen
timeval.microseconds > droplen
droplen in bytes[]
===
net.sourceforge.jpcap.net.RawPacket.RawPacket(net.sourceforge.jpcap.util.Timeval, byte[], int):::EXIT
this.timeval == orig(timeval)
this.timeval.seconds == timeval.seconds
this.timeval.microseconds == timeval.microseconds
this.timeval._rcsid == timeval._rcsid
this.bytes == orig(bytes)
this.bytes[] == bytes[]
this.droplen == orig(droplen)
timeval.seconds == orig(timeval.seconds)
timeval.microseconds == orig(timeval.microseconds)
timeval._rcsid == orig(timeval._rcsid)
bytes[] == orig(bytes[])
this.timeval != null
this.bytes != null
this._rcsid has only one value
this._rcsid != null
timeval._rcsid has only one value
timeval._rcsid != null
timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.timeval._rcsid.toString == timeval._rcsid.toString
this.droplen < timeval.seconds
this.droplen < timeval.microseconds
this.droplen in bytes[]
this._rcsid.toString < timeval._rcsid.toString
timeval.seconds > timeval.microseconds
bytes[] elements < timeval.seconds
timeval._rcsid.toString == orig(timeval._rcsid.toString)
===
net.sourceforge.jpcap.net.RawPacket.this():::ENTER
this.timeval == null
this.bytes == null
this._rcsid == null
===
net.sourceforge.jpcap.net.RawPacket.this():::EXIT
this.timeval == orig(this.timeval)
this.bytes == orig(this.bytes)
this.droplen == orig(this.droplen)
this.timeval == null
this.bytes == null
this._rcsid has only one value
this._rcsid != null
===
net.sourceforge.jpcap.net.TCPPacket:::OBJECT
this._sequenceNumber == this._acknowledgmentNumber
this._sequenceNumber == this._windowSize
this._sequenceNumber == this._tcpChecksum
this._sequenceNumber == this._urgentPointer
this._sequenceNumber == this._allFlags
this._sequenceNumberSet == this._acknowledgmentNumberSet
this._sequenceNumberSet == this._windowSizeSet
this._sequenceNumberSet == this._tcpChecksumSet
this._sequenceNumberSet == this._urgentPointerSet
this._sequenceNumberSet == this._allFlagsSet
this._sequenceNumberSet == this._isUrg

this._sequenceNumberSet == this._isUrgSet
this._sequenceNumberSet == this._isAck
this._sequenceNumberSet == this._isAckSet
this._sequenceNumberSet == this._isPsh
this._sequenceNumberSet == this._isPshSet
this._sequenceNumberSet == this._isRst
this._sequenceNumberSet == this._isRstSet
this._sequenceNumberSet == this._isSyn
this._sequenceNumberSet == this._isSynSet
this._sequenceNumberSet == this._isFin
this._sequenceNumberSet == this._isFinSet
this._sourcePort >= 0
this._destinationPort one of 0, 80
this._sequenceNumber == 0
this._sequenceNumberSet == false
this._tcpHeaderLength one of 0, 20, 32
this._payloadDataLength >= 0
this._tcpHeaderBytes == null
this._rcsid.toString == "$Id: TCPPacket.java,v 1.20 2003/07/02 18:43:05 pcharles Exp $"
this._sourcePort >= this._sequenceNumber
this._destinationPort >= this._sequenceNumber
this._destinationPort <= this._payloadDataLength
this._sequenceNumber <= this._tcpHeaderLength
this._sequenceNumber <= this._payloadDataLength
this._tcpDataBytes[] elements > this._sequenceNumber
this._tcpHeaderLength <= this._payloadDataLength
this._tcpDataBytes[] elements < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.TCPPacket(int, byte[]):::ENTER
lLen == 14
bytes != null
==========
net.sourceforge.jpcap.net.TCPPacket.TCPPacket(int, byte[]):::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
bytes[] == orig(bytes[])
this._sourcePort == 0
this._sourcePortSet == false

this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._sourcePort in bytes[]
this._tcpHeaderLength < this._payloadDataLength
this._tcpHeaderLength > orig(lLen)
bytes[] elements < this._payloadDataLength
this._payloadDataLength > orig(lLen)
==========
net.sourceforge.jpcap.net.TCPPacket.getData():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getData():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)

this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._tcpDataBytes == return
this._tcpDataBytes[] == return[]
this._rcsid == orig(this._rcsid)
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
return != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
return[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
return[] elements < this._payloadDataLength
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getDestinationPort():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber

this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpDataBytes[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getDestinationPort():::EXIT
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst

this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._destinationPort == return
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpDataBytes == orig(this._tcpDataBytes)
this._tcpDataBytes[] == orig(this._tcpDataBytes[])
this._rcsid == orig(this._rcsid)
this._sourcePort == 0
this._sourcePortSet == false
this._destinationPortSet == true
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
return == 80
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpDataBytes[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
this._tcpHeaderLength < return
this._payloadDataLength > return
return in this._tcpDataBytes[]
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getPayloadDataLength():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet

this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getPayloadDataLength():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)

this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == return
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._payloadDataLength)
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < return
this._tcpHeaderLength < return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getSourcePort():::ENTER
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePort == 0
this._sourcePortSet == false
this._destinationPort == 80
this._destinationPortSet == true
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null

this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpDataBytes[] elements > this._sourcePort
this._destinationPort > this._tcpHeaderLength
this._destinationPort < this._payloadDataLength
this._destinationPort in this._tcpDataBytes[]
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getSourcePort():::EXIT
this._sourcePort == return
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == orig(this._destinationPortSet)
this._destinationPort == orig(this._destinationPort)
this._sequenceNumber == orig(this._sourcePort)
this._sequenceNumber == orig(this._sequenceNumber)
this._sequenceNumber == orig(this._acknowledgmentNumber)
this._sequenceNumber == orig(this._windowSize)
this._sequenceNumber == orig(this._tcpChecksum)
this._sequenceNumber == orig(this._urgentPointer)
this._sequenceNumber == orig(this._allFlags)
this._sequenceNumberSet == orig(this._sourcePortSet)
this._sequenceNumberSet == orig(this._sequenceNumberSet)
this._sequenceNumberSet == orig(this._acknowledgmentNumberSet)
this._sequenceNumberSet == orig(this._windowSizeSet)
this._sequenceNumberSet == orig(this._tcpChecksumSet)
this._sequenceNumberSet == orig(this._urgentPointerSet)
this._sequenceNumberSet == orig(this._allFlagsSet)
this._sequenceNumberSet == orig(this._isUrg)
this._sequenceNumberSet == orig(this._isUrgSet)
this._sequenceNumberSet == orig(this._isAck)
this._sequenceNumberSet == orig(this._isAckSet)
this._sequenceNumberSet == orig(this._isPsh)
this._sequenceNumberSet == orig(this._isPshSet)
this._sequenceNumberSet == orig(this._isRst)
this._sequenceNumberSet == orig(this._isRstSet)
this._sequenceNumberSet == orig(this._isSyn)
this._sequenceNumberSet == orig(this._isSynSet)
this._sequenceNumberSet == orig(this._isFin)
this._sequenceNumberSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpDataBytes == orig(this._tcpDataBytes)
this._tcpDataBytes[] == orig(this._tcpDataBytes[])
this._rcsid == orig(this._rcsid)
this._sourcePortSet == true
this._destinationPort == 80
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
this._destinationPort > this._tcpHeaderLength
this._destinationPort < this._payloadDataLength
this._destinationPort in this._tcpDataBytes[]
this._destinationPort < return
this._sequenceNumber < this._tcpHeaderLength
this._sequenceNumber < this._payloadDataLength
this._sequenceNumber < return
this._tcpHeaderLength < this._payloadDataLength
this._tcpHeaderLength < return
this._payloadDataLength < return

this._tcpDataBytes[] elements < return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPData():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPData():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet

this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._tcpDataBytes == return
this._tcpDataBytes[] == return[]
this._rcsid == orig(this._rcsid)
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
return != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
return[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
return[] elements < this._payloadDataLength
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPHeaderLength():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet

this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPHeaderLength():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)

this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == return
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._tcpHeaderLength)
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid has only one value
this._rcsid != null
return one of 20, 32
this._sourcePort < this._payloadDataLength
this._sourcePort < return
this._payloadDataLength > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getTcpHeaderLength():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes

this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getTcpHeaderLength():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)

this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == return
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._tcpHeaderLength)
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid has only one value
this._rcsid != null
return one of 20, 32
this._sourcePort < this._payloadDataLength
this._sourcePort < return
this._payloadDataLength > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.this():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._tcpHeaderLength
this._sourcePort == this._payloadDataLength
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid == null
==========
net.sourceforge.jpcap.net.TCPPacket.this():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._tcpHeaderLength
this._sourcePort == this._payloadDataLength
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)

this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._tcpHeaderLength)
this._sourcePort == orig(this._payloadDataLength)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.util.ArrayHelper:::CLASS
net.sourceforge.jpcap.util.ArrayHelper._rcsid has only one value
net.sourceforge.jpcap.util.ArrayHelper._rcsid != null
net.sourceforge.jpcap.util.ArrayHelper._rcsid.toString == "$Id: ArrayHelper.java,v 1.8 2002/09/04 19:36:47 pcharles Exp
$"
==========
net.sourceforge.jpcap.util.ArrayHelper.extractInteger(byte[], int, int):::ENTER

bytes != null
cnt one of 1, 2, 4
pos > cnt
==========
net.sourceforge.jpcap.util.ArrayHelper.extractInteger(byte[], int, int):::EXIT
net.sourceforge.jpcap.util.ArrayHelper._rcsid == orig(net.sourceforge.jpcap.util.ArrayHelper._rcsid)
bytes[] == orig(bytes[])
return != 0
net.sourceforge.jpcap.util.ArrayHelper._rcsid.toString == orig(net.sourceforge.jpcap.util.ArrayHelper._rcsid.toString)
return != orig(pos)
return != orig(cnt)
==========
net.sourceforge.jpcap.util.Timeval:::OBJECT
this.seconds >= 0
this.microseconds >= 0
this._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.seconds >= this.microseconds
==========
net.sourceforge.jpcap.util.Timeval.Timeval(long, int):::ENTER
seconds > microseconds
==========
net.sourceforge.jpcap.util.Timeval.Timeval(long, int):::EXIT
this.seconds == orig(seconds)
this.microseconds == orig(microseconds)
this._rcsid has only one value
this._rcsid != null
this.seconds > this.microseconds
==========
net.sourceforge.jpcap.util.Timeval.this():::ENTER
this.seconds == this.microseconds
this.seconds == 0
this._rcsid == null
==========
net.sourceforge.jpcap.util.Timeval.this():::EXIT
this.seconds == this.microseconds
this.seconds == orig(this.seconds)
this.seconds == orig(this.microseconds)
this.seconds == 0
this._rcsid has only one value
this._rcsid != null
==========
pairs:::OBJECT
this.value >= 0.0
this.index >= 0
==========
pairs.pairs(float, int):::ENTER
arg0 >= 0.0
arg1 >= 0
==========
pairs.pairs(float, int):::EXIT
this.value == orig(arg0)
this.index == orig(arg1)
==========
Models:::OBJECT
this.UDP_model[] == this.TCP_model_in[]
this.UDP_model[] == this.UDP_model_in[]
this.UDP_model[] == this.TCP_model_out[]
this.UDP_model[] == this.UDP_model_out[]
this.UDP_model[] == this.TCP_model_outsrc[]
this.UDP_model[] == this.UDP_model_outsrc[]
this.UDP_model[].getClass() == this.UDP_lengthModel[].getClass()
this.UDP_lengthModel[] == this.TCP_lengthModel_in[]
this.UDP_lengthModel[] == this.UDP_lengthModel_in[]

this.UDP_lengthModel[] == this.TCP_lengthModel_out[]
this.UDP_lengthModel[] == this.UDP_lengthModel_out[]
this.UDP_lengthModel[] == this.TCP_lengthModel_outsrc[]
this.UDP_lengthModel[] == this.UDP_lengthModel_outsrc[]
this.min_sample == this.modification_time
this.lenProb_out[] == this.lenProb_in[]
this.lenProb_out[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold_out[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold_out[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength[] == this.UDPNormalLength_out[]
this.UDPNormalLength[] == this.UDPNormalLength_in[]
this.UDPNormalLength[] == this.UDPNormalLength_outsrc[]
this has only one value
this.TCP_model.getClass() == java.util.HashMap[].class
this.TCP_model[].getClass() one of [null, null, null,

null, null,

null, null,

null, null, null,

0.0, 0.0, 0.0,

0.0, 0.0, 0.0,

this.TCP_lengthModel.getClass() == this.UDP_lengthModel_in.getClass()
this.TCP_lengthModel.getClass() == this.TCP_lengthModel_out.getClass()
this.TCP_lengthModel.getClass() == this.UDP_lengthModel_out.getClass()
this.TCP_lengthModel.getClass() == this.TCP_lengthModel_outsrc.getClass()
this.TCP_lengthModel.getClass() == this.UDP_lengthModel_outsrc.getClass()
this.lenProb.getClass() == this.lenProb_out.getClass()
this.lenProb.getClass() == this.lenProb_in.getClass()
this.lenProb.getClass() == this.lenProb_outsrc.getClass()
this.normal_length_perc in this.anomalyLengthThreshold[]
this.anomalyLengthThreshold_out[] elements == this.normal_length_perc
this.normal_length_perc in this.anomalyLengthThreshold_out[]
this.UDPNormalLength.getClass() == this.UDPNormalLength_out.getClass()
this.UDPNormalLength.getClass() == this.UDPNormalLength_in.getClass()
this.UDPNormalLength.getClass() == this.UDPNormalLength_outsrc.getClass()
==========
Models.Models(int, java.util.HashMap, int, int, float, java.lang.String, int, double, float):::ENTER
arg0 == -1
arg1 has only one value
arg2 == 0
arg3 == 1000
arg4 == 0.0
arg5 has only one value
arg5.toString == "128.59.16.101"
arg6 == 1
arg7 == 0.05
arg8 == 0.0010
==========
Models.Models(int, java.util.HashMap, int, int, float, java.lang.String, int, double, float):::EXIT
this.TCP_model[] == this.UDP_model[]
this.TCP_model[] == this.TCP_model_in[]
this.TCP_model[] == this.UDP_model_in[]
this.TCP_model[] == this.TCP_model_out[]
this.TCP_model[] == this.UDP_model_out[]
this.TCP_model[] == this.TCP_model_outsrc[]
this.TCP_model[] == this.UDP_model_outsrc[]
this.TCP_lengthModel == this.UDP_lengthModel
this.TCP_lengthModel == this.TCP_lengthModel_in
this.TCP_lengthModel == this.UDP_lengthModel_in
this.TCP_lengthModel == this.TCP_lengthModel_out
this.TCP_lengthModel == this.UDP_lengthModel_out
this.TCP_lengthModel == this.TCP_lengthModel_outsrc
this.TCP_lengthModel == this.UDP_lengthModel_outsrc
this.trunc_length == orig(arg0)
this.port_trunc_length == orig(arg1)
this.min_sample == orig(arg2)
this.max_sample == orig(arg3)
this.sample_rate == orig(arg4)
this.host_ip == orig(arg5)
this.stablesize == orig(arg6)
this.stabledis == orig(arg7)
this.smooth_factor == orig(arg8)
this.lenProb[] == this.lenProb_out[]
this.lenProb[] == this.lenProb_in[]
this.lenProb[] == this.lenProb_outsrc[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_out[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_in[]
this.anomalyLengthThreshold[] == this.anomalyLengthThreshold_outsrc[]
this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model has only one value
this.TCP_model[] contains only nulls and has only one value, of length 1024
this.TCP_model[] elements == null

this.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.UDPNormalLength == this.UDPNormalLength_out
this.UDPNormalLength == this.UDPNormalLength_in
this.UDPNormalLength == this.UDPNormalLength_outsrc
this.TCP_model has only one value
this.TCP_model[] contains only nulls and has only one value, of length 1024
this.TCP_model[] elements == null
this.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.min_sample == orig(this.modification_time)
this.max_sample == orig(this.max_sample)
this.sample_rate == orig(this.sample_rate)
this.host_ip == orig(this.host_ip)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.smooth_factor == orig(this.smooth_factor)
this.aggregation_done == orig(this.aggregation_done)
this.lenProb_out[] == orig(this.lenProb[])
this.lenProb_out[] == orig(this.lenProb_out[])
this.lenProb_out[] == orig(this.lenProb_in[])
this.lenProb_out[] == orig(this.lenProb_outsrc[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_out[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_in[])
this.anomalyLengthThreshold_out[] == orig(this.anomalyLengthThreshold_outsrc[])
this.normal_length_perc == orig(this.normal_length_perc)
this.TCP_model has only one value
this.TCP_model[] has only one value, of length 1024
this.TCP_model[].getClass() == [null, null, null,

this.UDP_model has only one value
this.TCP_model_in has only one value
this.UDP_model_in has only one value
this.TCP_model_out has only one value
this.UDP_model_out has only one value
this.TCP_model_outsrc has only one value
this.UDP_model_outsrc has only one value
this.TCP_lengthModel has only one value
this.UDP_lengthModel has only one value
this.TCP_lengthModel_in has only one value
this.UDP_lengthModel_in has only one value
this.TCP_lengthModel_out has only one value
this.UDP_lengthModel_out has only one value
this.TCP_lengthModel_outsrc has only one value
this.UDP_lengthModel_outsrc has only one value
this.lenProb has only one value
this.lenProb[] has only one value, of length 1024
this.anomalyLengthThreshold has only one value
this.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.UDPNormalLength_outsrc has only one value
arg0.toString == "model.dat"
this.TCP_model.getClass() == this.UDPNormalLength_out.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_in.getClass()
this.TCP_model.getClass() == this.UDPNormalLength_outsrc.getClass()
this.TCP_model.getClass() == orig(this.TCP_model.getClass())
this.TCP_model.getClass() == orig(this.UDP_model.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_in.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_in.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_out.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_out.getClass())
this.TCP_model.getClass() == orig(this.TCP_model_outsrc.getClass())
this.TCP_model.getClass() == orig(this.UDP_model_outsrc.getClass())
this.host_ip.toString == orig(this.host_ip.toString)
this.lenProb.getClass() == orig(this.lenProb.getClass())
this.lenProb.getClass() == orig(this.lenProb_out.getClass())
this.lenProb.getClass() == orig(this.lenProb_in.getClass())
this.lenProb.getClass() == orig(this.lenProb_outsrc.getClass())
arg0.toString == orig(arg0.toString)
==========
OneCentroid:::OBJECT
this.num == this.start
this.num == this.end
this.num == this.count
this.num == this.resettimes
this.x[] == this.xsqr[]
this.x[] == this.avg[]
this.x[] == this.std[]
this.samples[] == this.sampleDistance[]
this.num == 0
this.x != null
this.x[] == [0.0, 0.0, 0.0,

arg2 != null
arg2[] elements >= 0.0010
==========
OneCentroid.MahaDis(float[], float[], float[]):::EXIT
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
arg0[] elements >= 0.0
arg1[] elements >= 0.0
arg2[] elements >= 0.0010
arg0[] elements < return
arg1[] elements < return
arg2[] elements < return
==========
OneCentroid.OneCentroid():::EXIT
==========
OneCentroid.indexSort(float[]):::ENTER
arg0 != null
arg0[] elements >= 0.0
==========
OneCentroid.indexSort(float[]):::EXIT
arg0[] == orig(arg0[])
arg0[] elements >= 0.0
return != null
return[] elements >= 0
==========
OneModel:::OBJECT
this.centroids[] == this.b_packets[]
this.centroids[] == this.b_nums[]
this.start == this.end
this.start == this.min_sample
this.start == this.max_sample
this.start == this.totalPacketNum
this.start == this.reduceN
this has only one value
this.centroids has only one value
this.centroids[] == []
this.centroids[].getClass() == []
this.NMax == 10
this.start == 0
this.sample_rate == 0.0
this.b_packets has only one value
this.b_nums has only one value
this.batch == false
==========
OneModel.OneModel():::EXIT
==========
Payload:::OBJECT
this.TEST_PACKET_COUNT == this.MM.trunc_length
this.TEST_PACKET_COUNT == this.testPH.trunc_length
this.MM == this.testPH.MM
this.MM.TCP_model == this.testPH.MM.TCP_model
this.MM.TCP_model[].getClass() == this.testPH.MM.TCP_model[].getClass()
this.MM.UDP_model == this.testPH.MM.UDP_model
this.MM.UDP_model[] == this.MM.TCP_model_in[]
this.MM.UDP_model[] == this.MM.UDP_model_in[]
this.MM.UDP_model[] == this.MM.TCP_model_out[]
this.MM.UDP_model[] == this.MM.UDP_model_out[]
this.MM.UDP_model[] == this.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDP_model_outsrc[]
this.MM.UDP_model[].getClass() == this.MM.UDP_lengthModel[].getClass()
this.MM.UDP_model[].getClass() == this.testPH.MM.UDP_model[].getClass()
this.MM.TCP_model_in == this.testPH.MM.TCP_model_in

this.MM.UDP_model_in == this.testPH.MM.UDP_model_in
this.MM.TCP_model_out == this.testPH.MM.TCP_model_out
this.MM.UDP_model_out == this.testPH.MM.UDP_model_out
this.MM.TCP_model_outsrc == this.testPH.MM.TCP_model_outsrc
this.MM.UDP_model_outsrc == this.testPH.MM.UDP_model_outsrc
this.MM.TCP_lengthModel == this.testPH.MM.TCP_lengthModel
this.MM.TCP_lengthModel[].getClass() == this.testPH.MM.TCP_lengthModel[].getClass()
this.MM.UDP_lengthModel == this.testPH.MM.UDP_lengthModel
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[].getClass() == this.testPH.MM.UDP_model[].getClass()
this.MM.TCP_lengthModel_in == this.testPH.MM.TCP_lengthModel_in
this.MM.UDP_lengthModel_in == this.testPH.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel_out == this.testPH.MM.TCP_lengthModel_out
this.MM.UDP_lengthModel_out == this.testPH.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel_outsrc == this.testPH.MM.TCP_lengthModel_outsrc
this.MM.UDP_lengthModel_outsrc == this.testPH.MM.UDP_lengthModel_outsrc
this.MM.trunc_length == this.trunc_length
this.MM.trunc_length == this.testPH.trunc_length
this.MM.port_trunc_length == this.port_trunc_length
this.MM.port_trunc_length == this.testPH.port_trunc_length
this.MM.min_sample == this.MM.modification_time
this.MM.min_sample == this.min_sample_num
this.MM.min_sample == this.testPH.count
this.MM.max_sample == this.max_sample_num
this.MM.max_sample == this.testPH.MM.max_sample
this.MM.sample_rate == this.sample_rate
this.MM.sample_rate == this.testPH.MM.sample_rate
this.MM.host_ip == this.host_ip
this.MM.host_ip == this.testPH.host_ip
this.MM.stablesize == this.stablesize
this.MM.stablesize == this.testPH.linkType
this.MM.stabledis == this.stabledis
this.MM.stabledis == this.testPH.MM.stabledis
this.MM.smooth_factor == this.smooth_factor
this.MM.smooth_factor == this.testPH.MM.smooth_factor
this.MM.aggregation_done == this.protocol_udp
this.MM.aggregation_done == this.file_filtering
this.MM.aggregation_done == this.testPH.file_filtering
this.MM.lenProb == this.testPH.MM.lenProb
this.MM.anomalyLengthThreshold == this.testPH.MM.anomalyLengthThreshold
this.MM.lenProb_out == this.testPH.MM.lenProb_out
this.MM.lenProb_out[] == this.MM.lenProb_in[]
this.MM.lenProb_out[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold_out == this.testPH.MM.anomalyLengthThreshold_out
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.lenProb_in == this.testPH.MM.lenProb_in
this.MM.anomalyLengthThreshold_in == this.testPH.MM.anomalyLengthThreshold_in
this.MM.lenProb_outsrc == this.testPH.MM.lenProb_outsrc
this.MM.anomalyLengthThreshold_outsrc == this.testPH.MM.anomalyLengthThreshold_outsrc
this.MM.UDPNormalLength == this.testPH.MM.UDPNormalLength
this.MM.UDPNormalLength[] == this.MM.UDPNormalLength_out[]
this.MM.UDPNormalLength[] == this.MM.UDPNormalLength_in[]
this.MM.UDPNormalLength[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDPNormalLength_out == this.testPH.MM.UDPNormalLength_out
this.MM.UDPNormalLength_in == this.testPH.MM.UDPNormalLength_in
this.MM.UDPNormalLength_outsrc == this.testPH.MM.UDPNormalLength_outsrc
this.MM.normal_length_perc == this.testPH.MM.normal_length_perc

this.host_ip == this.testPH.host_ip
this.in_src_ports[] == this.out_src_ports[]
this.in_src_ports[] == this.testPH.fileModel.b_packets[]
this.in_src_ports[].getClass() == this.testPH.fileModel.b_packets[].getClass()
this.protocol_tcp == this.print
this.protocol_tcp == this.auto_correlation
this.protocol_tcp == this.testPH.normal_alert
this.protocol_udp == this.testPH.file_filtering
this.trunc_length == this.testPH.trunc_length
this.port_trunc_length == this.testPH.port_trunc_length
this.port_keywords == this.testPH.port_keywords
this.port_keywords_range == this.testPH.port_keywords_range
this.file_filtering == this.EPOCH_TRAIN_FLAG
this.file_filtering == this.readtextmodel
this.file_filtering == this.communication_on
this.file_filtering == this.testPH.file_filtering
this.smooth_factor == this.testPH.MM.smooth_factor
this.min_sample_num == this.testPH.count
this.max_sample_num == this.testPH.MM.max_sample
this.sample_rate == this.testPH.MM.sample_rate
this.kernel_h == this.testPH.kernel_h
this.auto_adjust == this.normal_alert
this.auto_adjust == this.OFFLINE_TRAIN
this.auto_adjust == this.OFFLINE_TEST
this.auto_adjust == this.testPH.normal_alert
this.fpRate == this.testPH.smooth_factor
this.fpRate == this.testPH.fpRate
this.stablesize == this.testPH.linkType
this.stabledis == this.testPH.MM.stabledis
this.verbose == this.testPH.verbose
this.string_compare_method == this.testPH.string_compare_method
this.string_compare_threshold == this.testPH.string_compare_threshold
this.testPH.count == this.testPH.anomaly_count
this.testPH.count == this.testPH.old_anomaly
this.testPH.count == this.testPH.MM.min_sample
this.testPH.count == this.testPH.MM.modification_time
this.testPH.count == this.testPH.fileModel.start
this.testPH.count == this.testPH.fileModel.end
this.testPH.count == this.testPH.fileModel.min_sample
this.testPH.count == this.testPH.fileModel.max_sample
this.testPH.count == this.testPH.fileModel.totalPacketNum
this.testPH.count == this.testPH.fileModel.reduceN
this.testPH.trunc_length == this.testPH.MM.trunc_length
this.testPH.port_trunc_length == this.testPH.MM.port_trunc_length
this.testPH.file_filtering == this.testPH.MM.aggregation_done
this.testPH.file_filtering == this.testPH.fileModel.batch
this.testPH.smooth_factor == this.testPH.fpRate
this.testPH.MM.UDP_model[] == this.testPH.MM.TCP_model_in[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDP_model_in[]
this.testPH.MM.UDP_model[] == this.testPH.MM.TCP_model_out[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDP_model_out[]
this.testPH.MM.UDP_model[] == this.testPH.MM.TCP_model_outsrc[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDP_model_outsrc[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDPNormalLength[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_out[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_in[]
this.testPH.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_outsrc[]
this.testPH.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_in[]
this.testPH.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_in[]
this.testPH.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_out[]
this.testPH.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_out[]
this.testPH.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_outsrc[]
this.testPH.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_outsrc[]

this.testPH.MM.sample_rate == this.testPH.fileModel.sample_rate
this.testPH.MM.host_ip == this.testPH.host_ip
this.testPH.MM.stablesize == this.testPH.linkType
this.testPH.MM.lenProb_out[] == this.testPH.MM.lenProb_in[]
this.testPH.MM.lenProb_out[] == this.testPH.MM.lenProb_outsrc[]
this.testPH.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_in[]
this.testPH.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_outsrc[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_anomaly_count[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_count[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_anomaly_count[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_count_out[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_anomaly_count_out[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_count_out[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_anomaly_count_out[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_count_in[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_anomaly_count_in[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_count_in[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_anomaly_count_in[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_count_outsrc[]
this.testPH.adjustTCP_count[] == this.testPH.adjustTCP_anomaly_count_outsrc[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_count_outsrc[]
this.testPH.adjustTCP_count[] == this.testPH.adjustUDP_anomaly_count_outsrc[]
this.testPH.TCP_portready[] == this.testPH.UDP_portready[]
this.testPH.TCP_portready[] == this.testPH.TCP_portready_out[]
this.testPH.TCP_portready[] == this.testPH.UDP_portready_out[]
this.testPH.TCP_portready[] == this.testPH.TCP_portready_in[]
this.testPH.TCP_portready[] == this.testPH.UDP_portready_in[]
this.testPH.TCP_portready[] == this.testPH.TCP_portready_outsrc[]
this.testPH.TCP_portready[] == this.testPH.UDP_portready_outsrc[]
this.testPH.fileModel.b_packets[] == this.testPH.fileModel.b_nums[]
this has only one value
this.PACKET_COUNT one of 1000, 5000
this.TEST_PACKET_COUNT one of -1, 1000
this.FILTER != null
this.FILTER.toString one of "", "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM.TCP_model != null
this.MM.TCP_model.getClass() == java.util.HashMap[].class
this.MM.TCP_model[].getClass() one of [null, null, null,

null, null,

this.MM.UDP_model[] elements == null
this.MM.UDP_model[].getClass() == [null, null, null,

null, null,

null, null, null,

0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.MM.anomalyLengthThreshold_out[] elements == 0.0
this.MM.lenProb_in != null
this.MM.anomalyLengthThreshold_in != null
this.MM.lenProb_outsrc != null
this.MM.anomalyLengthThreshold_outsrc != null
this.MM.UDPNormalLength.getClass() == java.util.HashMap[].class
this.MM.UDPNormalLength[] contains only nulls and has only one value, of length 1024
this.MM.UDPNormalLength[] elements == null
this.MM.normal_length_perc == 0.0
this.config_device has only one value
this.config_device != null
this.config_device.toString == ”default”
this.host_ip != null
this.host_ip.toString one of ”128.59.16.101”, ”localhost”
this.in_dst_ports has only one value
this.in_dst_ports != null
this.in_dst_ports[] elements has only one value
this.in_dst_ports[] elements != null
this.in_dst_ports[].getClass() elements == java.lang.String.class
this.in_dst_ports[].getClass() one of [], [java.lang.String]
this.in_src_ports has only one value
this.in_src_ports != null
this.in_src_ports[] == []
this.in_src_ports[].getClass() == []
this.out_dst_ports has only one value
this.out_dst_ports != null
this.out_dst_ports[] elements has only one value
this.out_dst_ports[] elements != null
this.out_src_ports has only one value
this.out_src_ports != null
this.protocol_tcp == true
this.trunc_length == -1
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.file_filtering == false
this.smooth_factor == 0.0010
this.min_sample_num one of 0, 20
this.max_sample_num one of 200, 1000

this.sample_rate one of 0.0, 0.05
this.kernel_h == 5.0
this.normal_length_perc == 0.999
this.threshold_step one of 0.1, 0.2
this.fpRate one of 0.0010, 0.01
this.stablesize == 1
this.stabledis == 0.05
this.cluster_threshold one of 0.2, 0.3
this.modelfile != null
this.modelfile.toString == "model.dat"
this.verbose one of 1, 2
this.offlineTrainFile != null
this.offlineTrainFile.toString one of "", "train-p2.dmp"
this.TRAIN_HOUR one of 0.01, 0.02
this.STABLE_PORTS_PERC == 0.9
this.offlineTestFile != null
this.offlineTestFile.toString one of "", "test-p3.dmp"
this.string_compare_method != null
this.string_compare_method.toString == "string"
this.string_compare_threshold == 0.5
this.textmodelfile has only one value
this.textmodelfile != null
this.textmodelfile.toString == ""
this.communication_time one of 0, 1800
this.comm_ips has only one value
this.comm_ips != null
this.comm_ips[] elements != null
this.comm_ips[].getClass() elements == java.lang.String.class
this.comm_ips[].getClass() one of [], [java.lang.String, java.lang.String]
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.trainPH == null
this.testPH.count == 0
this.testPH.trunc_length == -1
this.testPH.port_trunc_length has only one value
this.testPH.port_keywords has only one value
this.testPH.port_keywords_range has only one value
this.testPH.file_filtering == false
this.testPH.smooth_factor == 0.0010000000474974513
this.testPH.maxnum == 1999.999905005102
this.testPH.threshold one of 64.0, 409.0
this.testPH.normal_alert == true
this.testPH.verbose == 2
this.testPH.temp == null
this.testPH.MM has only one value
this.testPH.MM.TCP_model has only one value
this.testPH.MM.TCP_model.getClass() == java.util.HashMap[].class
this.testPH.MM.TCP_model[] has only one value, of length 1024
this.testPH.MM.TCP_model[].getClass() == [null, null, null,

null, null,

null, null,

null, null, null,

this.testPH.MM.anomalyLengthThreshold[] elements one of 0.0, 1.76272529158797E-5
this.testPH.MM.lenProb_out has only one value
this.testPH.MM.lenProb_out[] contains only nulls and has only one value, of length 1024
this.testPH.MM.lenProb_out[] elements == null
this.testPH.MM.anomalyLengthThreshold_out has only one value
this.testPH.MM.anomalyLengthThreshold_out[] == [0.0, 0.0, 0.0,

0, 0, 0,

this.testPH.sdf has only one value
this.testPH.fileModel has only one value
this.testPH.fileModel.centroids has only one value
this.testPH.fileModel.centroids[] contains no nulls and has only one value, of length 35
this.testPH.fileModel.centroids[].getClass() == [OneCentroid, OneCentroid]
this.testPH.fileModel.centroids[].getClass() elements == OneCentroid.class
this.testPH.fileModel.NMax == 10
this.testPH.fileModel.b_packets has only one value
this.testPH.fileModel.b_packets[] == []
this.testPH.fileModel.b_packets[].getClass() == []
this.testPH.fileModel.b_nums has only one value
this.testPH.packetIndex one of 0, 299
this.testPH.TCP == 6
this.testPH.UDP == 17
this.testPH.maliciousBuffer has only one value
this.testPH.sign == null
this.testPH.linkType == 1
this.server == null
this.cc == null
this.PACKET_COUNT >= this.TEST_PACKET_COUNT
this.PACKET_COUNT > this.MM.trunc_length
this.PACKET_COUNT > this.MM.min_sample
this.PACKET_COUNT > this.MM.max_sample
this.PACKET_COUNT > this.MM.stablesize
this.PACKET_COUNT > this.trunc_length
this.PACKET_COUNT > this.min_sample_num
this.PACKET_COUNT > this.max_sample_num
this.PACKET_COUNT > this.stablesize
this.PACKET_COUNT > this.verbose
this.PACKET_COUNT > this.communication_time
this.TEST_PACKET_COUNT < this.MM.min_sample
this.TEST_PACKET_COUNT < this.MM.max_sample
this.TEST_PACKET_COUNT < this.MM.stablesize
this.TEST_PACKET_COUNT >= this.trunc_length
this.TEST_PACKET_COUNT != this.min_sample_num
this.TEST_PACKET_COUNT != this.max_sample_num
this.TEST_PACKET_COUNT != this.stablesize
this.TEST_PACKET_COUNT != this.verbose
this.TEST_PACKET_COUNT != this.communication_time
this.FILTER.toString > this.MM.host_ip.toString
this.FILTER.toString != this.config_device.toString
this.FILTER.toString != this.host_ip.toString
this.FILTER.toString < this.modelfile.toString
this.FILTER.toString <= this.offlineTrainFile.toString
this.FILTER.toString <= this.offlineTestFile.toString
this.FILTER.toString < this.string_compare_method.toString
this.FILTER.toString >= this.textmodelfile.toString
this.MM.TCP_model.getClass() == this.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() != this.MM.lenProb.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model.getClass()
this.MM.TCP_model[] == this.testPH.MM.TCP_model[] (elementwise)
this.MM.UDP_model[] == this.MM.UDPNormalLength[] (elementwise)

this.MM.UDP_model[] is the reverse of this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model[] (elementwise)
this.MM.UDP_model[] is the reverse of this.testPH.MM.UDP_model[]
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel[] == this.testPH.MM.TCP_lengthModel[] (elementwise)
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel[] (elementwise)
this.MM.UDP_lengthModel[] is the reverse of this.testPH.MM.UDP_lengthModel[]
this.MM.trunc_length < this.min_sample_num
this.MM.trunc_length < this.max_sample_num
this.MM.trunc_length < this.verbose
this.MM.trunc_length < this.communication_time
this.MM.min_sample < this.max_sample_num
this.MM.min_sample < this.verbose
this.MM.min_sample < this.communication_time
this.MM.min_sample in this.testPH.adjustTCP_count[]
this.testPH.adjustTCP_count[] elements == this.MM.min_sample
this.MM.max_sample > this.min_sample_num
this.MM.max_sample > this.verbose
this.MM.max_sample < this.communication_time
this.MM.host_ip.toString < this.config_device.toString
this.MM.host_ip.toString == this.host_ip.toString
this.MM.host_ip.toString < this.modelfile.toString
this.MM.host_ip.toString < this.offlineTrainFile.toString
this.MM.host_ip.toString < this.offlineTestFile.toString
this.MM.host_ip.toString < this.string_compare_method.toString
this.MM.host_ip.toString > this.textmodelfile.toString
this.MM.host_ip.toString == this.testPH.host_ip.toString
this.MM.stablesize > this.min_sample_num
this.MM.stablesize < this.max_sample_num
this.MM.stablesize < this.verbose
this.MM.stablesize < this.communication_time
this.MM.anomalyLengthThreshold[] elements < this.MM.stabledis
this.MM.stabledis < this.threshold_step
this.MM.stabledis > this.fpRate
this.MM.stabledis < this.cluster_threshold
this.MM.stabledis > this.TRAIN_HOUR
this.MM.smooth_factor > this.sample_rate
this.MM.lenProb.getClass() == this.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb.getClass()
this.MM.lenProb[] == this.testPH.MM.lenProb[] (elementwise)
this.MM.anomalyLengthThreshold[] >= this.MM.anomalyLengthThreshold_out[] (elementwise)
this.MM.anomalyLengthThreshold[] elements >= this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] elements < this.kernel_h
this.MM.anomalyLengthThreshold[] elements < this.normal_length_perc
this.MM.anomalyLengthThreshold[] elements < this.threshold_step
this.MM.anomalyLengthThreshold[] elements < this.fpRate
this.MM.anomalyLengthThreshold[] elements < this.stabledis
this.MM.anomalyLengthThreshold[] elements < this.cluster_threshold
this.MM.anomalyLengthThreshold[] elements < this.TRAIN_HOUR
this.MM.anomalyLengthThreshold[] elements < this.STABLE_PORTS_PERC
this.MM.anomalyLengthThreshold[] elements < this.string_compare_threshold
this.MM.anomalyLengthThreshold[] == this.testPH.MM.anomalyLengthThreshold[] (elementwise)
this.testPH.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]

this.MM.lenProb_out[] == this.testPH.MM.lenProb_out[] (elementwise)
this.MM.lenProb_out[] is the reverse of this.testPH.MM.lenProb_out[]
this.MM.anomalyLengthThreshold_out[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_out[] (elementwise)
this.MM.anomalyLengthThreshold_out[] is the reverse of this.testPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] elements == this.testPH.MM.normal_length_perc
this.testPH.MM.normal_length_perc in this.MM.anomalyLengthThreshold_out[]
this.MM.UDPNormalLength.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.UDPNormalLength.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.UDPNormalLength.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.UDPNormalLength.getClass() == this.testPH.MM.TCP_model.getClass()
this.MM.UDPNormalLength[] == this.testPH.MM.UDP_model[] (elementwise)
this.MM.UDPNormalLength[] is the reverse of this.testPH.MM.UDP_model[]
this.MM.normal_length_perc < this.threshold_step
this.MM.normal_length_perc < this.fpRate
this.MM.normal_length_perc < this.cluster_threshold
this.MM.normal_length_perc < this.TRAIN_HOUR
this.MM.normal_length_perc in this.testPH.MM.anomalyLengthThreshold[]
this.MM.normal_length_perc in this.testPH.MM.anomalyLengthThreshold_out[]
this.testPH.MM.anomalyLengthThreshold_out[] elements == this.MM.normal_length_perc
this.config_device.toString != this.host_ip.toString
this.config_device.toString < this.modelfile.toString
this.config_device.toString != this.offlineTrainFile.toString
this.config_device.toString != this.offlineTestFile.toString
this.config_device.toString < this.string_compare_method.toString
this.config_device.toString > this.textmodelfile.toString
this.host_ip.toString < this.modelfile.toString
this.host_ip.toString != this.offlineTrainFile.toString
this.host_ip.toString != this.offlineTestFile.toString
this.host_ip.toString < this.string_compare_method.toString
this.host_ip.toString > this.textmodelfile.toString
this.host_ip.toString == this.testPH.host_ip.toString
this.in_src_ports[] is the reverse of this.testPH.fileModel.b_packets[]
this.protocol_tcp in this.testPH.TCP_portready[]
this.testPH.TCP_portready[] elements == this.protocol_tcp
this.trunc_length < this.min_sample_num
this.trunc_length < this.max_sample_num
this.trunc_length < this.verbose
this.trunc_length < this.communication_time
this.smooth_factor != this.sample_rate
this.min_sample_num < this.max_sample_num
this.min_sample_num != this.stablesize
this.min_sample_num != this.verbose
this.min_sample_num != this.communication_time
this.min_sample_num in this.testPH.adjustTCP_count[]
this.testPH.adjustTCP_count[] elements == this.min_sample_num
this.max_sample_num > this.stablesize
this.max_sample_num > this.verbose
this.max_sample_num != this.communication_time
this.kernel_h > this.threshold_step
this.kernel_h > this.fpRate
this.kernel_h > this.cluster_threshold
this.kernel_h > this.TRAIN_HOUR
this.normal_length_perc > this.threshold_step
this.normal_length_perc > this.fpRate
this.normal_length_perc > this.cluster_threshold
this.normal_length_perc > this.TRAIN_HOUR
this.auto_adjust in this.testPH.TCP_portready[]
this.testPH.TCP_portready[] elements == this.auto_adjust
this.threshold_step > this.fpRate
this.threshold_step > this.stabledis
this.threshold_step <= this.cluster_threshold

this.threshold_step > this.TRAIN_HOUR
this.threshold_step < this.STABLE_PORTS_PERC
this.threshold_step < this.string_compare_threshold
this.fpRate < this.stabledis
this.fpRate < this.cluster_threshold
this.fpRate <= this.TRAIN_HOUR
this.fpRate < this.STABLE_PORTS_PERC
this.fpRate < this.string_compare_threshold
this.stablesize <= this.verbose
this.stablesize != this.communication_time
this.stabledis < this.cluster_threshold
this.stabledis > this.TRAIN_HOUR
this.cluster_threshold > this.TRAIN_HOUR
this.cluster_threshold < this.STABLE_PORTS_PERC
this.cluster_threshold < this.string_compare_threshold
this.modelfile.toString != this.offlineTrainFile.toString
this.modelfile.toString != this.offlineTestFile.toString
this.modelfile.toString < this.string_compare_method.toString
this.modelfile.toString > this.textmodelfile.toString
this.verbose != this.communication_time
this.offlineTrainFile.toString >= this.offlineTestFile.toString
this.offlineTrainFile.toString != this.string_compare_method.toString
this.offlineTrainFile.toString >= this.textmodelfile.toString
this.TRAIN_HOUR < this.STABLE_PORTS_PERC
this.TRAIN_HOUR < this.string_compare_threshold
this.offlineTestFile.toString != this.string_compare_method.toString
this.offlineTestFile.toString >= this.textmodelfile.toString
this.string_compare_method.toString > this.textmodelfile.toString
this.string_compare_method.toString == this.testPH.string_compare_method.toString
this.testPH.adjustTCP_count[] elements == this.testPH.count
this.testPH.count in this.testPH.adjustTCP_count[]
this.testPH.TCP_portready[] elements == this.testPH.normal_alert
this.testPH.normal_alert in this.testPH.TCP_portready[]
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDP_model.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_in.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_in.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_out.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_out.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_outsrc.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_outsrc.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_out.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_in.getClass()
this.testPH.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_outsrc.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_in.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_in.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_out.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_out.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_outsrc.getClass()
this.testPH.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_outsrc.getClass()
this.testPH.MM.host_ip.toString == this.testPH.host_ip.toString
this.testPH.MM.lenProb.getClass() == this.testPH.MM.lenProb_out.getClass()
this.testPH.MM.lenProb.getClass() == this.testPH.MM.lenProb_in.getClass()
this.testPH.MM.lenProb.getClass() == this.testPH.MM.lenProb_outsrc.getClass()
this.testPH.MM.normal_length_perc in this.testPH.MM.anomalyLengthThreshold[]
this.testPH.MM.anomalyLengthThreshold_out[] elements == this.testPH.MM.normal_length_perc
this.testPH.MM.normal_length_perc in this.testPH.MM.anomalyLengthThreshold_out[]
==========
Payload.Payload():::EXIT
this.PACKET_COUNT == this.TEST_PACKET_COUNT
this.FILTER == this.offlineTrainFile
this.FILTER == this.offlineTestFile

this.FILTER == this.textmodelfile
this.in_dst_ports[] == this.in_src_ports[]
this.in_dst_ports[] == this.out_dst_ports[]
this.in_dst_ports[] == this.out_src_ports[]
this.in_dst_ports[] == this.comm_ips[]
this.stablesize == this.verbose
this.PACKET_COUNT == 1000
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "localhost"
this.in_dst_ports[] == []
this.in_dst_ports[].getClass() == []
this.min_sample_num == 20
this.max_sample_num == 200
this.sample_rate == 0.05
this.threshold_step == 0.1
this.fpRate == 0.01
this.cluster_threshold == 0.3
this.modelfile has only one value
this.string_compare_method has only one value
this.communication_time == 0
this.testPH == null
this.FILTER.toString == this.offlineTrainFile.toString
this.FILTER.toString == this.offlineTestFile.toString
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.adjust_correlation_ports():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == this.testPH.MM.trunc_length
this.MM.TCP_model[] == this.testPH.MM.TCP_model[]
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_outsrc[]
this.MM.TCP_lengthModel[] == this.testPH.MM.TCP_lengthModel[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_outsrc[]
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.testPH.MM.lenProb[]
this.MM.anomalyLengthThreshold[] == this.testPH.MM.anomalyLengthThreshold[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_in[]

this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_outsrc[]
this.host_ip == this.testPH.MM.host_ip
this.in_src_ports[] == this.testPH.fileModel.b_nums[]
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == this.testPH.MM.aggregation_done
this.protocol_udp == this.testPH.stable
this.protocol_udp == this.testPH.fileModel.batch
this.port_trunc_length == this.testPH.MM.port_trunc_length
this.min_sample_num == this.testPH.anomaly_count
this.min_sample_num == this.testPH.old_anomaly
this.min_sample_num == this.testPH.MM.min_sample
this.min_sample_num == this.testPH.MM.modification_time
this.min_sample_num == this.testPH.fileModel.start
this.min_sample_num == this.testPH.fileModel.end
this.min_sample_num == this.testPH.fileModel.min_sample
this.min_sample_num == this.testPH.fileModel.max_sample
this.min_sample_num == this.testPH.fileModel.totalPacketNum
this.min_sample_num == this.testPH.fileModel.reduceN
this.min_sample_num == this.testPH.packetIndex
this.sample_rate == this.testPH.fileModel.sample_rate
this.threshold_step == this.cluster_threshold
this.stablesize == this.testPH.MM.stablesize
this.testPH.in_dst_ports_set == this.testPH.in_src_ports_set
this.testPH.in_dst_ports_set == this.testPH.out_dst_ports_set
this.testPH.in_dst_ports_set == this.testPH.out_src_ports_set
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0,

this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_outsrc.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_outsrc.getClass()
this.host_ip.toString == this.testPH.MM.host_ip.toString
==========
Payload.adjust_correlation_ports():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == this.testPH.MM.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.TEST_PACKET_COUNT == orig(this.testPH.trunc_length)
this.TEST_PACKET_COUNT == orig(this.testPH.MM.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM == orig(this.testPH.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model == orig(this.testPH.MM.TCP_model)
this.MM.TCP_model[] == this.testPH.MM.TCP_model[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.testPH.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model == orig(this.testPH.MM.UDP_model)
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.testPH.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.testPH.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.testPH.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDPNormalLength[])

this.MM.UDP_model[] == orig(this.testPH.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.testPH.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.TCP_model_in == orig(this.testPH.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.UDP_model_in == orig(this.testPH.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.TCP_model_out == orig(this.testPH.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.UDP_model_out == orig(this.testPH.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.TCP_model_outsrc == orig(this.testPH.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.testPH.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.testPH.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == this.testPH.MM.TCP_lengthModel[]
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.TCP_lengthModel[] == orig(this.testPH.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel == orig(this.testPH.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.testPH.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel_in == orig(this.testPH.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.testPH.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel_out == orig(this.testPH.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.testPH.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel_outsrc == orig(this.testPH.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.testPH.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb == orig(this.testPH.MM.lenProb)
this.MM.lenProb[] == this.testPH.MM.lenProb[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.testPH.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold == orig(this.testPH.MM.anomalyLengthThreshold)

this.MM.anomalyLengthThreshold[] == this.testPH.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.testPH.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out == orig(this.testPH.MM.lenProb_out)
this.MM.lenProb_out[] == this.testPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_outsrc[]
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.lenProb_out[] == orig(this.testPH.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.testPH.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.testPH.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out == orig(this.testPH.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.anomalyLengthThreshold_out[] == orig(this.testPH.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.testPH.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.testPH.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.lenProb_in == orig(this.testPH.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.anomalyLengthThreshold_in == orig(this.testPH.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.lenProb_outsrc == orig(this.testPH.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.testPH.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.testPH.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_out == orig(this.testPH.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_in == orig(this.testPH.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.UDPNormalLength_outsrc == orig(this.testPH.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.MM.normal_length_perc == orig(this.testPH.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == this.testPH.MM.host_ip
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.host_ip == orig(this.testPH.MM.host_ip)
this.host_ip == orig(this.testPH.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == this.testPH.fileModel.b_nums[]
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.in_src_ports[] == orig(this.testPH.fileModel.b_packets[])
this.in_src_ports[] == orig(this.testPH.fileModel.b_nums[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN

this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_tcp == orig(this.testPH.normal_alert)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == this.testPH.MM.aggregation_done
this.protocol_udp == this.testPH.stable
this.protocol_udp == this.testPH.fileModel.batch
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.protocol_udp == orig(this.testPH.file_filtering)
this.protocol_udp == orig(this.testPH.MM.aggregation_done)
this.protocol_udp == orig(this.testPH.stable)
this.protocol_udp == orig(this.testPH.fileModel.batch)
this.port_trunc_length == this.testPH.MM.port_trunc_length
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.testPH.port_trunc_length)
this.port_trunc_length == orig(this.testPH.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords == orig(this.testPH.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.port_keywords_range == orig(this.testPH.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.smooth_factor == orig(this.testPH.MM.smooth_factor)
this.min_sample_num == this.testPH.anomaly_count
this.min_sample_num == this.testPH.old_anomaly
this.min_sample_num == this.testPH.MM.min_sample
this.min_sample_num == this.testPH.MM.modification_time
this.min_sample_num == this.testPH.fileModel.start
this.min_sample_num == this.testPH.fileModel.end
this.min_sample_num == this.testPH.fileModel.min_sample
this.min_sample_num == this.testPH.fileModel.max_sample
this.min_sample_num == this.testPH.fileModel.totalPacketNum
this.min_sample_num == this.testPH.fileModel.reduceN
this.min_sample_num == this.testPH.packetIndex
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.min_sample_num == orig(this.testPH.count)
this.min_sample_num == orig(this.testPH.anomaly_count)
this.min_sample_num == orig(this.testPH.old_anomaly)
this.min_sample_num == orig(this.testPH.MM.min_sample)
this.min_sample_num == orig(this.testPH.MM.modification_time)
this.min_sample_num == orig(this.testPH.fileModel.start)
this.min_sample_num == orig(this.testPH.fileModel.end)
this.min_sample_num == orig(this.testPH.fileModel.min_sample)
this.min_sample_num == orig(this.testPH.fileModel.max_sample)
this.min_sample_num == orig(this.testPH.fileModel.totalPacketNum)
this.min_sample_num == orig(this.testPH.fileModel.reduceN)

this.min_sample_num == orig(this.testPH.packetIndex)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.max_sample_num == orig(this.testPH.MM.max_sample)
this.sample_rate == this.testPH.fileModel.sample_rate
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.sample_rate == orig(this.testPH.MM.sample_rate)
this.sample_rate == orig(this.testPH.fileModel.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.kernel_h == orig(this.testPH.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.fpRate == orig(this.testPH.fpRate)
this.stablesize == this.testPH.MM.stablesize
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stablesize == orig(this.testPH.MM.stablesize)
this.stablesize == orig(this.testPH.linkType)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.stabledis == orig(this.testPH.MM.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.verbose == orig(this.testPH.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_method == orig(this.testPH.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.string_compare_threshold == orig(this.testPH.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.testPH.smooth_factor == orig(this.testPH.smooth_factor)
this.testPH.maxnum == orig(this.testPH.maxnum)
this.testPH.threshold == orig(this.testPH.threshold)
this.testPH.temp == orig(this.testPH.temp)
this.testPH.in_dst_ports_set == this.testPH.in_src_ports_set
this.testPH.in_dst_ports_set == this.testPH.out_dst_ports_set
this.testPH.in_dst_ports_set == this.testPH.out_src_ports_set
this.testPH.in_dst_ports_set == orig(this.testPH.in_dst_ports_set)
this.testPH.in_dst_ports_set == orig(this.testPH.in_src_ports_set)
this.testPH.in_dst_ports_set == orig(this.testPH.out_dst_ports_set)
this.testPH.in_dst_ports_set == orig(this.testPH.out_src_ports_set)
this.testPH.adjustTCP_count == orig(this.testPH.adjustTCP_count)
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_count[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_anomaly_count[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_count[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_anomaly_count[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_count_out[])

this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_anomaly_count_out[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_count_out[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_anomaly_count_out[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_count_in[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_anomaly_count_in[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_count_in[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_anomaly_count_in[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_count_outsrc[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustTCP_anomaly_count_outsrc[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_count_outsrc[])
this.testPH.adjustTCP_count[] == orig(this.testPH.adjustUDP_anomaly_count_outsrc[])
this.testPH.adjustTCP_anomaly_count == orig(this.testPH.adjustTCP_anomaly_count)
this.testPH.adjustUDP_count == orig(this.testPH.adjustUDP_count)
this.testPH.adjustUDP_anomaly_count == orig(this.testPH.adjustUDP_anomaly_count)
this.testPH.TCP_portready == orig(this.testPH.TCP_portready)
this.testPH.TCP_portready[] == orig(this.testPH.TCP_portready[])
this.testPH.TCP_portready[] == orig(this.testPH.UDP_portready[])
this.testPH.TCP_portready[] == orig(this.testPH.TCP_portready_out[])
this.testPH.TCP_portready[] == orig(this.testPH.UDP_portready_out[])
this.testPH.TCP_portready[] == orig(this.testPH.TCP_portready_in[])
this.testPH.TCP_portready[] == orig(this.testPH.UDP_portready_in[])
this.testPH.TCP_portready[] == orig(this.testPH.TCP_portready_outsrc[])
this.testPH.TCP_portready[] == orig(this.testPH.UDP_portready_outsrc[])
this.testPH.UDP_portready == orig(this.testPH.UDP_portready)
this.testPH.adjustTCP_count_out == orig(this.testPH.adjustTCP_count_out)
this.testPH.adjustTCP_anomaly_count_out == orig(this.testPH.adjustTCP_anomaly_count_out)
this.testPH.adjustUDP_count_out == orig(this.testPH.adjustUDP_count_out)
this.testPH.adjustUDP_anomaly_count_out == orig(this.testPH.adjustUDP_anomaly_count_out)
this.testPH.TCP_portready_out == orig(this.testPH.TCP_portready_out)
this.testPH.UDP_portready_out == orig(this.testPH.UDP_portready_out)
this.testPH.adjustTCP_count_in == orig(this.testPH.adjustTCP_count_in)
this.testPH.adjustTCP_anomaly_count_in == orig(this.testPH.adjustTCP_anomaly_count_in)
this.testPH.adjustUDP_count_in == orig(this.testPH.adjustUDP_count_in)
this.testPH.adjustUDP_anomaly_count_in == orig(this.testPH.adjustUDP_anomaly_count_in)
this.testPH.TCP_portready_in == orig(this.testPH.TCP_portready_in)
this.testPH.UDP_portready_in == orig(this.testPH.UDP_portready_in)
this.testPH.adjustTCP_count_outsrc == orig(this.testPH.adjustTCP_count_outsrc)
this.testPH.adjustTCP_anomaly_count_outsrc == orig(this.testPH.adjustTCP_anomaly_count_outsrc)
this.testPH.adjustUDP_count_outsrc == orig(this.testPH.adjustUDP_count_outsrc)
this.testPH.adjustUDP_anomaly_count_outsrc == orig(this.testPH.adjustUDP_anomaly_count_outsrc)
this.testPH.TCP_portready_outsrc == orig(this.testPH.TCP_portready_outsrc)
this.testPH.UDP_portready_outsrc == orig(this.testPH.UDP_portready_outsrc)
this.testPH.sdf == orig(this.testPH.sdf)
this.testPH.fileModel == orig(this.testPH.fileModel)
this.testPH.fileModel.centroids == orig(this.testPH.fileModel.centroids)
this.testPH.fileModel.centroids[] == orig(this.testPH.fileModel.centroids[])
this.testPH.fileModel.NMax == orig(this.testPH.fileModel.NMax)
this.testPH.fileModel.b_packets == orig(this.testPH.fileModel.b_packets)
this.testPH.fileModel.b_nums == orig(this.testPH.fileModel.b_nums)
this.testPH.TCP == orig(this.testPH.TCP)
this.testPH.UDP == orig(this.testPH.UDP)
this.testPH.maliciousBuffer == orig(this.testPH.maliciousBuffer)
this.testPH.sign == orig(this.testPH.sign)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] has only one value, of length 1024

this.MM.TCP_model[].getClass() == [null, null, null,

0.0, 0.0, 0.0,

this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH has only one value
this.testPH.threshold == 64.0
this.testPH.in_dst_ports_set == null
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.testPH.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.UDP_lengthModel.getClass())

this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.testPH.MM.UDP_lengthModel_outsrc.getClass())
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.testPH.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.testPH.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.testPH.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.testPH.MM.lenProb_outsrc.getClass())
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == this.testPH.MM.host_ip.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.host_ip.toString == orig(this.testPH.MM.host_ip.toString)
this.host_ip.toString == orig(this.testPH.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.string_compare_method.toString == orig(this.testPH.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.buildPortModel():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel

this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

this.MM.lenProb has only one value
this.MM.lenProb[] contains only nulls and has only one value, of length 1024
this.MM.lenProb[] elements == null
this.MM.anomalyLengthThreshold has only one value
this.MM.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
==========
Payload.buildPortModel():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.MM.UDP_model[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_in[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_in[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_out[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_out[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.MM.lenProb_out[])

this.MM.lenProb[] == orig(this.MM.lenProb_in[])
this.MM.lenProb[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.MM.min_sample)

this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

this.MM.anomalyLengthThreshold[] elements == 0.0
this.MM.lenProb_out has only one value
this.MM.anomalyLengthThreshold_out has only one value
this.MM.lenProb_in has only one value
this.MM.anomalyLengthThreshold_in has only one value
this.MM.lenProb_outsrc has only one value
this.MM.anomalyLengthThreshold_outsrc has only one value
this.MM.UDPNormalLength == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)

this.textmodelfile.toString == orig(this.textmodelfile.toString)

Payload.buildport(java.util.BitSet, java.util.Vector):::ENTER

this.TEST_PACKET_COUNT == this.trunc_length

this.MM.TCP_model[] == this.MM.UDP_model[]

this.MM.TCP_model[] == this.MM.TCP_model_in[]

this.MM.TCP_model[] == this.MM.UDP_model_in[]

this.MM.TCP_model[] == this.MM.TCP_model_out[]

this.MM.TCP_model[] == this.MM.UDP_model_out[]

this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]

this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]

this.MM.TCP_lengthModel == this.MM.UDP_lengthModel

this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in

this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in

this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out

this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out

this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc

this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc

this.MM.modification_time == this.min_sample_num

this.MM.lenProb[] == this.MM.lenProb_out[]

this.MM.lenProb[] == this.MM.lenProb_in[]

this.MM.lenProb[] == this.MM.lenProb_outsrc[]

this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]

this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]

this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]

this.MM.UDPNormalLength == this.MM.UDPNormalLength_out

this.MM.UDPNormalLength == this.MM.UDPNormalLength_in

this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc

this.protocol_tcp == this.auto_adjust

this.protocol_tcp == this.normal_alert

this.protocol_tcp == this.OFFLINE_TRAIN

this.protocol_tcp == this.OFFLINE_TEST

this.protocol_udp == this.file_filtering

this.protocol_udp == this.EPOCH_TRAIN_FLAG

this.protocol_udp == this.readtextmodel

this.protocol_udp == this.communication_on

this.threshold_step == this.cluster_threshold

this.PACKET_COUNT == 5000

this.TEST_PACKET_COUNT == -1

this.FILTER has only one value

this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"

this.MM has only one value

this.MM.TCP_model has only one value

this.MM.TCP_model[] contains only nulls and has only one value, of length 1024

this.MM.TCP_model[] elements == null

this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.MM.anomalyLengthThreshold[] elements == 0.0
this.MM.lenProb_out has only one value
this.MM.anomalyLengthThreshold_out has only one value
this.MM.lenProb_in has only one value
this.MM.anomalyLengthThreshold_in has only one value
this.MM.lenProb_outsrc has only one value
this.MM.anomalyLengthThreshold_outsrc has only one value
this.MM.UDPNormalLength == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
arg1[].getClass() elements == java.lang.String.class
arg1[].getClass() one of [], [java.lang.String]
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
==========
Payload.buildport(java.util.BitSet, java.util.Vector):::EXIT422
this.in_src_ports[] == arg1[]
this.out_src_ports[] == arg1[]
arg1[] == orig(this.in_src_ports[])
arg1[] == orig(this.out_src_ports[])
arg1[] == []
arg1[].getClass() == []
==========
Payload.buildport(java.util.BitSet, java.util.Vector):::EXIT439
arg1[].getClass() == [java.lang.String]
==========
Payload.buildport(java.util.BitSet, java.util.Vector):::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)

this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.TCP_model[] == orig(this.MM.UDP_model[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_in[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_in[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_out[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_out[])
this.MM.TCP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.TCP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.lenProb[] == orig(this.MM.lenProb_out[])
this.MM.lenProb[] == orig(this.MM.lenProb_in[])
this.MM.lenProb[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)

this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.MM.stabledis)

this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
arg1[] == orig(arg1[])
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null,

this.MM.lenProb_in has only one value
this.MM.anomalyLengthThreshold_in has only one value
this.MM.lenProb_outsrc has only one value
this.MM.anomalyLengthThreshold_outsrc has only one value
this.MM.UDPNormalLength == null
this.host_ip has only one value
this.host_ip.toString == ”128.59.16.101”
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == ”train-p2.dmp”
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == ”test-p3.dmp”
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
arg1[].getClass() elements == java.lang.String.class
arg1[].getClass() one of [], [java.lang.String]
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements == this.MM.normal_length_perc
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.getTestPacketHandler():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.modification_time == this.min_sample_num
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN

this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

this.MM.TCP_model_outsrc has only one value
this.MM.UDP_model_outsrc has only one value
this.MM.TCP_lengthModel has only one value
this.MM.UDP_lengthModel has only one value
this.MM.TCP_lengthModel_in has only one value
this.MM.UDP_lengthModel_in has only one value
this.MM.TCP_lengthModel_out has only one value
this.MM.UDP_lengthModel_out has only one value
this.MM.TCP_lengthModel_outsrc has only one value
this.MM.UDP_lengthModel_outsrc has only one value
this.MM.lenProb has only one value
this.MM.lenProb[] has only one value, of length 1024
this.MM.anomalyLengthThreshold has only one value
this.MM.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
==========
Payload.getTestPacketHandler():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == this.testPH.MM.trunc_length
this.TEST_PACKET_COUNT == return.trunc_length
this.TEST_PACKET_COUNT == return.MM.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == return.MM
this.MM == orig(this.MM)
this.MM.TCP_model == return.MM.TCP_model
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.testPH.MM.TCP_model[]
this.MM.TCP_model[] == return.MM.TCP_model[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == return.MM.UDP_model
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == return.MM.UDP_model[]
this.MM.UDP_model[] == return.MM.TCP_model_in[]
this.MM.UDP_model[] == return.MM.UDP_model_in[]
this.MM.UDP_model[] == return.MM.TCP_model_out[]
this.MM.UDP_model[] == return.MM.UDP_model_out[]
this.MM.UDP_model[] == return.MM.TCP_model_outsrc[]

this.MM.UDP_model[] == return.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == return.MM.UDPNormalLength[]
this.MM.UDP_model[] == return.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == return.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == return.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == return.MM.TCP_model_in
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == return.MM.UDP_model_in
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == return.MM.TCP_model_out
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == return.MM.UDP_model_out
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == return.MM.TCP_model_outsrc
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == return.MM.UDP_model_outsrc
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == return.MM.TCP_lengthModel
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == this.testPH.MM.TCP_lengthModel[]
this.MM.TCP_lengthModel[] == return.MM.TCP_lengthModel[]
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == return.MM.UDP_lengthModel
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == return.MM.UDP_lengthModel[]
this.MM.UDP_lengthModel[] == return.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == return.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == return.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == return.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == return.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == return.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == return.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == return.MM.UDP_lengthModel_in
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == return.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == return.MM.UDP_lengthModel_out

this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == return.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == return.MM.UDP_lengthModel_outsrc
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == return.MM.lenProb
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.testPH.MM.lenProb[]
this.MM.lenProb[] == return.MM.lenProb[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == return.MM.anomalyLengthThreshold
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.testPH.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == return.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == return.MM.lenProb_out
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == this.testPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_outsrc[]
this.MM.lenProb_out[] == return.MM.lenProb_out[]
this.MM.lenProb_out[] == return.MM.lenProb_in[]
this.MM.lenProb_out[] == return.MM.lenProb_outsrc[]
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == return.MM.anomalyLengthThreshold_out
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold_out[] == return.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == return.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == return.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == return.MM.lenProb_in
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == return.MM.anomalyLengthThreshold_in
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == return.MM.lenProb_outsrc
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == return.MM.anomalyLengthThreshold_outsrc
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == return.MM.UDPNormalLength
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == return.MM.UDPNormalLength_out
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == return.MM.UDPNormalLength_in
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == return.MM.UDPNormalLength_outsrc
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == return.MM.normal_length_perc
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == this.testPH.MM.host_ip
this.host_ip == return.MM.host_ip
this.host_ip == return.host_ip
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)

this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == this.testPH.fileModel.b_nums[]
this.in_src_ports[] == return.fileModel.b_packets[]
this.in_src_ports[] == return.fileModel.b_nums[]
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == return.normal_alert
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == this.testPH.MM.aggregation_done
this.protocol_udp == this.testPH.stable
this.protocol_udp == this.testPH.fileModel.batch
this.protocol_udp == return.file_filtering
this.protocol_udp == return.MM.aggregation_done
this.protocol_udp == return.stable
this.protocol_udp == return.fileModel.batch
this.protocol_udp == orig(this.MM.aggregation_done)
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == this.testPH.MM.port_trunc_length
this.port_trunc_length == return.port_trunc_length
this.port_trunc_length == return.MM.port_trunc_length
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == return.port_keywords
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == return.port_keywords_range
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == return.MM.smooth_factor
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == this.testPH.anomaly_count
this.min_sample_num == this.testPH.old_anomaly
this.min_sample_num == this.testPH.MM.min_sample
this.min_sample_num == this.testPH.MM.modification_time
this.min_sample_num == this.testPH.fileModel.start
this.min_sample_num == this.testPH.fileModel.end
this.min_sample_num == this.testPH.fileModel.min_sample
this.min_sample_num == this.testPH.fileModel.max_sample
this.min_sample_num == this.testPH.fileModel.totalPacketNum
this.min_sample_num == this.testPH.fileModel.reduceN
this.min_sample_num == this.testPH.packetIndex
this.min_sample_num == return.count

this.min_sample_num == return.anomaly_count
this.min_sample_num == return.old_anomaly
this.min_sample_num == return.MM.min_sample
this.min_sample_num == return.MM.modification_time
this.min_sample_num == return.fileModel.start
this.min_sample_num == return.fileModel.end
this.min_sample_num == return.fileModel.min_sample
this.min_sample_num == return.fileModel.max_sample
this.min_sample_num == return.fileModel.totalPacketNum
this.min_sample_num == return.fileModel.reduceN
this.min_sample_num == return.packetIndex
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == return.MM.max_sample
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == this.testPH.fileModel.sample_rate
this.sample_rate == return.MM.sample_rate
this.sample_rate == return.fileModel.sample_rate
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == return.kernel_h
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == return.smooth_factor
this.fpRate == return.fpRate
this.fpRate == orig(this.fpRate)
this.stablesize == this.testPH.MM.stablesize
this.stablesize == return.MM.stablesize
this.stablesize == return.linkType
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == return.MM.stabledis
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == return.verbose
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == return.string_compare_method
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == return.string_compare_threshold
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == this.testPH.in_dst_ports_set
this.in_dst_ports_set == return.in_dst_ports_set
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == this.testPH.in_src_ports_set
this.in_src_ports_set == return.in_src_ports_set
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == this.testPH.out_dst_ports_set
this.out_dst_ports_set == return.out_dst_ports_set
this.out_dst_ports_set == orig(this.out_dst_ports_set)

this.out_src_ports_set == this.testPH.out_src_ports_set
this.out_src_ports_set == return.out_src_ports_set
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == return
this.testPH.smooth_factor == return.smooth_factor
this.testPH.maxnum == return.maxnum
this.testPH.threshold == return.threshold
this.testPH.temp == return.temp
this.testPH.adjustTCP_count == return.adjustTCP_count
this.testPH.adjustTCP_count[] == return.adjustTCP_count[]
this.testPH.adjustTCP_anomaly_count == return.adjustTCP_anomaly_count
this.testPH.adjustTCP_anomaly_count[] == return.adjustTCP_count[]
this.testPH.adjustUDP_count == return.adjustUDP_count
this.testPH.adjustUDP_count[] == return.adjustTCP_count[]
this.testPH.adjustUDP_anomaly_count == return.adjustUDP_anomaly_count
this.testPH.adjustUDP_anomaly_count[] == return.adjustTCP_count[]
this.testPH.TCP_portready == return.TCP_portready
this.testPH.TCP_portready[] == return.TCP_portready[]
this.testPH.UDP_portready == return.UDP_portready
this.testPH.UDP_portready[] == return.TCP_portready[]
this.testPH.adjustTCP_count_out == return.adjustTCP_count_out
this.testPH.adjustTCP_count_out[] == return.adjustTCP_count[]
this.testPH.adjustTCP_anomaly_count_out == return.adjustTCP_anomaly_count_out
this.testPH.adjustTCP_anomaly_count_out[] == return.adjustTCP_count[]
this.testPH.adjustUDP_count_out == return.adjustUDP_count_out
this.testPH.adjustUDP_count_out[] == return.adjustTCP_count[]
this.testPH.adjustUDP_anomaly_count_out == return.adjustUDP_anomaly_count_out
this.testPH.adjustUDP_anomaly_count_out[] == return.adjustTCP_count[]
this.testPH.TCP_portready_out == return.TCP_portready_out
this.testPH.TCP_portready_out[] == return.TCP_portready[]
this.testPH.UDP_portready_out == return.UDP_portready_out
this.testPH.UDP_portready_out[] == return.TCP_portready[]
this.testPH.adjustTCP_count_in == return.adjustTCP_count_in
this.testPH.adjustTCP_count_in[] == return.adjustTCP_count[]
this.testPH.adjustTCP_anomaly_count_in == return.adjustTCP_anomaly_count_in
this.testPH.adjustTCP_anomaly_count_in[] == return.adjustTCP_count[]
this.testPH.adjustUDP_count_in == return.adjustUDP_count_in
this.testPH.adjustUDP_count_in[] == return.adjustTCP_count[]
this.testPH.adjustUDP_anomaly_count_in == return.adjustUDP_anomaly_count_in
this.testPH.adjustUDP_anomaly_count_in[] == return.adjustTCP_count[]
this.testPH.TCP_portready_in == return.TCP_portready_in
this.testPH.TCP_portready_in[] == return.TCP_portready[]
this.testPH.UDP_portready_in == return.UDP_portready_in
this.testPH.UDP_portready_in[] == return.TCP_portready[]
this.testPH.adjustTCP_count_outsrc == return.adjustTCP_count_outsrc
this.testPH.adjustTCP_count_outsrc[] == return.adjustTCP_count[]
this.testPH.adjustTCP_anomaly_count_outsrc == return.adjustTCP_anomaly_count_outsrc
this.testPH.adjustTCP_anomaly_count_outsrc[] == return.adjustTCP_count[]
this.testPH.adjustUDP_count_outsrc == return.adjustUDP_count_outsrc
this.testPH.adjustUDP_count_outsrc[] == return.adjustTCP_count[]
this.testPH.adjustUDP_anomaly_count_outsrc == return.adjustUDP_anomaly_count_outsrc
this.testPH.adjustUDP_anomaly_count_outsrc[] == return.adjustTCP_count[]
this.testPH.TCP_portready_outsrc == return.TCP_portready_outsrc
this.testPH.TCP_portready_outsrc[] == return.TCP_portready[]
this.testPH.UDP_portready_outsrc == return.UDP_portready_outsrc
this.testPH.UDP_portready_outsrc[] == return.TCP_portready[]
this.testPH.sdf == return.sdf
this.testPH.fileModel == return.fileModel
this.testPH.fileModel.centroids == return.fileModel.centroids
this.testPH.fileModel.centroids[] == return.fileModel.centroids[]
this.testPH.fileModel.NMax == return.fileModel.NMax
this.testPH.fileModel.b_packets == return.fileModel.b_packets

this.testPH.fileModel.b_nums == return.fileModel.b_nums
this.testPH.TCP == return.TCP
this.testPH.UDP == return.UDP
this.testPH.maliciousBuffer == return.maliciousBuffer
this.testPH.sign == return.sign
this.server == orig(this.server)
this.cc == orig(this.cc)
return.adjustTCP_count[] == return.adjustTCP_anomaly_count[]
return.adjustTCP_count[] == return.adjustUDP_count[]
return.adjustTCP_count[] == return.adjustUDP_anomaly_count[]
return.adjustTCP_count[] == return.adjustTCP_count_out[]
return.adjustTCP_count[] == return.adjustTCP_anomaly_count_out[]
return.adjustTCP_count[] == return.adjustUDP_count_out[]
return.adjustTCP_count[] == return.adjustUDP_anomaly_count_out[]
return.adjustTCP_count[] == return.adjustTCP_count_in[]
return.adjustTCP_count[] == return.adjustTCP_anomaly_count_in[]
return.adjustTCP_count[] == return.adjustUDP_count_in[]
return.adjustTCP_count[] == return.adjustUDP_anomaly_count_in[]
return.adjustTCP_count[] == return.adjustTCP_count_outsrc[]
return.adjustTCP_count[] == return.adjustTCP_anomaly_count_outsrc[]
return.adjustTCP_count[] == return.adjustUDP_count_outsrc[]
return.adjustTCP_count[] == return.adjustUDP_anomaly_count_outsrc[]
return.TCP_portready[] == return.UDP_portready[]
return.TCP_portready[] == return.TCP_portready_out[]
return.TCP_portready[] == return.UDP_portready_out[]
return.TCP_portready[] == return.TCP_portready_in[]
return.TCP_portready[] == return.UDP_portready_in[]
return.TCP_portready[] == return.TCP_portready_outsrc[]
return.TCP_portready[] == return.UDP_portready_outsrc[]
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0,

0, 0, 0,

return.fileModel.b_nums has only one value
return.TCP == 6
return.UDP == 17
return.maliciousBuffer has only one value
return.sign == null
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == return.MM.TCP_model.getClass()
this.MM.TCP_model.getClass() == return.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == return.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == return.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == return.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == return.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == return.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == return.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == return.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == return.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == return.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == return.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == return.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())

this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == return.MM.lenProb.getClass()
this.MM.lenProb.getClass() == return.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == return.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == return.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == this.testPH.MM.host_ip.toString
this.host_ip.toString == return.MM.host_ip.toString
this.host_ip.toString == return.host_ip.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
return.TCP_portready[] elements == this.protocol_tcp
this.protocol_tcp in return.TCP_portready[]
return.adjustTCP_count[] elements == this.min_sample_num
this.min_sample_num in return.adjustTCP_count[]
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == return.string_compare_method.toString
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
Payload.init():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.FILTER == this.textmodelfile
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value

this.offlineTrainFile.toString == ”train-p2.dmp”
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == ”test-p3.dmp”
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.init():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.MM.TCP_model[] == this.MM.UDP_model[]
this.MM.TCP_model[] == this.MM.TCP_model_in[]
this.MM.TCP_model[] == this.MM.UDP_model_in[]
this.MM.TCP_model[] == this.MM.TCP_model_out[]
this.MM.TCP_model[] == this.MM.UDP_model_out[]
this.MM.TCP_model[] == this.MM.TCP_model_outsrc[]
this.MM.TCP_model[] == this.MM.UDP_model_outsrc[]
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_in
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_out
this.MM.TCP_lengthModel == this.MM.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel == this.MM.UDP_lengthModel_outsrc
this.MM.modification_time == this.min_sample_num
this.MM.lenProb[] == this.MM.lenProb_out[]
this.MM.lenProb[] == this.MM.lenProb_in[]
this.MM.lenProb[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold[] == this.MM.anomalyLengthThreshold_outsrc[]
this.MM.UDPNormalLength == this.MM.UDPNormalLength_out
this.MM.UDPNormalLength == this.MM.UDPNormalLength_in
this.MM.UDPNormalLength == this.MM.UDPNormalLength_outsrc
this.config_device == orig(this.config_device)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG

this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.FILTER)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] contains only nulls and has only one value, of length 1024
this.MM.TCP_model[] elements == null
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

arg0[].toString == [test, +g, globalFile.txt
]
arg0[].toString elements one of "+g", "globalFile.txt", "test"
==========
Payload.main(java.lang.String[]):::EXIT
arg0[] == orig(arg0[])
arg0[] contains no nulls and has only one value, of length 3
arg0[].toString == [test, +g, globalFile.txt
]
arg0[].toString elements one of "+g", "globalFile.txt", "test"
==========
Payload.portsString(java.util.Vector, java.lang.String):::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.FILTER == this.textmodelfile
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
arg1 has only one value
arg1.toString == "dst"
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.portsString(java.util.Vector, java.lang.String):::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == this.textmodelfile
this.FILTER == orig(this.FILTER)
this.FILTER == orig(this.textmodelfile)

this.MM == orig(this.MM)
this.config_device == orig(this.config_device)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)

this.server == orig(this.server)
this.cc == orig(this.cc)
arg0[] == orig(arg0[])
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
arg0[].getClass() == [java.lang.String]
arg0[].getClass() elements == java.lang.String.class
arg1.toString == "dst"
return.toString == "(dst port 80)"
this.FILTER.toString == this.textmodelfile.toString
this.FILTER.toString == orig(this.FILTER.toString)
this.FILTER.toString == orig(this.textmodelfile.toString)
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
arg1.toString == orig(arg1.toString)
==========
Payload.readGlobalFile(java.lang.String):::ENTER
this.PACKET_COUNT == this.TEST_PACKET_COUNT
this.FILTER == this.offlineTrainFile
this.FILTER == this.offlineTestFile
this.FILTER == this.textmodelfile
this.in_dst_ports[] == this.in_src_ports[]
this.in_dst_ports[] == this.out_dst_ports[]
this.in_dst_ports[] == this.out_src_ports[]
this.in_dst_ports[] == this.comm_ips[]
this.stablesize == this.verbose
this.PACKET_COUNT == 1000
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "localhost"
this.in_dst_ports[] == []
this.in_dst_ports[].getClass() == []

this.min_sample_num == 20
this.max_sample_num == 200
this.sample_rate == 0.05
this.threshold_step == 0.1
this.fpRate == 0.01
this.cluster_threshold == 0.3
this.modelfile has only one value
this.string_compare_method has only one value
this.communication_time == 0
this.testPH == null
arg0 has only one value
arg0.toString == "globalFile.txt"
this.FILTER.toString == this.offlineTrainFile.toString
this.FILTER.toString == this.offlineTestFile.toString
this.FILTER.toString == this.textmodelfile.toString
==========
Payload.readGlobalFile(java.lang.String):::EXIT
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == this.textmodelfile
this.FILTER == orig(this.FILTER)
this.FILTER == orig(this.offlineTrainFile)
this.FILTER == orig(this.offlineTestFile)
this.FILTER == orig(this.textmodelfile)
this.MM == orig(this.MM)
this.config_device == orig(this.config_device)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == orig(this.in_dst_ports[])
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_dst_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.in_src_ports[] == orig(this.comm_ips[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_src_ports == orig(this.out_src_ports)
this.min_sample_num == orig(this.communication_time)
this.max_sample_num == orig(this.PACKET_COUNT)
this.max_sample_num == orig(this.TEST_PACKET_COUNT)
this.stablesize == orig(this.stablesize)
this.stablesize == orig(this.verbose)
this.comm_ips == orig(this.comm_ips)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.testPH == orig(this.testPH)
this.server == orig(this.server)
this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == ""
this.MM == null
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0

this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
arg0.toString == "globalFile.txt"
this.FILTER.toString == this.textmodelfile.toString
this.FILTER.toString == orig(this.FILTER.toString)
this.FILTER.toString == orig(this.offlineTrainFile.toString)
this.FILTER.toString == orig(this.offlineTestFile.toString)
this.FILTER.toString == orig(this.textmodelfile.toString)
this.config_device.toString == orig(this.config_device.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
arg0.toString == orig(arg0.toString)
==========
Payload.startTest():::ENTER
this.TEST_PACKET_COUNT == this.trunc_length
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.modification_time == this.min_sample_num
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.threshold_step == this.cluster_threshold
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

null, null, null,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
this.MM.lenProb_out has only one value
this.MM.anomalyLengthThreshold_out has only one value
this.MM.lenProb_in has only one value
this.MM.anomalyLengthThreshold_in has only one value
this.MM.lenProb_outsrc has only one value
this.MM.anomalyLengthThreshold_outsrc has only one value
this.MM.UDPNormalLength has only one value
this.MM.UDPNormalLength_out has only one value
this.MM.UDPNormalLength_in has only one value
this.MM.UDPNormalLength_outsrc has only one value
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports[] contains no nulls and has only one value, of length 1
this.in_dst_ports[].getClass() == [java.lang.String]
this.out_dst_ports[] contains no nulls and has only one value, of length 1
this.protocol_udp == false
this.min_sample_num == 0
this.max_sample_num == 1000
this.sample_rate == 0.0
this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH == null
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
==========
Payload.startTest():::EXIT
this.PACKET_COUNT == orig(this.PACKET_COUNT)
this.TEST_PACKET_COUNT == this.trunc_length
this.TEST_PACKET_COUNT == this.testPH.MM.trunc_length
this.TEST_PACKET_COUNT == orig(this.TEST_PACKET_COUNT)

this.TEST_PACKET_COUNT == orig(this.MM.trunc_length)
this.TEST_PACKET_COUNT == orig(this.trunc_length)
this.FILTER == orig(this.FILTER)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == this.testPH.MM.TCP_model[]
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_in[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_out[]
this.MM.UDP_model[] == this.testPH.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.testPH.MM.UDPNormalLength_outsrc[]
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == this.testPH.MM.TCP_lengthModel[]
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.testPH.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.testPH.MM.UDP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)

this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.modification_time == this.min_sample_num
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == this.testPH.MM.lenProb[]
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == this.testPH.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == this.testPH.MM.lenProb_out[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_in[]
this.MM.lenProb_out[] == this.testPH.MM.lenProb_outsrc[]
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_out[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.testPH.MM.anomalyLengthThreshold_outsrc[]
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.config_device == orig(this.config_device)
this.host_ip == this.testPH.MM.host_ip
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports == orig(this.in_dst_ports)
this.in_dst_ports[] == orig(this.in_dst_ports[])
this.in_src_ports == orig(this.in_src_ports)
this.in_src_ports[] == this.testPH.fileModel.b_nums[]
this.in_src_ports[] == orig(this.in_src_ports[])
this.in_src_ports[] == orig(this.out_src_ports[])
this.out_dst_ports == orig(this.out_dst_ports)
this.out_dst_ports[] == orig(this.out_dst_ports[])
this.out_src_ports == orig(this.out_src_ports)
this.protocol_tcp == this.auto_adjust
this.protocol_tcp == this.normal_alert
this.protocol_tcp == this.OFFLINE_TRAIN
this.protocol_tcp == this.OFFLINE_TEST
this.protocol_tcp == this.testPH.stable
this.protocol_tcp == orig(this.protocol_tcp)
this.protocol_tcp == orig(this.auto_adjust)
this.protocol_tcp == orig(this.print)
this.protocol_tcp == orig(this.normal_alert)
this.protocol_tcp == orig(this.OFFLINE_TRAIN)
this.protocol_tcp == orig(this.OFFLINE_TEST)
this.protocol_tcp == orig(this.auto_correlation)
this.protocol_udp == this.file_filtering
this.protocol_udp == this.EPOCH_TRAIN_FLAG
this.protocol_udp == this.readtextmodel
this.protocol_udp == this.communication_on
this.protocol_udp == this.testPH.MM.aggregation_done
this.protocol_udp == this.testPH.fileModel.batch
this.protocol_udp == orig(this.MM.aggregation_done)

this.protocol_udp == orig(this.protocol_udp)
this.protocol_udp == orig(this.file_filtering)
this.protocol_udp == orig(this.EPOCH_TRAIN_FLAG)
this.protocol_udp == orig(this.readtextmodel)
this.protocol_udp == orig(this.communication_on)
this.port_trunc_length == this.testPH.MM.port_trunc_length
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_trunc_length == orig(this.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.smooth_factor == orig(this.MM.smooth_factor)
this.smooth_factor == orig(this.smooth_factor)
this.min_sample_num == this.testPH.anomaly_count
this.min_sample_num == this.testPH.old_anomaly
this.min_sample_num == this.testPH.MM.min_sample
this.min_sample_num == this.testPH.MM.modification_time
this.min_sample_num == this.testPH.fileModel.start
this.min_sample_num == this.testPH.fileModel.end
this.min_sample_num == this.testPH.fileModel.min_sample
this.min_sample_num == this.testPH.fileModel.max_sample
this.min_sample_num == this.testPH.fileModel.totalPacketNum
this.min_sample_num == this.testPH.fileModel.reduceN
this.min_sample_num == orig(this.MM.min_sample)
this.min_sample_num == orig(this.MM.modification_time)
this.min_sample_num == orig(this.min_sample_num)
this.max_sample_num == orig(this.MM.max_sample)
this.max_sample_num == orig(this.max_sample_num)
this.sample_rate == this.testPH.fileModel.sample_rate
this.sample_rate == orig(this.MM.sample_rate)
this.sample_rate == orig(this.sample_rate)
this.kernel_h == orig(this.kernel_h)
this.normal_length_perc == orig(this.normal_length_perc)
this.threshold_step == this.cluster_threshold
this.threshold_step == orig(this.threshold_step)
this.threshold_step == orig(this.cluster_threshold)
this.fpRate == orig(this.fpRate)
this.stablesize == this.testPH.MM.stablesize
this.stablesize == orig(this.MM.stablesize)
this.stablesize == orig(this.stablesize)
this.stabledis == orig(this.MM.stabledis)
this.stabledis == orig(this.stabledis)
this.modelfile == orig(this.modelfile)
this.verbose == orig(this.verbose)
this.offlineTrainFile == orig(this.offlineTrainFile)
this.TRAIN_HOUR == orig(this.TRAIN_HOUR)
this.STABLE_PORTS_PERC == orig(this.STABLE_PORTS_PERC)
this.offlineTestFile == orig(this.offlineTestFile)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.textmodelfile == orig(this.textmodelfile)
this.communication_time == orig(this.communication_time)
this.comm_ips == orig(this.comm_ips)
this.comm_ips[] == orig(this.comm_ips[])
this.in_dst_ports_set == this.testPH.in_dst_ports_set
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == this.testPH.in_src_ports_set
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == this.testPH.out_dst_ports_set
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == this.testPH.out_src_ports_set
this.out_src_ports_set == orig(this.out_src_ports_set)
this.trainPH == orig(this.trainPH)
this.server == orig(this.server)

this.cc == orig(this.cc)
this.PACKET_COUNT == 5000
this.TEST_PACKET_COUNT == -1
this.FILTER has only one value
this.FILTER.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst port 80))))"
this.MM has only one value
this.MM.TCP_model has only one value
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null,

this.MM.UDP_lengthModel_in has only one value
this.MM.TCP_lengthModel_out has only one value
this.MM.UDP_lengthModel_out has only one value
this.MM.TCP_lengthModel_outsrc has only one value
this.MM.UDP_lengthModel_outsrc has only one value
this.MM.lenProb has only one value
this.MM.lenProb[] has only one value, of length 1024
this.MM.anomalyLengthThreshold has only one value
this.MM.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

this.threshold_step == 0.2
this.fpRate == 0.0010
this.modelfile has only one value
this.verbose == 2
this.offlineTrainFile has only one value
this.offlineTrainFile.toString == "train-p2.dmp"
this.TRAIN_HOUR == 0.02
this.offlineTestFile has only one value
this.offlineTestFile.toString == "test-p3.dmp"
this.string_compare_method has only one value
this.communication_time == 1800
this.comm_ips[] contains no nulls and has only one value, of length 2
this.comm_ips[].getClass() == [java.lang.String, java.lang.String]
this.testPH has only one value
this.testPH.threshold == 409.0
this.testPH.packetIndex == 299
this.FILTER.toString == orig(this.FILTER.toString)
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.TCP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.testPH.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.testPH.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.testPH.MM.lenProb_outsrc.getClass()
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())

this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.config_device.toString == orig(this.config_device.toString)
this.host_ip.toString == this.testPH.MM.host_ip.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.modelfile.toString == orig(this.modelfile.toString)
this.offlineTrainFile.toString == orig(this.offlineTrainFile.toString)
this.offlineTestFile.toString == orig(this.offlineTestFile.toString)
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.textmodelfile.toString == orig(this.textmodelfile.toString)
==========
TestPacketHandler:::OBJECT
this.count == this.anomaly_count
this.count == this.old_anomaly
this.count == this.MM.min_sample
this.count == this.MM.modification_time
this.count == this.fileModel.start
this.count == this.fileModel.end
this.count == this.fileModel.min_sample
this.count == this.fileModel.max_sample
this.count == this.fileModel.totalPacketNum
this.count == this.fileModel.reduceN
this.trunc_length == this.MM.trunc_length
this.file_filtering == this.MM.aggregation_done
this.file_filtering == this.stable
this.file_filtering == this.fileModel.batch
this.MM.UDP_model[] == this.MM.TCP_model_in[]
this.MM.UDP_model[] == this.MM.UDP_model_in[]
this.MM.UDP_model[] == this.MM.TCP_model_out[]
this.MM.UDP_model[] == this.MM.UDP_model_out[]
this.MM.UDP_model[] == this.MM.TCP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDP_model_outsrc[]
this.MM.UDP_model[] == this.MM.UDPNormalLength[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_out[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_in[]
this.MM.UDP_model[] == this.MM.UDPNormalLength_outsrc[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_in[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_out[]
this.MM.UDP_lengthModel[] == this.MM.TCP_lengthModel_outsrc[]
this.MM.UDP_lengthModel[] == this.MM.UDP_lengthModel_outsrc[]
this.MM.sample_rate == this.fileModel.sample_rate
this.MM.stablesize == this.linkType
this.MM.lenProb_out[] == this.MM.lenProb_in[]
this.MM.lenProb_out[] == this.MM.lenProb_outsrc[]
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_in[]
this.MM.anomalyLengthThreshold_out[] == this.MM.anomalyLengthThreshold_outsrc[]
this.adjustUDP_count[] == this.adjustUDP_anomaly_count[]
this.adjustUDP_count[] == this.adjustTCP_count_out[]
this.adjustUDP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustUDP_count[] == this.adjustUDP_count_out[]
this.adjustUDP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustUDP_count[] == this.adjustTCP_count_in[]
this.adjustUDP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustUDP_count[] == this.adjustUDP_count_in[]
this.adjustUDP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustUDP_count[] == this.adjustTCP_count_outsrc[]
this.adjustUDP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustUDP_count[] == this.adjustUDP_count_outsrc[]
this.adjustUDP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.UDP_portready[] == this.TCP_portready_out[]

this.UDP_portready[] == this.UDP_portready_out[]
this.UDP_portready[] == this.TCP_portready_in[]
this.UDP_portready[] == this.UDP_portready_in[]
this.UDP_portready[] == this.TCP_portready_outsrc[]
this.UDP_portready[] == this.UDP_portready_outsrc[]
this.fileModel.b_packets[] == this.fileModel.b_nums[]
this has only one value
this.count == 0
this.trunc_length == -1
this.file_filtering == false
this.smooth_factor == 0.0010000000474974513
this.maxnum == 1999.999905005102
this.normal_alert == true
this.verbose == 2
this.temp == null
this.MM has only one value
this.MM != null
this.MM.TCP_model has only one value
this.MM.TCP_model != null
this.MM.TCP_model.getClass() == java.util.HashMap[].class
this.MM.TCP_model[] has only one value, of length 1024
this.MM.TCP_model[].getClass() == [null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null,
null, null,
null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, java.util.HashMap, null, null, null, null,
null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null, null

null, null, null,

this.MM.TCP_lengthModel != null
this.MM.TCP_lengthModel.getClass() == java.util.Vector[].class
this.MM.TCP_lengthModel[] has only one value, of length 1024
this.MM.TCP_lengthModel[].getClass() == [null, null, null,

this.MM.port_trunc_length has only one value
this.MM.port_trunc_length != null
this.MM.max_sample == 1000
this.MM.sample_rate == 0.0
this.MM.host_ip has only one value
this.MM.host_ip != null
this.MM.host_ip.toString == "128.59.16.101"
this.MM.stabledis == 0.05
this.MM.smooth_factor == 0.0010
this.MM.lenProb has only one value
this.MM.lenProb != null
this.MM.lenProb.getClass() == double[][].class
this.MM.lenProb[] has only one value, of length 1024
this.MM.anomalyLengthThreshold has only one value
this.MM.anomalyLengthThreshold != null
this.MM.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

this.adjustUDP_count != null
this.adjustUDP_count[] == [0, 0, 0,

1, 1, 1,

this.TCP_portready_outsrc has only one value
this.TCP_portready_outsrc != null
this.UDP_portready_outsrc has only one value
this.UDP_portready_outsrc != null
this.sdf has only one value
this.sdf != null
this.fileModel has only one value
this.fileModel != null
this.fileModel.centroids has only one value
this.fileModel.centroids != null
this.fileModel.centroids[] elements != null
this.fileModel.centroids[].getClass() elements == OneCentroid.class
this.fileModel.centroids[].getClass() one of [], [OneCentroid, OneCentroid]
this.fileModel.NMax == 10
this.fileModel.b_packets has only one value
this.fileModel.b_packets != null
this.fileModel.b_packets[] == []
this.fileModel.b_packets[].getClass() == []
this.fileModel.b_nums has only one value
this.fileModel.b_nums != null
this.packetIndex >= 0
this.TCP == 6
this.UDP == 17
this.maliciousBuffer has only one value
this.maliciousBuffer != null
this.sign == null
this.linkType == 1
this.adjustTCP_count[] elements >= this.count
this.count in this.adjustTCP_count[]
this.adjustTCP_anomaly_count[] elements >= this.count
this.count in this.adjustTCP_anomaly_count[]
this.adjustUDP_count[] elements == this.count
this.count in this.adjustUDP_count[]
this.count <= this.packetIndex
this.adjustTCP_count[] elements > this.trunc_length
this.adjustTCP_anomaly_count[] elements > this.trunc_length
this.trunc_length < this.packetIndex
this.smooth_factor < this.threshold
this.MM.anomalyLengthThreshold[] elements < this.smooth_factor
this.smooth_factor <= this.fpRate
this.maxnum > this.threshold
this.MM.anomalyLengthThreshold[] elements < this.maxnum
this.maxnum > this.fpRate
this.threshold > this.MM.stabledis
this.MM.anomalyLengthThreshold[] elements < this.threshold
this.threshold > this.MM.normal_length_perc
this.threshold > this.kernel_h
this.threshold
this.threshold > this.string_compare_threshold
this.threshold > this.fpRate
this.normal_alert in this.TCP_portready[]
this.UDP_portready[] elements == this.normal_alert
this.normal_alert in this.UDP_portready[]
this.MM.TCP_model.getClass() == this.MM.UDP_model.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_in.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDP_model_out.getClass()
this.MM.TCP_model.getClass() == this.MM.TCP_model_outsrc.getClass()

this.MM.TCP_model.getClass() == this.MM.UDP_model_outsrc.getClass()
this.MM.TCP_model.getClass() != this.MM.TCP_lengthModel.getClass()
this.MM.TCP_model.getClass() != this.MM.lenProb.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_out.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_in.getClass()
this.MM.TCP_model.getClass() == this.MM.UDPNormalLength_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_in.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_out.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.TCP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() == this.MM.UDP_lengthModel_outsrc.getClass()
this.MM.TCP_lengthModel.getClass() != this.MM.lenProb.getClass()
this.adjustTCP_count[] elements < this.MM.max_sample
this.adjustTCP_anomaly_count[] elements < this.MM.max_sample
this.MM.max_sample > this.packetIndex
this.MM.host_ip.toString < this.string_compare_method.toString
this.MM.host_ip.toString >= this.host_ip.toString
this.MM.anomalyLengthThreshold[] elements < this.MM.stabledis
this.MM.stabledis > this.fpRate
this.MM.lenProb.getClass() == this.MM.lenProb_out.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_in.getClass()
this.MM.lenProb.getClass() == this.MM.lenProb_outsrc.getClass()
this.MM.anomalyLengthThreshold[] >= this.MM.anomalyLengthThreshold_out[] (elementwise)
this.MM.anomalyLengthThreshold[] elements >= this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold[]
this.MM.anomalyLengthThreshold[] elements < this.kernel_h
this.MM.anomalyLengthThreshold[] elements < this.string_compare_threshold
this.MM.anomalyLengthThreshold[] elements < this.fpRate
this.MM.anomalyLengthThreshold_out[] elements == this.MM.normal_length_perc
this.MM.normal_length_perc in this.MM.anomalyLengthThreshold_out[]
this.MM.normal_length_perc < this.fpRate
this.kernel_h > this.fpRate
this.string_compare_method.toString > this.host_ip.toString
this.string_compare_threshold > this.fpRate
this.adjustTCP_count[] >= this.adjustTCP_anomaly_count[] (elementwise)
this.adjustTCP_count[] >= this.adjustUDP_count[] (elementwise)
this.adjustTCP_count[] elements <= this.packetIndex
this.adjustTCP_anomaly_count[] >= this.adjustUDP_count[] (elementwise)
this.adjustTCP_anomaly_count[] elements <= this.packetIndex
==========
TestPacketHandler.TestPacketHandler(Models, int, double, double, double, boolean, int):::ENTER
arg0.UDP_model[] == arg0.TCP_model_in[]
arg0.UDP_model[] == arg0.UDP_model_in[]
arg0.UDP_model[] == arg0.TCP_model_out[]
arg0.UDP_model[] == arg0.UDP_model_out[]
arg0.UDP_model[] == arg0.TCP_model_outsrc[]
arg0.UDP_model[] == arg0.UDP_model_outsrc[]
arg0.UDP_model[] == arg0.UDPNormalLength[]
arg0.UDP_model[] == arg0.UDPNormalLength_out[]
arg0.UDP_model[] == arg0.UDPNormalLength_in[]
arg0.UDP_model[] == arg0.UDPNormalLength_outsrc[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_outsrc[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_outsrc[]
arg0.trunc_length == arg1
arg0.min_sample == arg0.modification_time
arg0.lenProb_out[] == arg0.lenProb_in[]

arg0.lenProb_out[] == arg0.lenProb_outsrc[]
arg0 has only one value
arg0.TCP_model has only one value
arg0.TCP_model.getClass() == java.util.HashMap[].class
arg0.TCP_model[] has only one value, of length 1024
arg0.TCP_model[].getClass() == [null, null, null,

null, null, null,

null, null, null,

0.0, 0.0, 0.0,

arg0.lenProb_outsrc has only one value
arg0.anomalyLengthThreshold_outsrc has only one value
arg0.UDPNormalLength has only one value
arg0.UDPNormalLength_out has only one value
arg0.UDPNormalLength_in has only one value
arg0.UDPNormalLength_outsrc has only one value
arg0.normal_length_perc == 0.0
arg1 == -1
arg2 == 0.0010000000474974513
arg3 == 5.0
arg4 == 0.2
arg5 == true
arg6 == 2
arg0.TCP_model.getClass() == arg0.UDP_model.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_out.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_in.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_outsrc.getClass()
arg0.lenProb.getClass() == arg0.lenProb_out.getClass()
arg0.lenProb.getClass() == arg0.lenProb_in.getClass()
arg0.lenProb.getClass() == arg0.lenProb_outsrc.getClass()
arg0.normal_length_perc in arg0.anomalyLengthThreshold[]
arg0.anomalyLengthThreshold_out[] elements == arg0.normal_length_perc
arg0.normal_length_perc in arg0.anomalyLengthThreshold_out[]
==========
TestPacketHandler.TestPacketHandler(Models, int, double, double, double, boolean, int):::EXIT
this.count == this.packetIndex
this.count == arg0.min_sample
this.count == arg0.modification_time
this.count == orig(arg0.min_sample)
this.count == orig(arg0.modification_time)
this.trunc_length == arg0.trunc_length
this.trunc_length == orig(arg0.trunc_length)
this.trunc_length == orig(arg1)
this.port_trunc_length == this.port_keywords
this.port_trunc_length == this.port_keywords_range
this.file_filtering == arg0.aggregation_done
this.file_filtering == orig(arg0.aggregation_done)
this.smooth_factor == orig(arg2)
this.normal_alert == orig(arg5)
this.verbose == orig(arg6)
this.MM == orig(arg0)
this.MM.TCP_model == arg0.TCP_model
this.MM.TCP_model[] == arg0.TCP_model[]
this.MM.UDP_model == arg0.UDP_model
this.MM.UDP_model[] == arg0.UDP_model[]
this.MM.TCP_model_in == arg0.TCP_model_in
this.MM.TCP_model_in[] == arg0.UDP_model[]
this.MM.UDP_model_in == arg0.UDP_model_in
this.MM.UDP_model_in[] == arg0.UDP_model[]

this.MM.TCP_model_out == arg0.TCP_model_out
this.MM.TCP_model_out[] == arg0.UDP_model[]
this.MM.UDP_model_out == arg0.UDP_model_out
this.MM.UDP_model_out[] == arg0.UDP_model[]
this.MM.TCP_model_outsrc == arg0.TCP_model_outsrc
this.MM.TCP_model_outsrc[] == arg0.UDP_model[]
this.MM.UDP_model_outsrc == arg0.UDP_model_outsrc
this.MM.UDP_model_outsrc[] == arg0.UDP_model[]
this.MM.TCP_lengthModel == arg0.TCP_lengthModel
this.MM.TCP_lengthModel[] == arg0.TCP_lengthModel[]
this.MM.UDP_lengthModel == arg0.UDP_lengthModel
this.MM.UDP_lengthModel[] == arg0.UDP_lengthModel[]
this.MM.TCP_lengthModel_in == arg0.TCP_lengthModel_in
this.MM.TCP_lengthModel_in[] == arg0.UDP_lengthModel[]
this.MM.UDP_lengthModel_in == arg0.UDP_lengthModel_in
this.MM.UDP_lengthModel_in[] == arg0.UDP_lengthModel[]
this.MM.TCP_lengthModel_out == arg0.TCP_lengthModel_out
this.MM.TCP_lengthModel_out[] == arg0.UDP_lengthModel[]
this.MM.UDP_lengthModel_out == arg0.UDP_lengthModel_out
this.MM.UDP_lengthModel_out[] == arg0.UDP_lengthModel[]
this.MM.TCP_lengthModel_outsrc == arg0.TCP_lengthModel_outsrc
this.MM.TCP_lengthModel_outsrc[] == arg0.UDP_lengthModel[]
this.MM.UDP_lengthModel_outsrc == arg0.UDP_lengthModel_outsrc
this.MM.UDP_lengthModel_outsrc[] == arg0.UDP_lengthModel[]
this.MM.port_trunc_length == arg0.port_trunc_length
this.MM.max_sample == arg0.max_sample
this.MM.sample_rate == arg0.sample_rate
this.MM.host_ip == arg0.host_ip
this.MM.stabledis == arg0.stabledis
this.MM.smooth_factor == arg0.smooth_factor
this.MM.lenProb == arg0.lenProb
this.MM.lenProb[] == arg0.lenProb[]
this.MM.anomalyLengthThreshold == arg0.anomalyLengthThreshold
this.MM.lenProb_out == arg0.lenProb_out
this.MM.lenProb_out[] == arg0.lenProb_out[]
this.MM.anomalyLengthThreshold_out == arg0.anomalyLengthThreshold_out
this.MM.lenProb_in == arg0.lenProb_in
this.MM.lenProb_in[] == arg0.lenProb_out[]
this.MM.anomalyLengthThreshold_in == arg0.anomalyLengthThreshold_in
this.MM.lenProb_outsrc == arg0.lenProb_outsrc
this.MM.lenProb_outsrc[] == arg0.lenProb_out[]
this.MM.anomalyLengthThreshold_outsrc == arg0.anomalyLengthThreshold_outsrc
this.MM.UDPNormalLength == arg0.UDPNormalLength
this.MM.UDPNormalLength[] == arg0.UDP_model[]
this.MM.UDPNormalLength_out == arg0.UDPNormalLength_out
this.MM.UDPNormalLength_out[] == arg0.UDP_model[]
this.MM.UDPNormalLength_in == arg0.UDPNormalLength_in
this.MM.UDPNormalLength_in[] == arg0.UDP_model[]
this.MM.UDPNormalLength_outsrc == arg0.UDPNormalLength_outsrc
this.MM.UDPNormalLength_outsrc[] == arg0.UDP_model[]
this.MM.normal_length_perc == arg0.normal_length_perc
this.kernel_h == orig(arg3)
this.in_dst_ports_set == this.in_src_ports_set
this.in_dst_ports_set == this.out_dst_ports_set
this.in_dst_ports_set == this.out_src_ports_set
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]

this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.fileModel.sample_rate == arg0.sample_rate
this.linkType == arg0.stablesize
this.linkType == orig(arg0.stablesize)
arg0.TCP_model == orig(arg0.TCP_model)
arg0.TCP_model[] == orig(arg0.TCP_model[])
arg0.UDP_model == orig(arg0.UDP_model)
arg0.UDP_model[] == arg0.TCP_model_in[]
arg0.UDP_model[] == arg0.UDP_model_in[]
arg0.UDP_model[] == arg0.TCP_model_out[]
arg0.UDP_model[] == arg0.UDP_model_out[]
arg0.UDP_model[] == arg0.TCP_model_outsrc[]
arg0.UDP_model[] == arg0.UDP_model_outsrc[]
arg0.UDP_model[] == arg0.UDPNormalLength[]
arg0.UDP_model[] == arg0.UDPNormalLength_out[]
arg0.UDP_model[] == arg0.UDPNormalLength_in[]
arg0.UDP_model[] == arg0.UDPNormalLength_outsrc[]
arg0.UDP_model[] == orig(arg0.UDP_model[])
arg0.UDP_model[] == orig(arg0.TCP_model_in[])
arg0.UDP_model[] == orig(arg0.UDP_model_in[])
arg0.UDP_model[] == orig(arg0.TCP_model_out[])
arg0.UDP_model[] == orig(arg0.UDP_model_out[])
arg0.UDP_model[] == orig(arg0.TCP_model_outsrc[])
arg0.UDP_model[] == orig(arg0.UDP_model_outsrc[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength_out[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength_in[])
arg0.UDP_model[] == orig(arg0.UDPNormalLength_outsrc[])
arg0.TCP_model_in == orig(arg0.TCP_model_in)
arg0.UDP_model_in == orig(arg0.UDP_model_in)
arg0.TCP_model_out == orig(arg0.TCP_model_out)
arg0.UDP_model_out == orig(arg0.UDP_model_out)
arg0.TCP_model_outsrc == orig(arg0.TCP_model_outsrc)
arg0.UDP_model_outsrc == orig(arg0.UDP_model_outsrc)
arg0.TCP_lengthModel == orig(arg0.TCP_lengthModel)
arg0.TCP_lengthModel[] == orig(arg0.TCP_lengthModel[])
arg0.UDP_lengthModel == orig(arg0.UDP_lengthModel)
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_in[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_out[]
arg0.UDP_lengthModel[] == arg0.TCP_lengthModel_outsrc[]
arg0.UDP_lengthModel[] == arg0.UDP_lengthModel_outsrc[]
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel[])
arg0.UDP_lengthModel[] == orig(arg0.TCP_lengthModel_in[])
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel_in[])
arg0.UDP_lengthModel[] == orig(arg0.TCP_lengthModel_out[])
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel_out[])
arg0.UDP_lengthModel[] == orig(arg0.TCP_lengthModel_outsrc[])
arg0.UDP_lengthModel[] == orig(arg0.UDP_lengthModel_outsrc[])

arg0.TCP_lengthModel_in == orig(arg0.TCP_lengthModel_in)
arg0.UDP_lengthModel_in == orig(arg0.UDP_lengthModel_in)
arg0.TCP_lengthModel_out == orig(arg0.TCP_lengthModel_out)
arg0.UDP_lengthModel_out == orig(arg0.UDP_lengthModel_out)
arg0.TCP_lengthModel_outsrc == orig(arg0.TCP_lengthModel_outsrc)
arg0.UDP_lengthModel_outsrc == orig(arg0.UDP_lengthModel_outsrc)
arg0.port_trunc_length == orig(arg0.port_trunc_length)
arg0.max_sample == orig(arg0.max_sample)
arg0.sample_rate == orig(arg0.sample_rate)
arg0.host_ip == orig(arg0.host_ip)
arg0.stabledis == orig(arg0.stabledis)
arg0.smooth_factor == orig(arg0.smooth_factor)
arg0.lenProb == orig(arg0.lenProb)
arg0.lenProb[] == orig(arg0.lenProb[])
arg0.anomalyLengthThreshold == orig(arg0.anomalyLengthThreshold)
arg0.lenProb_out == orig(arg0.lenProb_out)
arg0.lenProb_out[] == arg0.lenProb_in[]
arg0.lenProb_out[] == arg0.lenProb_outsrc[]
arg0.lenProb_out[] == orig(arg0.lenProb_out[])
arg0.lenProb_out[] == orig(arg0.lenProb_in[])
arg0.lenProb_out[] == orig(arg0.lenProb_outsrc[])
arg0.anomalyLengthThreshold_out == orig(arg0.anomalyLengthThreshold_out)
arg0.lenProb_in == orig(arg0.lenProb_in)
arg0.anomalyLengthThreshold_in == orig(arg0.anomalyLengthThreshold_in)
arg0.lenProb_outsrc == orig(arg0.lenProb_outsrc)
arg0.anomalyLengthThreshold_outsrc == orig(arg0.anomalyLengthThreshold_outsrc)
arg0.UDPNormalLength == orig(arg0.UDPNormalLength)
arg0.UDPNormalLength_out == orig(arg0.UDPNormalLength_out)
arg0.UDPNormalLength_in == orig(arg0.UDPNormalLength_in)
arg0.UDPNormalLength_outsrc == orig(arg0.UDPNormalLength_outsrc)
arg0.normal_length_perc == orig(arg0.normal_length_perc)
this.port_trunc_length == null
this.threshold == 64.0
this.string_compare_method has only one value
this.fpRate == 0.01
this.host_ip has only one value
this.host_ip.toString == ""
this.in_dst_ports_set == null
this.adjustTCP_count[] == [0, 0, 0,

1, 1, 1,

null, null, null,

arg0.TCP_lengthModel.getClass() == java.util.Vector[].class
arg0.TCP_lengthModel[] has only one value, of length 1024
arg0.TCP_lengthModel[].getClass() == [null, null, null,

arg0.lenProb.getClass() == double[][].class
arg0.lenProb[] has only one value, of length 1024
arg0.anomalyLengthThreshold has only one value
arg0.anomalyLengthThreshold[] == [0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

arg0.anomalyLengthThreshold_out[] elements == 0.0
arg0.lenProb_in has only one value
arg0.anomalyLengthThreshold_in has only one value
arg0.lenProb_outsrc has only one value
arg0.anomalyLengthThreshold_outsrc has only one value
arg0.UDPNormalLength has only one value
arg0.UDPNormalLength_out has only one value
arg0.UDPNormalLength_in has only one value
arg0.UDPNormalLength_outsrc has only one value
arg0.normal_length_perc == 0.0
this.adjustTCP_count[] elements == this.count
this.port_trunc_length in arg0.TCP_model[]
arg0.UDP_model[] elements == this.port_trunc_length
this.port_trunc_length in arg0.UDP_model[]
this.TCP_portready[] elements == this.normal_alert
this.MM.TCP_model.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_model_in.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model_in.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_model_out.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model_out.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_model_outsrc.getClass() == arg0.TCP_model.getClass()
this.MM.UDP_model_outsrc.getClass() == arg0.TCP_model.getClass()
this.MM.TCP_lengthModel.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel_in.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel_in.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel_out.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel_out.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.TCP_lengthModel_outsrc.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.UDP_lengthModel_outsrc.getClass() == arg0.TCP_lengthModel.getClass()
this.MM.host_ip.toString == arg0.host_ip.toString
this.MM.lenProb.getClass() == arg0.lenProb.getClass()
this.MM.lenProb_out.getClass() == arg0.lenProb.getClass()
this.MM.lenProb_in.getClass() == arg0.lenProb.getClass()
this.MM.lenProb_outsrc.getClass() == arg0.lenProb.getClass()
this.MM.UDPNormalLength.getClass() == arg0.TCP_model.getClass()
this.MM.UDPNormalLength_out.getClass() == arg0.TCP_model.getClass()
this.MM.UDPNormalLength_in.getClass() == arg0.TCP_model.getClass()
this.MM.UDPNormalLength_outsrc.getClass() == arg0.TCP_model.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_in.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.UDP_model_out.getClass()
arg0.TCP_model.getClass() == arg0.TCP_model_outsrc.getClass()

arg0.TCP_model.getClass() == arg0.UDP_model_outsrc.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_out.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_in.getClass()
arg0.TCP_model.getClass() == arg0.UDPNormalLength_outsrc.getClass()
arg0.TCP_model.getClass() == orig(arg0.TCP_model.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model.getClass())
arg0.TCP_model.getClass() == orig(arg0.TCP_model_in.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model_in.getClass())
arg0.TCP_model.getClass() == orig(arg0.TCP_model_out.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model_out.getClass())
arg0.TCP_model.getClass() == orig(arg0.TCP_model_outsrc.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDP_model_outsrc.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength_out.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength_in.getClass())
arg0.TCP_model.getClass() == orig(arg0.UDPNormalLength_outsrc.getClass())
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_in.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_out.getClass()
arg0.TCP_lengthModel.getClass() == arg0.TCP_lengthModel_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == arg0.UDP_lengthModel_outsrc.getClass()
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel_in.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel_in.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel_out.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel_out.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.TCP_lengthModel_outsrc.getClass())
arg0.TCP_lengthModel.getClass() == orig(arg0.UDP_lengthModel_outsrc.getClass())
arg0.host_ip.toString == orig(arg0.host_ip.toString)
arg0.lenProb.getClass() == arg0.lenProb_out.getClass()
arg0.lenProb.getClass() == arg0.lenProb_in.getClass()
arg0.lenProb.getClass() == arg0.lenProb_outsrc.getClass()
arg0.lenProb.getClass() == orig(arg0.lenProb.getClass())
arg0.lenProb.getClass() == orig(arg0.lenProb_out.getClass())
arg0.lenProb.getClass() == orig(arg0.lenProb_in.getClass())
arg0.lenProb.getClass() == orig(arg0.lenProb_outsrc.getClass())
arg0.normal_length_perc in arg0.anomalyLengthThreshold[]
arg0.anomalyLengthThreshold_out[] elements == arg0.normal_length_perc
arg0.normal_length_perc in arg0.anomalyLengthThreshold_out[]
==========
TestPacketHandler.byteToFloat(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
TestPacketHandler.byteToFloat(byte[], int):::EXIT
arg0[] == orig(arg0[])
return != null
return[] elements >= 0.0
arg0[] elements > orig(arg1)
==========
TestPacketHandler.byteToShort(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
TestPacketHandler.byteToShort(byte[], int):::EXIT
arg0[] > return[] (lexically)
arg0[] >= return[] (lexically)

arg0[] == orig(arg0[])
return != null
return[] elements >= 0
arg0[] elements > orig(arg1)
return[] elements > orig(arg1)
==========
TestPacketHandler.inBuffer(java.lang.String, byte[]):::ENTER
this.port_trunc_length == this.MM.port_trunc_length
this.smooth_factor == this.fpRate
this.MM.host_ip == this.host_ip
this.adjustTCP_count[] < arg1[] (lexically)
this.adjustTCP_count[] <= arg1[] (lexically)
this.adjustTCP_anomaly_count[] < arg1[] (lexically)
this.adjustTCP_anomaly_count[] <= arg1[] (lexically)
this.adjustUDP_count[] < arg1[] (lexically)
this.adjustUDP_count[] <= arg1[] (lexically)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.TCP_portready[] == [1, 1, 1,

this.TCP_portready[] elements != null
this.fileModel.centroids[] contains no nulls and has only one value, of length 35
this.fileModel.centroids[].getClass() == [OneCentroid, OneCentroid]
arg0 != null
arg0.toString == "tcp_in80"
arg1 != null
this.count < this.packetIndex
arg1[] elements > this.count
arg1[] elements > this.trunc_length
this.TCP_portready[] elements == this.normal_alert
arg1[] elements > this.verbose
arg1[] elements < this.MM.max_sample
this.MM.host_ip.toString == this.host_ip.toString
this.string_compare_method.toString < arg0.toString
this.host_ip.toString < arg0.toString
this.adjustTCP_count[] elements < this.packetIndex
this.adjustTCP_anomaly_count[] elements < this.packetIndex
arg1[] elements >= this.fileModel.NMax
this.fileModel.NMax in arg1[]
this.packetIndex >= this.linkType
arg1[] elements > this.TCP
arg1[] elements > this.linkType
==========
TestPacketHandler.inBuffer(java.lang.String, byte[]):::EXIT996
==========
TestPacketHandler.inBuffer(java.lang.String, byte[]):::EXIT
this.count == orig(this.count)
this.count == orig(this.anomaly_count)
this.count == orig(this.old_anomaly)
this.count == orig(this.MM.min_sample)
this.count == orig(this.MM.modification_time)
this.count == orig(this.fileModel.start)
this.count == orig(this.fileModel.end)
this.count == orig(this.fileModel.min_sample)
this.count == orig(this.fileModel.max_sample)
this.count == orig(this.fileModel.totalPacketNum)
this.count == orig(this.fileModel.reduceN)
this.trunc_length == orig(this.trunc_length)
this.trunc_length == orig(this.MM.trunc_length)
this.port_trunc_length == this.MM.port_trunc_length
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.file_filtering == orig(this.file_filtering)
this.file_filtering == orig(this.MM.aggregation_done)
this.file_filtering == orig(this.stable)
this.file_filtering == orig(this.fileModel.batch)
this.smooth_factor == this.fpRate
this.smooth_factor == orig(this.smooth_factor)
this.maxnum == orig(this.maxnum)
this.threshold == orig(this.threshold)
this.normal_alert == orig(this.normal_alert)
this.verbose == orig(this.verbose)
this.temp == orig(this.temp)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)

this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)

this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] < arg1[] (lexically)
this.adjustTCP_count[] <= arg1[] (lexically)
this.adjustTCP_count[] == orig(this.adjustTCP_count[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustTCP_anomaly_count[] < arg1[] (lexically)
this.adjustTCP_anomaly_count[] <= arg1[] (lexically)
this.adjustTCP_anomaly_count[] == orig(this.adjustTCP_anomaly_count[])
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_count[] < arg1[] (lexically)
this.adjustUDP_count[] <= arg1[] (lexically)
this.adjustUDP_count[] == orig(this.adjustUDP_count[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.TCP_portready[] == orig(this.TCP_portready[])
this.TCP_portready[] == orig(this.UDP_portready[])
this.TCP_portready[] == orig(this.TCP_portready_out[])
this.TCP_portready[] == orig(this.UDP_portready_out[])
this.TCP_portready[] == orig(this.TCP_portready_in[])
this.TCP_portready[] == orig(this.UDP_portready_in[])
this.TCP_portready[] == orig(this.TCP_portready_outsrc[])
this.TCP_portready[] == orig(this.UDP_portready_outsrc[])
this.UDP_portready == orig(this.UDP_portready)
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)

this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.packetIndex == orig(this.packetIndex)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
arg1[] == orig(arg1[])
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.TCP_portready[] == [1, 1, 1,

this.fileModel.centroids[] contains no nulls and has only one value, of length 35
this.fileModel.centroids[].getClass() == [OneCentroid, OneCentroid]
arg0.toString == "tcp_in80"
return == -1.0
this.count < this.packetIndex
arg1[] elements > this.count
arg1[] elements > this.trunc_length
this.threshold > return
this.TCP_portready[] elements == this.normal_alert
arg1[] elements > this.verbose
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
arg1[] elements < this.MM.max_sample
this.MM.host_ip.toString == this.host_ip.toString
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements > return
this.string_compare_method.toString < arg0.toString
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.host_ip.toString < arg0.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.adjustTCP_count[] elements < this.packetIndex
this.adjustTCP_anomaly_count[] elements < this.packetIndex
arg1[] elements >= this.fileModel.NMax
this.fileModel.NMax in arg1[]
this.packetIndex >= this.linkType
arg1[] elements > this.TCP
arg1[] elements > this.linkType
arg0.toString == orig(arg0.toString)
==========
TestPacketHandler.isReady():::ENTER
this.port_trunc_length == this.MM.port_trunc_length
this.smooth_factor == this.fpRate
this.MM.host_ip == this.host_ip
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]

this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.port_trunc_length has only one value
this.port_keywords has only one value
this.port_keywords_range has only one value
this.threshold == 409.0
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_src_ports_set has only one value
this.out_dst_ports_set has only one value
this.out_src_ports_set has only one value
this.adjustTCP_count[] elements one of 0, 299
this.adjustTCP_count[] one of [0, 0, 0,

0, 0, 0,

Centroid, OneCentroid]

this.packetIndex == 299

this.TCP_portready[] elements == this.normal_alert

this.MM.host_ip.toString == this.host_ip.toString

==========

TestPacketHandler.isReady():::EXIT

this.count == orig(this.count)

this.count == orig(this.anomaly_count)

this.count == orig(this.old_anomaly)

this.count == orig(this.MM.min_sample)

this.count == orig(this.MM.modification_time)

this.count == orig(this.fileModel.start)

this.count == orig(this.fileModel.end)

this.count == orig(this.fileModel.min_sample)

this.count == orig(this.fileModel.max_sample)

this.count == orig(this.fileModel.totalPacketNum)

this.count == orig(this.fileModel.reduceN)

this.trunc_length == orig(this.trunc_length)

this.trunc_length == orig(this.MM.trunc_length)

this.port_trunc_length == this.MM.port_trunc_length

this.port_trunc_length == orig(this.port_trunc_length)

this.port_trunc_length == orig(this.MM.port_trunc_length)

this.port_keywords == orig(this.port_keywords)

this.port_keywords_range == orig(this.port_keywords_range)

this.file_filtering == orig(this.file_filtering)

this.file_filtering == orig(this.MM.aggregation_done)

this.file_filtering == orig(this.stable)

this.file_filtering == orig(this.fileModel.batch)

this.smooth_factor == this.fpRate

this.smooth_factor == orig(this.smooth_factor)

this.maxnum == orig(this.maxnum)

this.threshold == orig(this.threshold)

this.normal_alert == orig(this.normal_alert)

this.verbose == orig(this.verbose)

this.temp == orig(this.temp)

this.MM == orig(this.MM)

this.MM.TCP_model == orig(this.MM.TCP_model)

this.MM.TCP_model[] == orig(this.MM.TCP_model[])

this.MM.UDP_model == orig(this.MM.UDP_model)

this.MM.UDP_model[] == orig(this.MM.UDP_model[])

this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])

this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])

this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])

this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])

this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])

this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])

this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])

this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])

this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])

this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])

this.MM.TCP_model_in == orig(this.MM.TCP_model_in)

this.MM.UDP_model_in == orig(this.MM.UDP_model_in)

this.MM.TCP_model_out == orig(this.MM.TCP_model_out)

this.MM.UDP_model_out == orig(this.MM.UDP_model_out)

this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)

this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)

this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)

this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])

this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)

this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])

this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])

this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.adjustTCP_count[] == orig(this.adjustUDP_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_out[])

this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.UDP_portready == orig(this.UDP_portready)
this.UDP_portready[] == orig(this.TCP_portready[])
this.UDP_portready[] == orig(this.UDP_portready[])
this.UDP_portready[] == orig(this.TCP_portready_out[])
this.UDP_portready[] == orig(this.UDP_portready_out[])
this.UDP_portready[] == orig(this.TCP_portready_in[])
this.UDP_portready[] == orig(this.UDP_portready_in[])
this.UDP_portready[] == orig(this.TCP_portready_outsrc[])
this.UDP_portready[] == orig(this.UDP_portready_outsrc[])
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.packetIndex == orig(this.packetIndex)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
(this.TCP_portready[] == [1, 1, 1,

1, 1, 1,

0, 0, 0,

1, 1, 1,

1, 1, 1,

1, 1, 1,

1, 1, 1,

1, 1, 1,

1, 1, 1,

1, 1, 1,

1, 1, 1,

0, 0, 0,

return == orig(this.normal_alert)
this.TCP_portready[] == [1, 1, 1,

0, 0, 0,

this.adjustTCP_count[] one of [0, 0, 0,

0, 0, 0,

0, 0, 0,

this.MM.TCP_model.getClass() == arg0.getClass()
arg1[] elements < this.MM.max_sample
arg2[] elements < this.MM.max_sample
this.MM.host_ip.toString == this.host_ip.toString
this.string_compare_method.toString > arg4.toString
this.host_ip.toString < arg4.toString
this.adjustTCP_count[] >= arg1[] (elementwise)
this.adjustTCP_count[] >= arg2[] (elementwise)
this.adjustTCP_anomaly_count[] elements < this.packetIndex
this.adjustTCP_anomaly_count[] >= arg2[] (elementwise)
this.adjustUDP_count[] <= arg1[] (elementwise)
this.adjustUDP_count[] <= arg2[] (elementwise)
arg1[] elements <= this.packetIndex
arg2[] elements < this.packetIndex
arg1[] >= arg2[] (elementwise)
==========
TestPacketHandler.isReady_onedir(java.util.HashMap[], int[], int[], boolean[], java.lang.String):::EXIT
this.count == orig(this.count)
this.count == orig(this.anomaly_count)
this.count == orig(this.old_anomaly)
this.count == orig(this.MM.min_sample)
this.count == orig(this.MM.modification_time)
this.count == orig(this.fileModel.start)
this.count == orig(this.fileModel.end)
this.count == orig(this.fileModel.min_sample)
this.count == orig(this.fileModel.max_sample)
this.count == orig(this.fileModel.totalPacketNum)
this.count == orig(this.fileModel.reduceN)
this.trunc_length == orig(this.trunc_length)
this.trunc_length == orig(this.MM.trunc_length)
this.port_trunc_length == this.MM.port_trunc_length
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.file_filtering == orig(this.file_filtering)
this.file_filtering == orig(this.MM.aggregation_done)
this.file_filtering == orig(this.stable)
this.file_filtering == orig(this.fileModel.batch)
this.smooth_factor == this.fpRate
this.smooth_factor == orig(this.smooth_factor)
this.maxnum == orig(this.maxnum)
this.threshold == orig(this.threshold)
this.normal_alert == orig(this.normal_alert)
this.verbose == orig(this.verbose)
this.temp == orig(this.temp)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)

this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] == orig(this.adjustTCP_count[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustTCP_anomaly_count[] == orig(this.adjustTCP_anomaly_count[])
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_count[] == orig(this.adjustUDP_count[])

this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.UDP_portready == orig(this.UDP_portready)
this.UDP_portready[] == orig(this.UDP_portready[])
this.UDP_portready[] == orig(this.TCP_portready_out[])
this.UDP_portready[] == orig(this.UDP_portready_out[])
this.UDP_portready[] == orig(this.TCP_portready_in[])
this.UDP_portready[] == orig(this.UDP_portready_in[])
this.UDP_portready[] == orig(this.TCP_portready_outsrc[])
this.UDP_portready[] == orig(this.UDP_portready_outsrc[])
this.UDP_portready[] == orig(arg3[])
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.packetIndex == orig(this.packetIndex)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
(arg1[] == [0, 0, 0,

0, 0, 0,

1, 1, 1,

0, 0, 0,

0, 0, 0,

0, 0, 0,

1, 1, 1,

0, 0, 0,

1, 1, 1,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

orig(arg2))
(arg1[] == [0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

0, 0, 0,

this.MM.TCP_model.getClass() == orig(arg0.getClass())
this.MM.UDP_model.getClass() == orig(arg0.getClass())
this.MM.TCP_model_in.getClass() == orig(arg0.getClass())
this.MM.UDP_model_in.getClass() == orig(arg0.getClass())
this.MM.TCP_model_out.getClass() == orig(arg0.getClass())
this.MM.UDP_model_out.getClass() == orig(arg0.getClass())
this.MM.TCP_model_outsrc.getClass() == orig(arg0.getClass())
this.MM.UDP_model_outsrc.getClass() == orig(arg0.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() != orig(arg0.getClass())
arg1[] elements < this.MM.max_sample
arg2[] elements < this.MM.max_sample
this.MM.host_ip.toString == this.host_ip.toString
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.lenProb.getClass() != orig(arg0.getClass())
this.MM.UDPNormalLength.getClass() == orig(arg0.getClass())
this.MM.UDPNormalLength_out.getClass() == orig(arg0.getClass())
this.MM.UDPNormalLength_in.getClass() == orig(arg0.getClass())
this.MM.UDPNormalLength_outsrc.getClass() == orig(arg0.getClass())
this.string_compare_method.toString > arg4.toString
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.host_ip.toString < arg4.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.adjustTCP_count[] >= arg1[] (elementwise)
this.adjustTCP_count[] >= arg2[] (elementwise)
this.adjustTCP_anomaly_count[] elements < this.packetIndex
this.adjustTCP_anomaly_count[] >= arg2[] (elementwise)
this.adjustUDP_count[] <= arg1[] (elementwise)
this.adjustUDP_count[] <= arg2[] (elementwise)
return in this.TCP_portready[]
arg1[] elements <= this.packetIndex
arg2[] elements < this.packetIndex
arg1[] >= arg2[] (elementwise)
return in arg3[]
arg4.toString == orig(arg4.toString)
orig(this.MM.UDP_model.getClass()) == orig(arg0.getClass())
orig(this.MM.TCP_model_in.getClass()) == orig(arg0.getClass())
orig(this.MM.UDP_model_in.getClass()) == orig(arg0.getClass())
orig(this.MM.TCP_model_out.getClass()) == orig(arg0.getClass())
orig(this.MM.UDP_model_out.getClass()) == orig(arg0.getClass())
orig(this.MM.TCP_model_outsrc.getClass()) == orig(arg0.getClass())
orig(this.MM.UDP_model_outsrc.getClass()) == orig(arg0.getClass())
orig(this.MM.UDPNormalLength.getClass()) == orig(arg0.getClass())
orig(this.MM.UDPNormalLength_out.getClass()) == orig(arg0.getClass())
orig(this.MM.UDPNormalLength_in.getClass()) == orig(arg0.getClass())
orig(this.MM.UDPNormalLength_outsrc.getClass()) == orig(arg0.getClass())
==========
TestPacketHandler.isReady_onedir(java.util.HashMap[], int[], int[], boolean[], java.lang.String):::EXIT;condition="return
== true"
this.normal_alert == return
this.adjustUDP_count[] == arg1[]
this.adjustUDP_anomaly_count[] == arg1[]

this.TCP_portready[] == orig(this.TCP_portready[])
this.UDP_portready[] == arg3[]
this.adjustTCP_count_out[] == arg1[]
this.adjustTCP_anomaly_count_out[] == arg1[]
this.adjustUDP_count_out[] == arg1[]
this.adjustUDP_anomaly_count_out[] == arg1[]
this.TCP_portready_out[] == arg3[]
this.UDP_portready_out[] == arg3[]
this.adjustTCP_count_in[] == arg1[]
this.adjustTCP_anomaly_count_in[] == arg1[]
this.adjustUDP_count_in[] == arg1[]
this.adjustUDP_anomaly_count_in[] == arg1[]
this.TCP_portready_in[] == arg3[]
this.UDP_portready_in[] == arg3[]
this.adjustTCP_count_outsrc[] == arg1[]
this.adjustTCP_anomaly_count_outsrc[] == arg1[]
this.adjustUDP_count_outsrc[] == arg1[]
this.adjustUDP_anomaly_count_outsrc[] == arg1[]
this.TCP_portready_outsrc[] == arg3[]
this.UDP_portready_outsrc[] == arg3[]
arg1[] == arg2[]
arg1[] == orig(this.adjustUDP_count[])
arg1[] == orig(this.adjustUDP_anomaly_count[])
arg1[] == orig(this.adjustTCP_count_out[])
arg1[] == orig(this.adjustTCP_anomaly_count_out[])
arg1[] == orig(this.adjustUDP_count_out[])
arg1[] == orig(this.adjustUDP_anomaly_count_out[])
arg1[] == orig(this.adjustTCP_count_in[])
arg1[] == orig(this.adjustTCP_anomaly_count_in[])
arg1[] == orig(this.adjustUDP_count_in[])
arg1[] == orig(this.adjustUDP_anomaly_count_in[])
arg1[] == orig(this.adjustTCP_count_outsrc[])
arg1[] == orig(this.adjustTCP_anomaly_count_outsrc[])
arg1[] == orig(this.adjustUDP_count_outsrc[])
arg1[] == orig(this.adjustUDP_anomaly_count_outsrc[])
arg1[] == orig(arg2[])
arg3[] == orig(this.UDP_portready[])
arg3[] == orig(this.TCP_portready_out[])
arg3[] == orig(this.UDP_portready_out[])
arg3[] == orig(this.TCP_portready_in[])
arg3[] == orig(this.UDP_portready_in[])
arg3[] == orig(this.TCP_portready_outsrc[])
arg3[] == orig(this.UDP_portready_outsrc[])
arg3[] == orig(arg3[])
return == orig(this.normal_alert)
arg1[] == [0, 0, 0,

0, 0, 0,

0, 0, 0,

arg0 != null
arg1 != null
==========
TestPacketHandler.isSame(byte[], byte[]):::EXIT984
==========
TestPacketHandler.isSame(byte[], byte[]):::EXIT984;condition="not(return == true)"
==========
TestPacketHandler.isSame(byte[], byte[]):::EXIT988
==========
TestPacketHandler.isSame(byte[], byte[]):::EXIT988;condition="not(return == true)"
==========
TestPacketHandler.isSame(byte[], byte[]):::EXIT
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
return == false
==========
TestPacketHandler.isSame(byte[], byte[]):::EXIT;condition="not(return == true)"
==========
TestPacketHandler.rawPacketArrived(net.sourceforge.jpcap.net.RawPacket):::ENTER
this.count == arg0.droplen
this.port_trunc_length == this.MM.port_trunc_length
this.smooth_factor == this.fpRate
this.MM.host_ip == this.host_ip
this.adjustTCP_count[] < arg0.bytes[] (lexically)
this.adjustTCP_count[] <= arg0.bytes[] (lexically)
this.adjustTCP_anomaly_count[] < arg0.bytes[] (lexically)
this.adjustTCP_anomaly_count[] <= arg0.bytes[] (lexically)
this.adjustUDP_count[] < arg0.bytes[] (lexically)
this.adjustUDP_count[] <= arg0.bytes[] (lexically)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.TCP_portready[] == [1, 1, 1,

1, 1, 1,

arg0.timeval.seconds > arg0.timeval.microseconds
arg0.bytes[] elements < arg0.timeval.seconds
arg0.bytes[] elements < arg0.timeval.microseconds
arg0.timeval._rcsid.toString > arg0._rcsid.toString
7190749 * this.packetIndex - 719119 * arg0.timeval.seconds + arg0.timeval.microseconds + 7.94257731448847E14 == 0
==========
TestPacketHandler.rawPacketArrived(net.sourceforge.jpcap.net.RawPacket):::EXIT679
==========
TestPacketHandler.rawPacketArrived(net.sourceforge.jpcap.net.RawPacket):::EXIT
this.count == arg0.droplen
this.count == orig(this.count)
this.count == orig(this.anomaly_count)
this.count == orig(this.old_anomaly)
this.count == orig(this.MM.min_sample)
this.count == orig(this.MM.modification_time)
this.count == orig(this.fileModel.start)
this.count == orig(this.fileModel.end)
this.count == orig(this.fileModel.min_sample)
this.count == orig(this.fileModel.max_sample)
this.count == orig(this.fileModel.totalPacketNum)
this.count == orig(this.fileModel.reduceN)
this.count == orig(arg0.droplen)
this.trunc_length == orig(this.trunc_length)
this.trunc_length == orig(this.MM.trunc_length)
this.port_trunc_length == this.MM.port_trunc_length
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.file_filtering == orig(this.file_filtering)
this.file_filtering == orig(this.MM.aggregation_done)
this.file_filtering == orig(this.stable)
this.file_filtering == orig(this.fileModel.batch)
this.smooth_factor == this.fpRate
this.smooth_factor == orig(this.smooth_factor)
this.maxnum == orig(this.maxnum)
this.normal_alert == orig(this.normal_alert)
this.verbose == orig(this.verbose)
this.temp == orig(this.temp)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)

this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] > this.adjustUDP_count[] (lexically)
this.adjustTCP_count[] < arg0.bytes[] (lexically)
this.adjustTCP_count[] <= arg0.bytes[] (lexically)
this.adjustTCP_count[] > orig(this.adjustTCP_count[]) (lexically)
this.adjustTCP_count[] > orig(this.adjustTCP_anomaly_count[]) (lexically)
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustTCP_anomaly_count[] > this.adjustUDP_count[] (lexically)
this.adjustTCP_anomaly_count[] < arg0.bytes[] (lexically)
this.adjustTCP_anomaly_count[] <= arg0.bytes[] (lexically)
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_count[] < arg0.bytes[] (lexically)
this.adjustUDP_count[] <= arg0.bytes[] (lexically)

this.adjustUDP_count[] == orig(this.adjustUDP_count[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.TCP_portready[] == orig(this.TCP_portready[])
this.TCP_portready[] == orig(this.UDP_portready[])
this.TCP_portready[] == orig(this.TCP_portready_out[])
this.TCP_portready[] == orig(this.UDP_portready_out[])
this.TCP_portready[] == orig(this.TCP_portready_in[])
this.TCP_portready[] == orig(this.UDP_portready_in[])
this.TCP_portready[] == orig(this.TCP_portready_outsrc[])
this.TCP_portready[] == orig(this.UDP_portready_outsrc[])
this.UDP_portready == orig(this.UDP_portready)
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)

this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
arg0.timeval == orig(arg0.timeval)
arg0.timeval.seconds == orig(arg0.timeval.seconds)
arg0.timeval.microseconds == orig(arg0.timeval.microseconds)
arg0.timeval._rcsid == orig(arg0.timeval._rcsid)
arg0.bytes == orig(arg0.bytes)
arg0.bytes[] > orig(this.adjustTCP_count[]) (lexically)
arg0.bytes[] >= orig(this.adjustTCP_count[]) (lexically)
arg0.bytes[] > orig(this.adjustTCP_anomaly_count[]) (lexically)
arg0.bytes[] >= orig(this.adjustTCP_anomaly_count[]) (lexically)
arg0.bytes[] == orig(arg0.bytes[])
arg0._rcsid == orig(arg0._rcsid)
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.TCP_portready[] == [1, 1, 1,

arg0.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
arg0.bytes != null
arg0._rcsid has only one value
arg0._rcsid != null
arg0._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this.count < this.packetIndex
this.count < arg0.timeval.seconds
this.count < arg0.timeval.microseconds
this.count in arg0.bytes[]
orig(this.adjustTCP_count[]) elements >= this.count
this.count in orig(this.adjustTCP_count[])
orig(this.adjustTCP_anomaly_count[]) elements >= this.count
this.count in orig(this.adjustTCP_anomaly_count[])
this.count <= orig(this.packetIndex)
this.trunc_length < arg0.timeval.seconds
this.trunc_length < arg0.timeval.microseconds
orig(this.adjustTCP_count[]) elements > this.trunc_length
orig(this.adjustTCP_anomaly_count[]) elements > this.trunc_length
this.trunc_length < orig(this.packetIndex)
this.smooth_factor < orig(this.threshold)
this.maxnum > orig(this.threshold)
this.TCP_portready[] elements == this.normal_alert
this.verbose < arg0.timeval.seconds
this.verbose < arg0.timeval.microseconds
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.max_sample < arg0.timeval.seconds
this.MM.max_sample < arg0.timeval.microseconds
arg0.bytes[] elements < this.MM.max_sample
orig(this.adjustTCP_count[]) elements < this.MM.max_sample
orig(this.adjustTCP_anomaly_count[]) elements < this.MM.max_sample
this.MM.max_sample > orig(this.packetIndex)
this.MM.host_ip.toString == this.host_ip.toString
this.MM.stabledis < orig(this.threshold)
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements < orig(this.threshold)
this.MM.normal_length_perc < orig(this.threshold)
this.kernel_h < orig(this.threshold)
this.string_compare_method.toString > arg0.timeval._rcsid.toString
this.string_compare_method.toString > arg0._rcsid.toString
this.string_compare_method.toString == orig(this.string_compare_method.toString)
orig(this.threshold)

this.string_compare_threshold < orig(this.threshold)
this.fpRate < orig(this.threshold)
this.host_ip.toString > arg0.timeval._rcsid.toString
this.host_ip.toString > arg0._rcsid.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.packetIndex in this.adjustTCP_count[]
this.adjustTCP_count[] elements < arg0.timeval.seconds
this.adjustTCP_count[] elements < arg0.timeval.microseconds
this.adjustTCP_count[] >= orig(this.adjustTCP_count[]) (elementwise)
this.adjustTCP_count[] >= orig(this.adjustTCP_anomaly_count[]) (elementwise)
this.adjustTCP_anomaly_count[] elements < arg0.timeval.seconds
this.adjustTCP_anomaly_count[] elements < arg0.timeval.microseconds
this.adjustTCP_anomaly_count[] >= orig(this.adjustTCP_anomaly_count[]) (elementwise)
this.adjustUDP_count[] <= orig(this.adjustTCP_count[]) (elementwise)
this.adjustUDP_count[] <= orig(this.adjustTCP_anomaly_count[]) (elementwise)
this.fileModel.NMax < arg0.timeval.seconds
this.fileModel.NMax < arg0.timeval.microseconds
this.fileModel.NMax in arg0.bytes[]
this.packetIndex >= this.linkType
this.packetIndex < arg0.timeval.seconds
this.packetIndex < arg0.timeval.microseconds
orig(this.adjustTCP_count[]) elements < this.packetIndex
orig(this.adjustTCP_anomaly_count[]) elements < this.packetIndex
this.packetIndex - orig(this.packetIndex) - 1 == 0
this.TCP < arg0.timeval.seconds
this.TCP < arg0.timeval.microseconds
this.TCP in arg0.bytes[]
this.UDP < arg0.timeval.seconds
this.UDP < arg0.timeval.microseconds
this.linkType < arg0.timeval.seconds
this.linkType < arg0.timeval.microseconds
this.linkType in arg0.bytes[]
arg0.timeval.seconds > arg0.timeval.microseconds
arg0.bytes[] elements < arg0.timeval.seconds
orig(this.adjustTCP_count[]) elements < arg0.timeval.seconds
orig(this.adjustTCP_anomaly_count[]) elements < arg0.timeval.seconds
arg0.timeval.seconds > orig(this.packetIndex)
arg0.bytes[] elements < arg0.timeval.microseconds
orig(this.adjustTCP_count[]) elements < arg0.timeval.microseconds
orig(this.adjustTCP_anomaly_count[]) elements < arg0.timeval.microseconds
arg0.timeval.microseconds > orig(this.packetIndex)
arg0.timeval._rcsid.toString > arg0._rcsid.toString
arg0.timeval._rcsid.toString == orig(arg0.timeval._rcsid.toString)
arg0._rcsid.toString == orig(arg0._rcsid.toString)
7190749 * this.packetIndex - 719119 * arg0.timeval.seconds + arg0.timeval.microseconds + 7.94257724258098E14 == 0
719119 * arg0.timeval.seconds - arg0.timeval.microseconds - 7190749 * orig(this.packetIndex) - 7.94257731448847E14 == 0
==========
TestPacketHandler.readFileModels():::ENTER
this.count == this.packetIndex
this.port_trunc_length == this.port_keywords
this.port_trunc_length == this.port_keywords_range
this.in_dst_ports_set == this.in_src_ports_set
this.in_dst_ports_set == this.out_dst_ports_set
this.in_dst_ports_set == this.out_src_ports_set
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]

this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.fileModel.centroids[] == this.fileModel.b_packets[]
this.fileModel.centroids[] == this.fileModel.b_nums[]
this.port_trunc_length == null
this.threshold == 64.0
this.string_compare_method has only one value
this.fpRate == 0.01
this.host_ip has only one value
this.host_ip.toString == ""
this.in_dst_ports_set == null
this.adjustTCP_count[] == [0, 0, 0,

1, 1, 1,

this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.port_trunc_length == orig(this.MM.port_trunc_length)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == orig(this.MM.host_ip)
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == this.in_src_ports_set
this.in_dst_ports_set == this.out_dst_ports_set
this.in_dst_ports_set == this.out_src_ports_set
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_dst_ports_set == orig(this.in_src_ports_set)
this.in_dst_ports_set == orig(this.out_dst_ports_set)
this.in_dst_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]

this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.adjustTCP_count[] == orig(this.adjustTCP_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.TCP_portready[] == orig(this.TCP_portready[])
this.TCP_portready[] == orig(this.UDP_portready[])
this.TCP_portready[] == orig(this.TCP_portready_out[])
this.TCP_portready[] == orig(this.UDP_portready_out[])
this.TCP_portready[] == orig(this.TCP_portready_in[])
this.TCP_portready[] == orig(this.UDP_portready_in[])
this.TCP_portready[] == orig(this.TCP_portready_outsrc[])
this.TCP_portready[] == orig(this.UDP_portready_outsrc[])
this.UDP_portready == orig(this.UDP_portready)
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)

this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.centroids[])
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
this.port_trunc_length == null
this.threshold == 64.0
this.string_compare_method has only one value
this.fpRate == 0.01
this.host_ip has only one value
this.host_ip.toString == ""
this.in_dst_ports_set == null
this.adjustTCP_count[] == [0, 0, 0,

this.fileModel.centroids[] contains no nulls and has only one value, of length 35
this.fileModel.centroids[].getClass() == [OneCentroid, OneCentroid]
this.adjustTCP_count[] elements == this.count
this.port_trunc_length in this.MM.TCP_model[]
this.MM.UDP_model[] elements == this.port_trunc_length
this.port_trunc_length in this.MM.UDP_model[]
this.TCP_portready[] elements == this.normal_alert
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.host_ip.toString == orig(this.MM.host_ip.toString)
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.host_ip.toString == orig(this.host_ip.toString)
==========
TestPacketHandler.reset(double):::ENTER
this.port_trunc_length == this.MM.port_trunc_length
this.smooth_factor == this.fpRate
this.MM.host_ip == this.host_ip
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.port_trunc_length has only one value
this.port_keywords has only one value
this.port_keywords_range has only one value
this.threshold == 409.0
this.string_compare_method has only one value

this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_src_ports_set has only one value
this.out_dst_ports_set has only one value
this.out_src_ports_set has only one value
this.adjustTCP_count[] == [0, 0, 0,

this.count == orig(this.count)
this.count == orig(this.anomaly_count)
this.count == orig(this.old_anomaly)
this.count == orig(this.MM.min_sample)
this.count == orig(this.MM.modification_time)
this.count == orig(this.fileModel.start)
this.count == orig(this.fileModel.end)
this.count == orig(this.fileModel.min_sample)
this.count == orig(this.fileModel.max_sample)
this.count == orig(this.fileModel.totalPacketNum)
this.count == orig(this.fileModel.reduceN)
this.trunc_length == orig(this.trunc_length)
this.trunc_length == orig(this.MM.trunc_length)
this.port_trunc_length == this.MM.port_trunc_length
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.file_filtering == orig(this.file_filtering)
this.file_filtering == orig(this.MM.aggregation_done)
this.file_filtering == orig(this.stable)
this.file_filtering == orig(this.fileModel.batch)
this.smooth_factor == this.fpRate
this.smooth_factor == orig(this.smooth_factor)
this.maxnum == orig(this.maxnum)
this.threshold == orig(this.threshold)
this.normal_alert == orig(this.normal_alert)
this.verbose == orig(this.verbose)
this.temp == orig(this.temp)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)

this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.adjustTCP_count[] == orig(this.adjustTCP_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_in[])

this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.TCP_portready[] == orig(this.UDP_portready[])
this.TCP_portready[] == orig(this.TCP_portready_out[])
this.TCP_portready[] == orig(this.UDP_portready_out[])
this.TCP_portready[] == orig(this.TCP_portready_in[])
this.TCP_portready[] == orig(this.UDP_portready_in[])
this.TCP_portready[] == orig(this.TCP_portready_outsrc[])
this.TCP_portready[] == orig(this.UDP_portready_outsrc[])
this.UDP_portready == orig(this.UDP_portready)
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.packetIndex == orig(this.packetIndex)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
this.port_trunc_length has only one value
this.port_keywords has only one value
this.port_keywords_range has only one value
this.threshold == 409.0
this.string_compare_method has only one value

this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_src_ports_set has only one value
this.out_dst_ports_set has only one value
this.out_src_ports_set has only one value
this.adjustTCP_count[] == [0, 0, 0,

this.MM.TCP_model.getClass() == orig(this.MM.UDP_model.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.TCP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDP_model_outsrc.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_out.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
this.MM.host_ip.toString == this.host_ip.toString
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
==========
TestPacketHandler.reset_onedir(boolean[], java.util.Vector[], java.util.HashMap[], double, java.lang.String):::ENTER
this.port_trunc_length == this.MM.port_trunc_length
this.smooth_factor == this.fpRate
this.MM.host_ip == this.host_ip
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.TCP_portready[] == arg0[]
this.port_trunc_length has only one value
this.port_keywords has only one value
this.port_keywords_range has only one value
this.threshold == 409.0
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_src_ports_set has only one value
this.out_dst_ports_set has only one value
this.out_src_ports_set has only one value
this.adjustTCP_count[] == [0, 0, 0,

0, 0, 0,

this.verbose == orig(this.verbose)
this.temp == orig(this.temp)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)
this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)

this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] == this.adjustTCP_anomaly_count[]
this.adjustTCP_count[] == this.adjustUDP_count[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count[]
this.adjustTCP_count[] == this.adjustTCP_count_out[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustUDP_count_out[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_out[]
this.adjustTCP_count[] == this.adjustTCP_count_in[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustUDP_count_in[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_in[]
this.adjustTCP_count[] == this.adjustTCP_count_outsrc[]
this.adjustTCP_count[] == this.adjustTCP_anomaly_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_count_outsrc[]
this.adjustTCP_count[] == this.adjustUDP_anomaly_count_outsrc[]
this.adjustTCP_count[] == orig(this.adjustTCP_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_count[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_out[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_in[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustTCP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustTCP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.TCP_portready[] == arg0[]
this.UDP_portready == orig(this.UDP_portready)
this.UDP_portready[] == arg0[]
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.TCP_portready_out[] == arg0[]
this.UDP_portready_out == orig(this.UDP_portready_out)
this.UDP_portready_out[] == arg0[]
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)

this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.TCP_portready_in[] == arg0[]
this.UDP_portready_in == orig(this.UDP_portready_in)
this.UDP_portready_in[] == arg0[]
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.TCP_portready_outsrc[] == arg0[]
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.UDP_portready_outsrc[] == arg0[]
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.packetIndex == orig(this.packetIndex)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
arg0[] == orig(this.UDP_portready[])
arg0[] == orig(this.TCP_portready_out[])
arg0[] == orig(this.UDP_portready_out[])
arg0[] == orig(this.TCP_portready_in[])
arg0[] == orig(this.UDP_portready_in[])
arg0[] == orig(this.TCP_portready_outsrc[])
arg0[] == orig(this.UDP_portready_outsrc[])
arg1[] == orig(arg1[])
arg2[] == orig(arg2[])
this.port_trunc_length has only one value
this.port_keywords has only one value
this.port_keywords_range has only one value
this.threshold == 409.0
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_src_ports_set has only one value
this.out_dst_ports_set has only one value
this.out_src_ports_set has only one value
this.adjustTCP_count[] == [0, 0, 0,

0, 0, 0,

1, 1, 1,

orig(this.MM.UDP_model_outsrc.getClass()) == orig(arg2.getClass())
orig(this.MM.UDP_lengthModel.getClass()) == orig(arg1.getClass())
orig(this.MM.TCP_lengthModel_in.getClass()) == orig(arg1.getClass())
orig(this.MM.UDP_lengthModel_in.getClass()) == orig(arg1.getClass())
orig(this.MM.TCP_lengthModel_out.getClass()) == orig(arg1.getClass())
orig(this.MM.UDP_lengthModel_out.getClass()) == orig(arg1.getClass())
orig(this.MM.TCP_lengthModel_outsrc.getClass()) == orig(arg1.getClass())
orig(this.MM.UDP_lengthModel_outsrc.getClass()) == orig(arg1.getClass())
orig(this.MM.UDPNormalLength.getClass()) == orig(arg2.getClass())
orig(this.MM.UDPNormalLength_out.getClass()) == orig(arg2.getClass())
orig(this.MM.UDPNormalLength_in.getClass()) == orig(arg2.getClass())
orig(this.MM.UDPNormalLength_outsrc.getClass()) == orig(arg2.getClass())
orig(arg1.getClass()) != orig(arg2.getClass())
==========
TestPacketHandler.testTCPModel(java.util.HashMap, java.lang.String, byte[]):::ENTER
this.port_trunc_length == this.MM.port_trunc_length
this.smooth_factor == this.fpRate
this.MM.host_ip == this.host_ip
this.adjustTCP_count[] < arg2[] (lexically)
this.adjustTCP_count[] <= arg2[] (lexically)
this.adjustTCP_anomaly_count[] < arg2[] (lexically)
this.adjustTCP_anomaly_count[] <= arg2[] (lexically)
this.adjustUDP_count[] < arg2[] (lexically)
this.adjustUDP_count[] <= arg2[] (lexically)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]
this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value
this.port_keywords_range != null
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.TCP_portready[] == [1, 1, 1,

1, 1, 1,

this.count == orig(this.fileModel.min_sample)
this.count == orig(this.fileModel.max_sample)
this.count == orig(this.fileModel.totalPacketNum)
this.count == orig(this.fileModel.reduceN)
this.trunc_length == orig(this.trunc_length)
this.trunc_length == orig(this.MM.trunc_length)
this.port_trunc_length == this.MM.port_trunc_length
this.port_trunc_length == orig(this.port_trunc_length)
this.port_trunc_length == orig(this.MM.port_trunc_length)
this.port_keywords == orig(this.port_keywords)
this.port_keywords_range == orig(this.port_keywords_range)
this.file_filtering == orig(this.file_filtering)
this.file_filtering == orig(this.MM.aggregation_done)
this.file_filtering == orig(this.stable)
this.file_filtering == orig(this.fileModel.batch)
this.smooth_factor == this.fpRate
this.smooth_factor == orig(this.smooth_factor)
this.maxnum == orig(this.maxnum)
this.normal_alert == orig(this.normal_alert)
this.verbose == orig(this.verbose)
this.temp == orig(this.temp)
this.MM == orig(this.MM)
this.MM.TCP_model == orig(this.MM.TCP_model)
this.MM.TCP_model[] == orig(this.MM.TCP_model[])
this.MM.UDP_model == orig(this.MM.UDP_model)
this.MM.UDP_model[] == orig(this.MM.UDP_model[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_in[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_in[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_out[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_out[])
this.MM.UDP_model[] == orig(this.MM.TCP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDP_model_outsrc[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_out[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_in[])
this.MM.UDP_model[] == orig(this.MM.UDPNormalLength_outsrc[])
this.MM.TCP_model_in == orig(this.MM.TCP_model_in)
this.MM.UDP_model_in == orig(this.MM.UDP_model_in)
this.MM.TCP_model_out == orig(this.MM.TCP_model_out)
this.MM.UDP_model_out == orig(this.MM.UDP_model_out)
this.MM.TCP_model_outsrc == orig(this.MM.TCP_model_outsrc)
this.MM.UDP_model_outsrc == orig(this.MM.UDP_model_outsrc)
this.MM.TCP_lengthModel == orig(this.MM.TCP_lengthModel)
this.MM.TCP_lengthModel[] == orig(this.MM.TCP_lengthModel[])
this.MM.UDP_lengthModel == orig(this.MM.UDP_lengthModel)
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_in[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_out[])
this.MM.UDP_lengthModel[] == orig(this.MM.TCP_lengthModel_outsrc[])
this.MM.UDP_lengthModel[] == orig(this.MM.UDP_lengthModel_outsrc[])
this.MM.TCP_lengthModel_in == orig(this.MM.TCP_lengthModel_in)
this.MM.UDP_lengthModel_in == orig(this.MM.UDP_lengthModel_in)
this.MM.TCP_lengthModel_out == orig(this.MM.TCP_lengthModel_out)
this.MM.UDP_lengthModel_out == orig(this.MM.UDP_lengthModel_out)
this.MM.TCP_lengthModel_outsrc == orig(this.MM.TCP_lengthModel_outsrc)
this.MM.UDP_lengthModel_outsrc == orig(this.MM.UDP_lengthModel_outsrc)
this.MM.max_sample == orig(this.MM.max_sample)
this.MM.sample_rate == orig(this.MM.sample_rate)
this.MM.sample_rate == orig(this.fileModel.sample_rate)
this.MM.host_ip == this.host_ip
this.MM.stabledis == orig(this.MM.stabledis)

this.MM.smooth_factor == orig(this.MM.smooth_factor)
this.MM.lenProb == orig(this.MM.lenProb)
this.MM.lenProb[] == orig(this.MM.lenProb[])
this.MM.anomalyLengthThreshold == orig(this.MM.anomalyLengthThreshold)
this.MM.anomalyLengthThreshold[] == orig(this.MM.anomalyLengthThreshold[])
this.MM.lenProb_out == orig(this.MM.lenProb_out)
this.MM.lenProb_out[] == orig(this.MM.lenProb_out[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_in[])
this.MM.lenProb_out[] == orig(this.MM.lenProb_outsrc[])
this.MM.anomalyLengthThreshold_out == orig(this.MM.anomalyLengthThreshold_out)
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_out[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_in[])
this.MM.anomalyLengthThreshold_out[] == orig(this.MM.anomalyLengthThreshold_outsrc[])
this.MM.lenProb_in == orig(this.MM.lenProb_in)
this.MM.anomalyLengthThreshold_in == orig(this.MM.anomalyLengthThreshold_in)
this.MM.lenProb_outsrc == orig(this.MM.lenProb_outsrc)
this.MM.anomalyLengthThreshold_outsrc == orig(this.MM.anomalyLengthThreshold_outsrc)
this.MM.UDPNormalLength == orig(this.MM.UDPNormalLength)
this.MM.UDPNormalLength_out == orig(this.MM.UDPNormalLength_out)
this.MM.UDPNormalLength_in == orig(this.MM.UDPNormalLength_in)
this.MM.UDPNormalLength_outsrc == orig(this.MM.UDPNormalLength_outsrc)
this.MM.normal_length_perc == orig(this.MM.normal_length_perc)
this.kernel_h == orig(this.kernel_h)
this.string_compare_method == orig(this.string_compare_method)
this.string_compare_threshold == orig(this.string_compare_threshold)
this.fpRate == orig(this.fpRate)
this.host_ip == orig(this.MM.host_ip)
this.host_ip == orig(this.host_ip)
this.in_dst_ports_set == orig(this.in_dst_ports_set)
this.in_src_ports_set == orig(this.in_src_ports_set)
this.out_dst_ports_set == orig(this.out_dst_ports_set)
this.out_src_ports_set == orig(this.out_src_ports_set)
this.adjustTCP_count == orig(this.adjustTCP_count)
this.adjustTCP_count[] < arg2[] (lexically)
this.adjustTCP_count[] <= arg2[] (lexically)
this.adjustTCP_count[] == orig(this.adjustTCP_count[])
this.adjustTCP_anomaly_count == orig(this.adjustTCP_anomaly_count)
this.adjustTCP_anomaly_count[] < arg2[] (lexically)
this.adjustTCP_anomaly_count[] <= arg2[] (lexically)
this.adjustTCP_anomaly_count[] == orig(this.adjustTCP_anomaly_count[])
this.adjustUDP_count == orig(this.adjustUDP_count)
this.adjustUDP_count[] < arg2[] (lexically)
this.adjustUDP_count[] <= arg2[] (lexically)
this.adjustUDP_count[] == orig(this.adjustUDP_count[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_out[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_out[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_in[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_in[])
this.adjustUDP_count[] == orig(this.adjustTCP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustTCP_anomaly_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_count_outsrc[])
this.adjustUDP_count[] == orig(this.adjustUDP_anomaly_count_outsrc[])
this.adjustUDP_anomaly_count == orig(this.adjustUDP_anomaly_count)
this.TCP_portready == orig(this.TCP_portready)
this.TCP_portready[] == this.UDP_portready[]
this.TCP_portready[] == this.TCP_portready_out[]
this.TCP_portready[] == this.UDP_portready_out[]
this.TCP_portready[] == this.TCP_portready_in[]

this.TCP_portready[] == this.UDP_portready_in[]
this.TCP_portready[] == this.TCP_portready_outsrc[]
this.TCP_portready[] == this.UDP_portready_outsrc[]
this.TCP_portready[] == orig(this.TCP_portready[])
this.TCP_portready[] == orig(this.UDP_portready[])
this.TCP_portready[] == orig(this.TCP_portready_out[])
this.TCP_portready[] == orig(this.UDP_portready_out[])
this.TCP_portready[] == orig(this.TCP_portready_in[])
this.TCP_portready[] == orig(this.UDP_portready_in[])
this.TCP_portready[] == orig(this.TCP_portready_outsrc[])
this.TCP_portready[] == orig(this.UDP_portready_outsrc[])
this.UDP_portready == orig(this.UDP_portready)
this.adjustTCP_count_out == orig(this.adjustTCP_count_out)
this.adjustTCP_anomaly_count_out == orig(this.adjustTCP_anomaly_count_out)
this.adjustUDP_count_out == orig(this.adjustUDP_count_out)
this.adjustUDP_anomaly_count_out == orig(this.adjustUDP_anomaly_count_out)
this.TCP_portready_out == orig(this.TCP_portready_out)
this.UDP_portready_out == orig(this.UDP_portready_out)
this.adjustTCP_count_in == orig(this.adjustTCP_count_in)
this.adjustTCP_anomaly_count_in == orig(this.adjustTCP_anomaly_count_in)
this.adjustUDP_count_in == orig(this.adjustUDP_count_in)
this.adjustUDP_anomaly_count_in == orig(this.adjustUDP_anomaly_count_in)
this.TCP_portready_in == orig(this.TCP_portready_in)
this.UDP_portready_in == orig(this.UDP_portready_in)
this.adjustTCP_count_outsrc == orig(this.adjustTCP_count_outsrc)
this.adjustTCP_anomaly_count_outsrc == orig(this.adjustTCP_anomaly_count_outsrc)
this.adjustUDP_count_outsrc == orig(this.adjustUDP_count_outsrc)
this.adjustUDP_anomaly_count_outsrc == orig(this.adjustUDP_anomaly_count_outsrc)
this.TCP_portready_outsrc == orig(this.TCP_portready_outsrc)
this.UDP_portready_outsrc == orig(this.UDP_portready_outsrc)
this.sdf == orig(this.sdf)
this.fileModel == orig(this.fileModel)
this.fileModel.centroids == orig(this.fileModel.centroids)
this.fileModel.centroids[] == orig(this.fileModel.centroids[])
this.fileModel.NMax == orig(this.fileModel.NMax)
this.fileModel.b_packets == orig(this.fileModel.b_packets)
this.fileModel.b_packets[] == orig(this.fileModel.b_packets[])
this.fileModel.b_packets[] == orig(this.fileModel.b_nums[])
this.fileModel.b_nums == orig(this.fileModel.b_nums)
this.packetIndex == orig(this.packetIndex)
this.TCP == orig(this.TCP)
this.UDP == orig(this.UDP)
this.maliciousBuffer == orig(this.maliciousBuffer)
this.sign == orig(this.sign)
this.linkType == orig(this.MM.stablesize)
this.linkType == orig(this.linkType)
arg2[] == orig(arg2[])
(this.threshold < return) ==> (orig(this.threshold) one of 32.0, 64.0, 409.0)
(this.threshold < return) ==> (this.fileModel.NMax != this.packetIndex)
(this.threshold < return) ==> (this.packetIndex != this.TCP)
(this.threshold < return) ==> (this.packetIndex != this.UDP)
(this.threshold < return) ==> (this.threshold <= orig(this.threshold))
(this.threshold < return) ==> (this.threshold == 32.0)
(this.threshold < return) ==> (this.verbose != this.packetIndex)
(this.threshold > return) ==> (this.adjustTCP_anomaly_count[] > this.adjustUDP_count[] (lexically))
(this.threshold > return) ==> (this.adjustTCP_count[] > this.adjustUDP_count[] (lexically))
(this.threshold > return) ==> (this.packetIndex > this.linkType)
(this.threshold > return) ==> (this.verbose <= this.packetIndex)
this.port_trunc_length has only one value
this.port_trunc_length != null
this.port_keywords has only one value
this.port_keywords != null
this.port_keywords_range has only one value

this.port_keywords_range != null
this.string_compare_method has only one value
this.fpRate == 0.0010
this.host_ip has only one value
this.host_ip.toString == "128.59.16.101"
this.in_dst_ports_set has only one value
this.in_dst_ports_set != null
this.in_src_ports_set has only one value
this.in_src_ports_set != null
this.out_dst_ports_set has only one value
this.out_dst_ports_set != null
this.out_src_ports_set has only one value
this.out_src_ports_set != null
this.TCP_portready[] == [1, 1, 1,

this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_in.getClass())
this.MM.TCP_model.getClass() == orig(this.MM.UDPNormalLength_outsrc.getClass())
orig(arg0) in this.MM.TCP_model[]
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_in.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_out.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.TCP_lengthModel_outsrc.getClass())
this.MM.TCP_lengthModel.getClass() == orig(this.MM.UDP_lengthModel_outsrc.getClass())
arg2[] elements < this.MM.max_sample
this.MM.host_ip.toString == this.host_ip.toString
this.MM.stabledis < return
this.MM.stabledis < orig(this.threshold)
this.MM.lenProb.getClass() == orig(this.MM.lenProb.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_out.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_in.getClass())
this.MM.lenProb.getClass() == orig(this.MM.lenProb_outsrc.getClass())
this.MM.anomalyLengthThreshold[] elements < return
this.MM.anomalyLengthThreshold[] elements < orig(this.threshold)
this.MM.normal_length_perc < return
this.MM.normal_length_perc < orig(this.threshold)
this.kernel_h < return
this.kernel_h < orig(this.threshold)
this.string_compare_method.toString < arg1.toString
this.string_compare_method.toString == orig(this.string_compare_method.toString)
this.string_compare_threshold < return
orig(this.threshold)
this.string_compare_threshold < orig(this.threshold)
this.fpRate < return
this.fpRate < orig(this.threshold)
this.host_ip.toString < arg1.toString
this.host_ip.toString == orig(this.MM.host_ip.toString)
this.host_ip.toString == orig(this.host_ip.toString)
this.adjustTCP_count[] elements < this.packetIndex
this.adjustTCP_anomaly_count[] elements < this.packetIndex
arg2[] elements >= this.fileModel.NMax
this.fileModel.NMax in arg2[]
this.packetIndex >= this.linkType
arg2[] elements > this.TCP
arg2[] elements > this.linkType
arg1.toString == orig(arg1.toString)
return != orig(this.threshold)
==========
TestPacketHandler.vecContains(java.util.Vector, byte[]):::ENTER
arg0 has only one value
arg0 != null
arg0[] elements != null
arg0[].getClass() elements == byte[].class
arg1 != null
==========
TestPacketHandler.vecContains(java.util.Vector, byte[]):::EXIT979
==========
TestPacketHandler.vecContains(java.util.Vector, byte[]):::EXIT979;condition="not(return == true)"
==========
TestPacketHandler.vecContains(java.util.Vector, byte[]):::EXIT
arg0[] == orig(arg0[])
arg1[] == orig(arg1[])
arg0[] elements != null
arg0[].getClass() elements == byte[].class
return == false
==========

TestPacketHandler.vecContains(java.util.Vector, byte[]):::EXIT;condition="not(return == true)"
==========
TrainPacketHandler.byteToFloat(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
TrainPacketHandler.byteToFloat(byte[], int):::EXIT
arg0[] == orig(arg0[])
return != null
return[] elements >= 0.0
arg0[] elements > orig(arg1)
==========
TrainPacketHandler.byteToShort(byte[], int):::ENTER
arg0 != null
arg1 == -1
arg0[] elements > arg1
==========
TrainPacketHandler.byteToShort(byte[], int):::EXIT
arg0[] > return[] (lexically)
arg0[] >= return[] (lexically)
arg0[] == orig(arg0[])
return != null
return[] elements >= 0
arg0[] elements > orig(arg1)
return[] elements > orig(arg1)
==========
net.sourceforge.jpcap.capture.PacketCapture:::CLASS
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER != null
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == "jpcap"
net.sourceforge.jpcap.capture.PacketCapture.nextInstance one of 0, 1
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == 10
net.sourceforge.jpcap.capture.PacketCapture.nextInstance < net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
==========
net.sourceforge.jpcap.capture.PacketCapture:::OBJECT
this has only one value
this.instanceNum == 0
this._rcsid.toString == "$Id: PacketCapture.java,v 1.14 2003/06/24 16:54:27 pcharles Exp $"
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString > this._rcsid.toString
net.sourceforge.jpcap.capture.PacketCapture.nextInstance >= this.instanceNum
==========
net.sourceforge.jpcap.capture.PacketCapture.PacketCapture():::EXIT
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketCapture.capture(int):::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
count one of -1, 5000
==========
net.sourceforge.jpcap.capture.PacketCapture.capture(int):::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketCapture.close():::ENTER

net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketCapture.close():::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketCapture.openOffline(java.lang.String):::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
fileName has only one value
fileName.toString == "test-p3.dmp"
==========
net.sourceforge.jpcap.capture.PacketCapture.openOffline(java.lang.String):::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
fileName.toString == "test-p3.dmp"
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
fileName.toString == orig(fileName.toString)
==========
net.sourceforge.jpcap.capture.PacketCapture.setFilter(java.lang.String, boolean):::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
filterExpression has only one value
filterExpression.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst
port 80))))"
optimize == true
==========
net.sourceforge.jpcap.capture.PacketCapture.setFilter(java.lang.String, boolean):::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
this.instanceNum == orig(this.instanceNum)
this._rcsid == orig(this._rcsid)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 1
this._rcsid has only one value
filterExpression.toString == "ip[9]=6 and ((dst host 128.59.16.101 and ((dst port 80))) or (src host 128.59.16.101 and ((dst
port 80))))"
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
this._rcsid.toString == orig(this._rcsid.toString)
filterExpression.toString == orig(filterExpression.toString)
==========
net.sourceforge.jpcap.capture.PacketCapture.this():::ENTER
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == this.instanceNum
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 0
this._rcsid == null
==========
net.sourceforge.jpcap.capture.PacketCapture.this():::EXIT
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == this.instanceNum

net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(net.sourceforge.jpcap.capture.PacketCapture.nextInstance)
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == orig(this.instanceNum)
net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX == orig(net.sourceforge.jpcap.capture.PacketCapture.INSTANCE_MAX
net.sourceforge.jpcap.capture.PacketCapture.nextInstance == 0
this._rcsid has only one value
net.sourceforge.jpcap.capture.PacketCapture.LIB_PCAP_WRAPPER.toString == orig(net.sourceforge.jpcap.capture.PacketCapture.LIB
==========
net.sourceforge.jpcap.capture.PacketCaptureBase:::OBJECT
this.receivedCount == this.droppedCount
this has only one value
this.getClass() == net.sourceforge.jpcap.capture.PacketCapture.class
this.linkType one of 0, 1
this.receivedCount == 0
this._rcsid.toString == "$Id: PacketCaptureBase.java,v 1.4 2001/07/29 20:13:08 pcharles Exp $"
this.linkType >= this.receivedCount
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.PacketCaptureBase():::EXIT
this.linkType == this.receivedCount
this.linkType == this.droppedCount
this.linkType == 0
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.handlePacket(int, int, int, int, byte[]):::ENTER
length == caplen
this.linkType == 1
this._rcsid has only one value
this._rcsid != null
data != null
this.linkType < length
this.linkType < seconds
this.linkType < useconds
this.linkType in data[]
this.receivedCount < length
this.receivedCount < seconds
this.receivedCount < useconds
this.receivedCount in data[]
length < seconds
length < useconds
data[] elements < length
seconds > useconds
data[] elements < seconds
data[] elements < useconds
7190749 * length + 3528 * seconds - 80 * useconds - 3.89898564504E12 == 0
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.handlePacket(int, int, int, int, byte[]):::EXIT
this.linkType == orig(this.linkType)
this.receivedCount == orig(this.receivedCount)
this.receivedCount == orig(this.droppedCount)
this._rcsid == orig(this._rcsid)
data[] == orig(data[])
this.linkType == 1
this._rcsid has only one value
this._rcsid != null
this.linkType in data[]
this.linkType < orig(length)
this.linkType < orig(seconds)
this.linkType < orig(useconds)
this.receivedCount in data[]
this.receivedCount < orig(length)
this.receivedCount < orig(seconds)
this.receivedCount < orig(useconds)
this._rcsid.toString == orig(this._rcsid.toString)
data[] elements < orig(length)

data[] elements < orig(seconds)
data[] elements < orig(useconds)
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.this():::ENTER
this.linkType == this.receivedCount
this.linkType == this.droppedCount
this.linkType == 0
this._rcsid == null
==========
net.sourceforge.jpcap.capture.PacketCaptureBase.this():::EXIT
this.linkType == this.receivedCount
this.linkType == this.droppedCount
this.linkType == orig(this.linkType)
this.linkType == orig(this.receivedCount)
this.linkType == orig(this.droppedCount)
this.linkType == 0
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketDispatcher:::OBJECT
this has only one value
this.getClass() == net.sourceforge.jpcap.capture.PacketCapture.class
this._rcsid.toString == "$Id: PacketDispatcher.java,v 1.2 2001/06/18 05:02:26 pcharles Exp $"
==========
net.sourceforge.jpcap.capture.PacketDispatcher.PacketDispatcher():::EXIT
this.rawListeners has only one value
this.objListeners has only one value
this._rcsid has only one value
==========
net.sourceforge.jpcap.capture.PacketDispatcher.addRawPacketListener(net.sourceforge.jpcap.capture.RawPacketListener):::ENTER
this.rawListeners has only one value
this.objListeners has only one value
this._rcsid has only one value
rawListener has only one value
rawListener.getClass() == TestPacketHandler.class
==========
net.sourceforge.jpcap.capture.PacketDispatcher.addRawPacketListener(net.sourceforge.jpcap.capture.RawPacketListener):::EXIT
this.rawListeners == orig(this.rawListeners)
this.objListeners == orig(this.objListeners)
this._rcsid == orig(this._rcsid)
this.rawListeners has only one value
this.objListeners has only one value
this._rcsid has only one value
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchPacket(net.sourceforge.jpcap.net.Packet):::ENTER
this.rawListeners has only one value
this.rawListeners != null
this.objListeners has only one value
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
packet != null
packet._rcsid has only one value
packet._rcsid != null
packet._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString > packet._rcsid.toString
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchPacket(net.sourceforge.jpcap.net.Packet):::EXIT
this.rawListeners == orig(this.rawListeners)
this.objListeners == orig(this.objListeners)
this._rcsid == orig(this._rcsid)
packet._rcsid == orig(packet._rcsid)
this.rawListeners has only one value

this.rawListeners != null
this.objListeners has only one value
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
packet._rcsid has only one value
packet._rcsid != null
packet._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString > packet._rcsid.toString
this._rcsid.toString == orig(this._rcsid.toString)
packet._rcsid.toString == orig(packet._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchRawPacket(net.sourceforge.jpcap.net.RawPacket):::ENTER
this.rawListeners has only one value
this.rawListeners != null
this.objListeners has only one value
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
rawPacket != null
rawPacket.timeval != null
rawPacket.timeval._rcsid has only one value
rawPacket.timeval._rcsid != null
rawPacket.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
rawPacket.bytes != null
rawPacket.droplen == 0
rawPacket._rcsid has only one value
rawPacket._rcsid != null
rawPacket._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString < rawPacket.timeval._rcsid.toString
this._rcsid.toString < rawPacket._rcsid.toString
rawPacket.timeval.seconds > rawPacket.timeval.microseconds
rawPacket.bytes[] elements < rawPacket.timeval.seconds
rawPacket.timeval.seconds > rawPacket.droplen
rawPacket.bytes[] elements < rawPacket.timeval.microseconds
rawPacket.timeval.microseconds > rawPacket.droplen
rawPacket.timeval._rcsid.toString > rawPacket._rcsid.toString
rawPacket.droplen in rawPacket.bytes[]
==========
net.sourceforge.jpcap.capture.PacketDispatcher.dispatchRawPacket(net.sourceforge.jpcap.net.RawPacket):::EXIT
this.rawListeners == orig(this.rawListeners)
this.objListeners == orig(this.objListeners)
this._rcsid == orig(this._rcsid)
rawPacket.timeval == orig(rawPacket.timeval)
rawPacket.timeval.seconds == orig(rawPacket.timeval.seconds)
rawPacket.timeval.microseconds == orig(rawPacket.timeval.microseconds)
rawPacket.timeval._rcsid == orig(rawPacket.timeval._rcsid)
rawPacket.bytes == orig(rawPacket.bytes)
rawPacket.bytes[] == orig(rawPacket.bytes[])
rawPacket.droplen == orig(rawPacket.droplen)
rawPacket._rcsid == orig(rawPacket._rcsid)
this.rawListeners has only one value
this.rawListeners != null
this.objListeners has only one value
this.objListeners != null
this._rcsid has only one value
this._rcsid != null
rawPacket.timeval != null
rawPacket.timeval._rcsid has only one value
rawPacket.timeval._rcsid != null
rawPacket.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
rawPacket.bytes != null
rawPacket.droplen == 0

rawPacket._rcsid has only one value
rawPacket._rcsid != null
rawPacket._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this._rcsid.toString < rawPacket.timeval._rcsid.toString
this._rcsid.toString < rawPacket._rcsid.toString
this._rcsid.toString == orig(this._rcsid.toString)
rawPacket.timeval.seconds > rawPacket.timeval.microseconds
rawPacket.bytes[] elements < rawPacket.timeval.seconds
rawPacket.timeval.seconds > rawPacket.droplen
rawPacket.bytes[] elements < rawPacket.timeval.microseconds
rawPacket.timeval.microseconds > rawPacket.droplen
rawPacket.timeval._rcsid.toString > rawPacket._rcsid.toString
rawPacket.timeval._rcsid.toString == orig(rawPacket.timeval._rcsid.toString)
rawPacket.droplen in rawPacket.bytes[]
rawPacket._rcsid.toString == orig(rawPacket._rcsid.toString)
==========
net.sourceforge.jpcap.capture.PacketDispatcher.this():::ENTER
this.rawListeners == this.objListeners
this.rawListeners == null
this._rcsid == null
==========
net.sourceforge.jpcap.capture.PacketDispatcher.this():::EXIT
this.rawListeners has only one value
this.objListeners has only one value
this._rcsid has only one value
==========
net.sourceforge.jpcap.net.EthernetPacket:::OBJECT
this._ethOffset == this._ethernetHeaderLength
this._ethernetHeaderBytes == this._ethernetDataBytes
this._sourceHwAddress == this._destinationHwAddress
this._ethOffset one of 0, 14
this._ethernetHeaderBytes == null
this._sourceHwAddress == null
this._etherProtocol == 0
this._etherProtocolSet == false
this._rcsid.toString == "$Id: EthernetPacket.java,v 1.17 2003/06/24 23:09:49 pcharles Exp $"
this._etherProtocol in this._bytes[]
this._ethOffset >= this._etherProtocol
==========
net.sourceforge.jpcap.net.EthernetPacket.EthernetPacket(int, byte[]):::ENTER
lLen == 14
bytes != null
==========
net.sourceforge.jpcap.net.EthernetPacket.EthernetPacket(int, byte[]):::EXIT
this._bytes == orig(bytes)
this._bytes[] == bytes[]
this._ethOffset == orig(lLen)
bytes[] == orig(bytes[])
this._bytes != null
this._ethOffset == 14
this._rcsid has only one value
this._rcsid != null
this._etherProtocol in bytes[]
==========
net.sourceforge.jpcap.net.EthernetPacket.this():::ENTER
this._bytes == this._ethernetHeaderBytes
this._bytes == this._ethernetDataBytes
this._ethOffset == this._etherProtocol
this._sourceHwAddress == this._rcsid
this._bytes == null
this._ethOffset == 0
==========
net.sourceforge.jpcap.net.EthernetPacket.this():::EXIT

this._bytes == this._ethernetHeaderBytes
this._bytes == this._ethernetDataBytes
this._bytes == orig(this._bytes)
this._bytes == orig(this._ethernetHeaderBytes)
this._bytes == orig(this._ethernetDataBytes)
this._ethOffset == this._etherProtocol
this._ethOffset == orig(this._ethOffset)
this._ethOffset == orig(this._ethernetHeaderLength)
this._ethOffset == orig(this._etherProtocol)
this._sourceHwAddress == orig(this._sourceHwAddress)
this._sourceHwAddress == orig(this._destinationHwAddress)
this._sourceHwAddress == orig(this._rcsid)
this._etherProtocolSet == orig(this._etherProtocolSet)
this._bytes == null
this._ethOffset == 0
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.IPAddress:::CLASS
==========
net.sourceforge.jpcap.net.IPAddress.extract(int, byte[]):::ENTER
offset one of 26, 30
bytes != null
net.sourceforge.jpcap.net.IPAddress.WIDTH < offset
==========
net.sourceforge.jpcap.net.IPAddress.extract(int, byte[]):::EXIT
bytes[] == orig(bytes[])
return != null
==========
net.sourceforge.jpcap.net.IPAddress.toString(int):::ENTER
address != 0
net.sourceforge.jpcap.net.IPAddress.WIDTH != address
==========
net.sourceforge.jpcap.net.IPAddress.toString(int):::EXIT
return != null
==========
net.sourceforge.jpcap.net.IPPacket:::OBJECT
this._version == this._typeOfService
this._version == this._id
this._version == this._fragmentFlags
this._version == this._fragmentOffset
this._version == this._timeToLive
this._version == this._ipChecksum
this._version == this._sourceAddressAsLong
this._version == this._destinationAddressAsLong
this._versionSet == this._typeOfServiceSet
this._versionSet == this._idSet
this._versionSet == this._fragmentFlagsSet
this._versionSet == this._fragmentOffsetSet
this._versionSet == this._timeToLiveSet
this._versionSet == this._ipChecksumSet
this._versionSet == this._sourceAddressAsLongSet
this._versionSet == this._destinationAddressAsLongSet
this._versionSet == this._isValidChecksum
this._versionSet == this._isValidChecksumSet
this._sourceAddressBytes == this._destinationAddressBytes
this._sourceAddressBytes == this._ipHeaderBytes
this._sourceAddressBytes == this._ipDataBytes
this._ipOffset one of 0, 34
this._version == 0
this._versionSet == false
this._ipHeaderLength one of 0, 20
this._length >= 0

this._ipProtocol one of 0, 6
this._sourceAddressBytes == null
this._destinationAddress.toString == "128.59.16.101"
this._rcsid.toString == "$Id: IPPacket.java,v 1.20 2003/07/02 18:41:10 pcharles Exp $"
this._ipOffset >= this._version
this._ipOffset >= this._ipHeaderLength
this._ipOffset >= this._ipProtocol
this._version <= this._ipHeaderLength
this._version <= this._length
this._version <= this._ipProtocol
this._ipHeaderLength >= this._ipProtocol
this._length >= this._ipProtocol
this._sourceAddress.toString > this._destinationAddress.toString
this._sourceAddress.toString > this._rcsid.toString
this._destinationAddress.toString > this._rcsid.toString
==========
net.sourceforge.jpcap.net.IPPacket.IPPacket(int, byte[]):::ENTER
lLen == 14
bytes != null
==========
net.sourceforge.jpcap.net.IPPacket.IPPacket(int, byte[]):::EXIT
this._version == this._length
this._version == this._ipProtocol
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
bytes[] == orig(bytes[])
this._ipOffset == 34
this._ipHeaderLength == 20
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._version in bytes[]
==========
net.sourceforge.jpcap.net.IPPacket.getDestinationAddress():::ENTER
this._lengthSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
==========
net.sourceforge.jpcap.net.IPPacket.getDestinationAddress():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)

this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._lengthSet == orig(this._ipProtocolSet)
this._ipProtocol == orig(this._ipProtocol)
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._destinationAddress == return
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return != null
return.toString == ”128.59.16.101”
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
this._destinationAddress.toString == return.toString
this._rcsid.toString < return.toString
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getIPHeaderLength():::ENTER
this._version == this._ipProtocol
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
==========
net.sourceforge.jpcap.net.IPPacket.getIPHeaderLength():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == this._ipProtocol
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)

this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == return
this._length == orig(this._length)
this._lengthSet == orig(this._lengthSet)
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._ipHeaderLength)
this._ipOffset == 34
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return == 20
this._ipOffset < this._length
this._version < this._length
this._length > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getIPProtocol():::ENTER
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._ipOffset > this._ipProtocol
this._version < this._length
this._ipHeaderLength < this._length
this._ipHeaderLength > this._ipProtocol
this._length > this._ipProtocol
==========
net.sourceforge.jpcap.net.IPPacket.getIPProtocol():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)

this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._ipProtocol == return
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._destinationAddress == orig(this._destinationAddress)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._rcsid has only one value
this._rcsid != null
return == 6
this._ipOffset < this._length
this._ipOffset > orig(this._ipProtocol)
this._version < this._length
this._version <= orig(this._ipProtocol)
this._ipHeaderLength < this._length
this._ipHeaderLength > orig(this._ipProtocol)
this._length > return
this._length > orig(this._ipProtocol)
this._sourceAddress.toString == orig(this._sourceAddress.toString)
this._destinationAddress.toString == orig(this._destinationAddress.toString)
this._rcsid.toString == orig(this._rcsid.toString)
return >= orig(this._ipProtocol)
==========
net.sourceforge.jpcap.net.IPPacket.getIpHeaderLength():::ENTER
this._version == this._ipProtocol
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
==========
net.sourceforge.jpcap.net.IPPacket.getIpHeaderLength():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == this._ipProtocol
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)

this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == return
this._length == orig(this._length)
this._lengthSet == orig(this._lengthSet)
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._ipHeaderLength)
this._ipOffset == 34
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
return == 20
this._ipOffset < this._length
this._version < this._length
this._length > return
this._rcsid.toString == orig(this._rcsid.toString)
===
net.sourceforge.jpcap.net.IPPacket.getLength():::ENTER
this._version == this._length
this._version == this._ipProtocol
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._sourceAddress == this._destinationAddress
this._ipOffset == 34
this._ipHeaderLength == 20
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
===
net.sourceforge.jpcap.net.IPPacket.getLength():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == this._ipProtocol
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._length)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)

this._version == orig(this._timeToLive)
this._version == orig(this._ipProtocol)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._lengthSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == return
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < return
this._version < return
this._ipHeaderLength < return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.getProtocol():::ENTER
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._ipOffset > this._ipProtocol
this._version < this._length
this._ipHeaderLength < this._length
this._ipHeaderLength > this._ipProtocol
this._length > this._ipProtocol
==========
net.sourceforge.jpcap.net.IPPacket.getProtocol():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)
this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)

this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._ipProtocol == return
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._destinationAddress == orig(this._destinationAddress)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._rcsid has only one value
this._rcsid != null
return == 6
this._ipOffset < this._length
this._ipOffset > orig(this._ipProtocol)
this._version < this._length
this._version <= orig(this._ipProtocol)
this._ipHeaderLength < this._length
this._ipHeaderLength > orig(this._ipProtocol)
this._length > return
this._length > orig(this._ipProtocol)
this._sourceAddress.toString == orig(this._sourceAddress.toString)
this._destinationAddress.toString == orig(this._destinationAddress.toString)
this._rcsid.toString == orig(this._rcsid.toString)
return >= orig(this._ipProtocol)
==========
net.sourceforge.jpcap.net.IPPacket.getSourceAddress():::ENTER
this._lengthSet == this._ipProtocolSet
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._sourceAddress == null
this._destinationAddress != null
this._rcsid has only one value
this._rcsid != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
==========
net.sourceforge.jpcap.net.IPPacket.getSourceAddress():::EXIT
this._ipOffset == orig(this._ipOffset)
this._version == orig(this._version)
this._version == orig(this._typeOfService)
this._version == orig(this._id)

this._version == orig(this._fragmentFlags)
this._version == orig(this._fragmentOffset)
this._version == orig(this._timeToLive)
this._version == orig(this._ipChecksum)
this._version == orig(this._sourceAddressAsLong)
this._version == orig(this._destinationAddressAsLong)
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._ipHeaderLength == orig(this._ipHeaderLength)
this._length == orig(this._length)
this._lengthSet == this._ipProtocolSet
this._lengthSet == orig(this._lengthSet)
this._lengthSet == orig(this._ipProtocolSet)
this._ipProtocol == orig(this._ipProtocol)
this._sourceAddress == return
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._destinationAddress == orig(this._destinationAddress)
this._rcsid == orig(this._rcsid)
this._ipOffset == 34
this._ipHeaderLength == 20
this._lengthSet == true
this._ipProtocol == 6
this._destinationAddress != null
this._rcsid has only one value
this._rcsid != null
return != null
this._ipOffset < this._length
this._version < this._length
this._ipHeaderLength < this._length
this._length > this._ipProtocol
this._sourceAddress.toString == return.toString
this._destinationAddress.toString < return.toString
this._destinationAddress.toString == orig(this._destinationAddress.toString)
this._rcsid.toString < return.toString
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.IPPacket.this():::ENTER
this._ipOffset == this._version
this._ipOffset == this._ipHeaderLength
this._ipOffset == this._typeOfService
this._ipOffset == this._length
this._ipOffset == this._id
this._ipOffset == this._fragmentFlags
this._ipOffset == this._fragmentOffset
this._ipOffset == this._timeToLive
this._ipOffset == this._ipProtocol
this._ipOffset == this._ipChecksum
this._ipOffset == this._sourceAddressAsLong
this._ipOffset == this._destinationAddressAsLong
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet

this._sourceAddress == this._destinationAddress
this._sourceAddress == this._rcsid
this._ipOffset == 0
this._sourceAddress == null
==========
net.sourceforge.jpcap.net.IPPacket.this():::EXIT
this._ipOffset == this._version
this._ipOffset == this._ipHeaderLength
this._ipOffset == this._typeOfService
this._ipOffset == this._length
this._ipOffset == this._id
this._ipOffset == this._fragmentFlags
this._ipOffset == this._fragmentOffset
this._ipOffset == this._timeToLive
this._ipOffset == this._ipProtocol
this._ipOffset == this._ipChecksum
this._ipOffset == this._sourceAddressAsLong
this._ipOffset == this._destinationAddressAsLong
this._ipOffset == orig(this._ipOffset)
this._ipOffset == orig(this._version)
this._ipOffset == orig(this._ipHeaderLength)
this._ipOffset == orig(this._typeOfService)
this._ipOffset == orig(this._length)
this._ipOffset == orig(this._id)
this._ipOffset == orig(this._fragmentFlags)
this._ipOffset == orig(this._fragmentOffset)
this._ipOffset == orig(this._timeToLive)
this._ipOffset == orig(this._ipProtocol)
this._ipOffset == orig(this._ipChecksum)
this._ipOffset == orig(this._sourceAddressAsLong)
this._ipOffset == orig(this._destinationAddressAsLong)
this._versionSet == this._lengthSet
this._versionSet == this._ipProtocolSet
this._versionSet == orig(this._versionSet)
this._versionSet == orig(this._typeOfServiceSet)
this._versionSet == orig(this._lengthSet)
this._versionSet == orig(this._idSet)
this._versionSet == orig(this._fragmentFlagsSet)
this._versionSet == orig(this._fragmentOffsetSet)
this._versionSet == orig(this._timeToLiveSet)
this._versionSet == orig(this._ipProtocolSet)
this._versionSet == orig(this._ipChecksumSet)
this._versionSet == orig(this._sourceAddressAsLongSet)
this._versionSet == orig(this._destinationAddressAsLongSet)
this._versionSet == orig(this._isValidChecksum)
this._versionSet == orig(this._isValidChecksumSet)
this._sourceAddress == this._destinationAddress
this._sourceAddress == orig(this._sourceAddress)
this._sourceAddress == orig(this._destinationAddress)
this._sourceAddress == orig(this._rcsid)
this._sourceAddressBytes == orig(this._sourceAddressBytes)
this._sourceAddressBytes == orig(this._destinationAddressBytes)
this._sourceAddressBytes == orig(this._ipHeaderBytes)
this._sourceAddressBytes == orig(this._ipDataBytes)
this._ipOffset == 0
this._sourceAddress == null
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.IPProtocol:::CLASS
net.sourceforge.jpcap.net.IPProtocol.messages has only one value
net.sourceforge.jpcap.net.IPProtocol.messages != null
==========

net.sourceforge.jpcap.net.IPProtocol.extractProtocol(int, byte[]):::ENTER
lLen == 14
packetBytes != null
==========
net.sourceforge.jpcap.net.IPProtocol.extractProtocol(int, byte[]):::EXIT
net.sourceforge.jpcap.net.IPProtocol.messages == orig(net.sourceforge.jpcap.net.IPProtocol.messages)
packetBytes[] == orig(packetBytes[])
return == 6
return in packetBytes[]
==========
net.sourceforge.jpcap.net.LinkLayer:::CLASS
net.sourceforge.jpcap.net.LinkLayer.descriptions has only one value
net.sourceforge.jpcap.net.LinkLayer.descriptions != null
==========
net.sourceforge.jpcap.net.LinkLayer.getLinkLayerLength(int):::ENTER
layerType == 1
==========
net.sourceforge.jpcap.net.LinkLayer.getLinkLayerLength(int):::EXIT59
==========
net.sourceforge.jpcap.net.LinkLayer.getLinkLayerLength(int):::EXIT
net.sourceforge.jpcap.net.LinkLayer.descriptions == orig(net.sourceforge.jpcap.net.LinkLayer.descriptions)
return == 14
==========
net.sourceforge.jpcap.net.LinkLayer.getProtoOffset(int):::ENTER
layerType == 1
==========
net.sourceforge.jpcap.net.LinkLayer.getProtoOffset(int):::EXIT100
==========
net.sourceforge.jpcap.net.LinkLayer.getProtoOffset(int):::EXIT
net.sourceforge.jpcap.net.LinkLayer.descriptions == orig(net.sourceforge.jpcap.net.LinkLayer.descriptions)
return == 12
==========
net.sourceforge.jpcap.net.Packet:::OBJECT
this._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
==========
net.sourceforge.jpcap.net.Packet.Packet():::EXIT
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.Packet.this():::ENTER
this._rcsid == null
==========
net.sourceforge.jpcap.net.Packet.this():::EXIT
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.PacketEncoding:::CLASS
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY has only one value
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY != null
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] == []
==========
net.sourceforge.jpcap.net.PacketEncoding.extractData(int, int, byte[], int):::ENTER
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] < bytes[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] <= bytes[] (lexically)
offset == 34
headerLen one of 20, 32
bytes != null
offset > headerLen
offset < dataLength
headerLen < dataLength
bytes[] elements < dataLength
==========
net.sourceforge.jpcap.net.PacketEncoding.extractData(int, int, byte[], int):::EXIT108

```
==========================================================================
net.sourceforge.jpcap.net.PacketEncoding.extractData(int, int, byte[], int):::EXIT
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY == orig(net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_A
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] < bytes[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] <= bytes[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] < return[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] <= return[] (lexically)
net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_ARRAY[] == orig(net.sourceforge.jpcap.net.PacketEncoding.EMPTY_BYTE_
bytes[] < return[] (lexically)
bytes[] <= return[] (lexically)
bytes[] == orig(bytes[])
return != null
bytes[] elements < orig(dataLength)
return[] elements < orig(dataLength)
==========================================================================
net.sourceforge.jpcap.net.PacketFactory:::CLASS
net.sourceforge.jpcap.net.PacketFactory.lLen one of  0, 14
==========================================================================
net.sourceforge.jpcap.net.PacketFactory.dataToPacket(int, byte[]):::ENTER
linkType == 1
bytes != null
net.sourceforge.jpcap.net.PacketFactory.lLen != linkType
linkType in bytes[]
==========================================================================
net.sourceforge.jpcap.net.PacketFactory.dataToPacket(int, byte[]):::EXIT58
==========================================================================
net.sourceforge.jpcap.net.PacketFactory.dataToPacket(int, byte[]):::EXIT
bytes[] == orig(bytes[])
net.sourceforge.jpcap.net.PacketFactory.lLen == 14
return != null
return._rcsid has only one value
return._rcsid != null
return._rcsid.toString == "$Id: Packet.java,v 1.7 2003/06/24 23:09:49 pcharles Exp $"
net.sourceforge.jpcap.net.PacketFactory.lLen >= orig(net.sourceforge.jpcap.net.PacketFactory.lLen)
orig(linkType) in bytes[]
==========================================================================
net.sourceforge.jpcap.net.RawPacket:::OBJECT
this.timeval._rcsid has only one value
this.timeval._rcsid != null
this.timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.droplen == 0
this._rcsid.toString == "$Id: RawPacket.java,v 1.3 2003/06/24 23:09:49 pcharles Exp $"
this.timeval.seconds > this.timeval.microseconds
this.bytes[] elements < this.timeval.seconds
this.timeval.seconds > this.droplen
this.bytes[] elements < this.timeval.microseconds
this.timeval.microseconds > this.droplen
this.timeval._rcsid.toString > this._rcsid.toString
this.droplen in this.bytes[]
==========================================================================
net.sourceforge.jpcap.net.RawPacket.RawPacket(net.sourceforge.jpcap.util.Timeval, byte[], int):::ENTER
timeval != null
timeval._rcsid has only one value
timeval._rcsid != null
timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
bytes != null
droplen == 0
timeval.seconds > timeval.microseconds
bytes[] elements < timeval.seconds
timeval.seconds > droplen
bytes[] elements < timeval.microseconds
timeval.microseconds > droplen
droplen in bytes[]
```

```
==========
net.sourceforge.jpcap.net.RawPacket.RawPacket(net.sourceforge.jpcap.util.Timeval, byte[], int):::EXIT
this.timeval == orig(timeval)
this.timeval.seconds == timeval.seconds
this.timeval.microseconds == timeval.microseconds
this.timeval._rcsid == timeval._rcsid
this.bytes == orig(bytes)
this.bytes[] == bytes[]
this.droplen == orig(droplen)
timeval.seconds == orig(timeval.seconds)
timeval.microseconds == orig(timeval.microseconds)
timeval._rcsid == orig(timeval._rcsid)
bytes[] == orig(bytes[])
this.timeval != null
this.bytes != null
this._rcsid has only one value
this._rcsid != null
timeval._rcsid has only one value
timeval._rcsid != null
timeval._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.timeval._rcsid.toString == timeval._rcsid.toString
this.droplen < timeval.seconds
this.droplen < timeval.microseconds
this.droplen in bytes[]
this._rcsid.toString < timeval._rcsid.toString
timeval.seconds > timeval.microseconds
bytes[] elements < timeval.seconds
bytes[] elements < timeval.microseconds
timeval._rcsid.toString == orig(timeval._rcsid.toString)
==========
net.sourceforge.jpcap.net.RawPacket.getData():::ENTER
this.timeval != null
this.bytes != null
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.RawPacket.getData():::EXIT
this.timeval == orig(this.timeval)
this.timeval.seconds == orig(this.timeval.seconds)
this.timeval.microseconds == orig(this.timeval.microseconds)
this.timeval._rcsid == orig(this.timeval._rcsid)
this.bytes == return
this.bytes[] == return[]
this.droplen == orig(this.droplen)
this._rcsid == orig(this._rcsid)
return == orig(this.bytes)
return[] == orig(this.bytes[])
this.timeval != null
this._rcsid has only one value
this._rcsid != null
return != null
return[] elements < this.timeval.seconds
return[] elements < this.timeval.microseconds
this.timeval._rcsid.toString == orig(this.timeval._rcsid.toString)
this.droplen in return[]
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.RawPacket.getTimeval():::ENTER
this.timeval != null
this.bytes != null
this._rcsid has only one value
this._rcsid != null
==========
```

net.sourceforge.jpcap.net.RawPacket.getTimeval():::EXIT
this.timeval == return
this.timeval.seconds == return.seconds
this.timeval.microseconds == return.microseconds
this.timeval._rcsid == return._rcsid
this.bytes == orig(this.bytes)
this.bytes[] == orig(this.bytes[])
this.droplen == orig(this.droplen)
this._rcsid == orig(this._rcsid)
return == orig(this.timeval)
return.seconds == orig(this.timeval.seconds)
return.microseconds == orig(this.timeval.microseconds)
return._rcsid == orig(this.timeval._rcsid)
this.bytes != null
this._rcsid has only one value
this._rcsid != null
return != null
return._rcsid has only one value
return._rcsid != null
return._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.timeval._rcsid.toString == return._rcsid.toString
this.bytes[] elements < return.seconds
this.bytes[] elements < return.microseconds
this.droplen < return.seconds
this.droplen < return.microseconds
this._rcsid.toString < return._rcsid.toString
this._rcsid.toString == orig(this._rcsid.toString)
return.seconds > return.microseconds
return._rcsid.toString == orig(this.timeval._rcsid.toString)
==========
net.sourceforge.jpcap.net.RawPacket.this():::ENTER
this.timeval == null
this.bytes == null
this._rcsid == null
==========
net.sourceforge.jpcap.net.RawPacket.this():::EXIT
this.timeval == orig(this.timeval)
this.bytes == orig(this.bytes)
this.droplen == orig(this.droplen)
this.timeval == null
this.bytes == null
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.net.TCPPacket:::OBJECT
this._sequenceNumber == this._acknowledgmentNumber
this._sequenceNumber == this._windowSize
this._sequenceNumber == this._tcpChecksum
this._sequenceNumber == this._urgentPointer
this._sequenceNumber == this._allFlags
this._sequenceNumberSet == this._acknowledgmentNumberSet
this._sequenceNumberSet == this._windowSizeSet
this._sequenceNumberSet == this._tcpChecksumSet
this._sequenceNumberSet == this._urgentPointerSet
this._sequenceNumberSet == this._allFlagsSet
this._sequenceNumberSet == this._isUrg
this._sequenceNumberSet == this._isUrgSet
this._sequenceNumberSet == this._isAck
this._sequenceNumberSet == this._isAckSet
this._sequenceNumberSet == this._isPsh
this._sequenceNumberSet == this._isPshSet
this._sequenceNumberSet == this._isRst
this._sequenceNumberSet == this._isRstSet

this._sequenceNumberSet == this._isSyn
this._sequenceNumberSet == this._isSynSet
this._sequenceNumberSet == this._isFin
this._sequenceNumberSet == this._isFinSet
this._sourcePort >= 0
this._destinationPort one of 0, 80
this._sequenceNumber == 0
this._sequenceNumberSet == false
this._tcpHeaderLength one of 0, 20, 32
this._payloadDataLength >= 0
this._tcpHeaderBytes == null
this._rcsid.toString == "$Id: TCPPacket.java,v 1.20 2003/07/02 18:43:05 pcharles Exp $"
this._sourcePort >= this._sequenceNumber
this._destinationPort >= this._sequenceNumber
this._destinationPort <= this._payloadDataLength
this._sequenceNumber <= this._tcpHeaderLength
this._sequenceNumber <= this._payloadDataLength
this._tcpDataBytes[] elements > this._sequenceNumber
this._tcpHeaderLength <= this._payloadDataLength
this._tcpDataBytes[] elements < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.TCPPacket(int, byte[]):::ENTER
lLen == 14
bytes != null
==========
net.sourceforge.jpcap.net.TCPPacket.TCPPacket(int, byte[]):::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
bytes[] == orig(bytes[])
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._sourcePort in bytes[]
this._tcpHeaderLength < this._payloadDataLength

this._tcpHeaderLength > orig(lLen)
bytes[] elements < this._payloadDataLength
this._payloadDataLength > orig(lLen)
==========
net.sourceforge.jpcap.net.TCPPacket.getData():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getData():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet

this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._tcpDataBytes == return
this._tcpDataBytes[] == return[]
this._rcsid == orig(this._rcsid)
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
return != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
return[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
return[] elements < this._payloadDataLength
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getDestinationPort():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet

this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpDataBytes[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getDestinationPort():::EXIT
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)

this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._destinationPort == return
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpDataBytes == orig(this._tcpDataBytes)
this._tcpDataBytes[] == orig(this._tcpDataBytes[])
this._rcsid == orig(this._rcsid)
this._sourcePort == 0
this._sourcePortSet == false
this._destinationPortSet == true
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
return == 80
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpDataBytes[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
this._tcpHeaderLength < return
this._payloadDataLength > return
return in this._tcpDataBytes[]
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getPayloadDataLength():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet

this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getPayloadDataLength():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)

this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == return
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._payloadDataLength)
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < return
this._tcpHeaderLength < return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getSourcePort():::ENTER
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePort == 0
this._sourcePortSet == false
this._destinationPort == 80
this._destinationPortSet == true
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpDataBytes[] elements > this._sourcePort
this._destinationPort > this._tcpHeaderLength
this._destinationPort < this._payloadDataLength

this._destinationPort in this._tcpDataBytes[]
this._tcpHeaderLength < this._payloadDataLength
===
net.sourceforge.jpcap.net.TCPPacket.getSourcePort():::EXIT
this._sourcePort == return
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == orig(this._destinationPortSet)
this._destinationPort == orig(this._destinationPort)
this._sequenceNumber == orig(this._sourcePort)
this._sequenceNumber == orig(this._sequenceNumber)
this._sequenceNumber == orig(this._acknowledgmentNumber)
this._sequenceNumber == orig(this._windowSize)
this._sequenceNumber == orig(this._tcpChecksum)
this._sequenceNumber == orig(this._urgentPointer)
this._sequenceNumber == orig(this._allFlags)
this._sequenceNumberSet == orig(this._sourcePortSet)
this._sequenceNumberSet == orig(this._sequenceNumberSet)
this._sequenceNumberSet == orig(this._acknowledgmentNumberSet)
this._sequenceNumberSet == orig(this._windowSizeSet)
this._sequenceNumberSet == orig(this._tcpChecksumSet)
this._sequenceNumberSet == orig(this._urgentPointerSet)
this._sequenceNumberSet == orig(this._allFlagsSet)
this._sequenceNumberSet == orig(this._isUrg)
this._sequenceNumberSet == orig(this._isUrgSet)
this._sequenceNumberSet == orig(this._isAck)
this._sequenceNumberSet == orig(this._isAckSet)
this._sequenceNumberSet == orig(this._isPsh)
this._sequenceNumberSet == orig(this._isPshSet)
this._sequenceNumberSet == orig(this._isRst)
this._sequenceNumberSet == orig(this._isRstSet)
this._sequenceNumberSet == orig(this._isSyn)
this._sequenceNumberSet == orig(this._isSynSet)
this._sequenceNumberSet == orig(this._isFin)
this._sequenceNumberSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpDataBytes == orig(this._tcpDataBytes)
this._tcpDataBytes[] == orig(this._tcpDataBytes[])
this._rcsid == orig(this._rcsid)
this._sourcePortSet == true
this._destinationPort == 80
this._tcpHeaderLength one of 20, 32
this._tcpDataBytes != null
this._rcsid has only one value
this._rcsid != null
this._destinationPort > this._tcpHeaderLength
this._destinationPort < this._payloadDataLength
this._destinationPort in this._tcpDataBytes[]
this._destinationPort < return
this._sequenceNumber < this._tcpHeaderLength
this._sequenceNumber < this._payloadDataLength
this._sequenceNumber < return
this._tcpHeaderLength < this._payloadDataLength
this._tcpHeaderLength < return
this._payloadDataLength < return
this._tcpDataBytes[] elements < return
this._rcsid.toString == orig(this._rcsid.toString)
===
net.sourceforge.jpcap.net.TCPPacket.getTCPData():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber

this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPData():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst

this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == orig(this._tcpHeaderLength)
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._tcpDataBytes == return
this._tcpDataBytes[] == return[]
this._rcsid == orig(this._rcsid)
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
return != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
return[] elements > this._sourcePort
this._tcpHeaderLength < this._payloadDataLength
return[] elements < this._payloadDataLength
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPHeaderLength():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck

this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength
this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getTCPHeaderLength():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)

this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == return
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._rcsid == orig(this._rcsid)
return == orig(this._tcpHeaderLength)
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid has only one value
this._rcsid != null
return one of 20, 32
this._sourcePort < this._payloadDataLength
this._sourcePort < return
this._payloadDataLength > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.getTcpHeaderLength():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._tcpHeaderLength one of 20, 32
this._rcsid has only one value
this._rcsid != null
this._sourcePort < this._tcpHeaderLength
this._sourcePort < this._payloadDataLength

this._tcpHeaderLength < this._payloadDataLength
==========
net.sourceforge.jpcap.net.TCPPacket.getTcpHeaderLength():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)
this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderLength == return
this._payloadDataLength == orig(this._payloadDataLength)
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._rcsid == orig(this._rcsid)

return == orig(this._tcpHeaderLength)
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid has only one value
this._rcsid != null
return one of 20, 32
this._sourcePort < this._payloadDataLength
this._sourcePort < return
this._payloadDataLength > return
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.net.TCPPacket.this():::ENTER
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._tcpHeaderLength
this._sourcePort == this._payloadDataLength
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._tcpHeaderBytes == this._tcpDataBytes
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid == null
==========
net.sourceforge.jpcap.net.TCPPacket.this():::EXIT
this._sourcePort == this._destinationPort
this._sourcePort == this._sequenceNumber
this._sourcePort == this._acknowledgmentNumber
this._sourcePort == this._tcpHeaderLength
this._sourcePort == this._payloadDataLength
this._sourcePort == this._windowSize
this._sourcePort == this._tcpChecksum
this._sourcePort == this._urgentPointer
this._sourcePort == this._allFlags
this._sourcePort == orig(this._sourcePort)
this._sourcePort == orig(this._destinationPort)
this._sourcePort == orig(this._sequenceNumber)
this._sourcePort == orig(this._acknowledgmentNumber)
this._sourcePort == orig(this._tcpHeaderLength)
this._sourcePort == orig(this._payloadDataLength)
this._sourcePort == orig(this._windowSize)
this._sourcePort == orig(this._tcpChecksum)

this._sourcePort == orig(this._urgentPointer)
this._sourcePort == orig(this._allFlags)
this._sourcePortSet == this._destinationPortSet
this._sourcePortSet == this._sequenceNumberSet
this._sourcePortSet == this._acknowledgmentNumberSet
this._sourcePortSet == this._windowSizeSet
this._sourcePortSet == this._tcpChecksumSet
this._sourcePortSet == this._urgentPointerSet
this._sourcePortSet == this._allFlagsSet
this._sourcePortSet == this._isUrg
this._sourcePortSet == this._isUrgSet
this._sourcePortSet == this._isAck
this._sourcePortSet == this._isAckSet
this._sourcePortSet == this._isPsh
this._sourcePortSet == this._isPshSet
this._sourcePortSet == this._isRst
this._sourcePortSet == this._isRstSet
this._sourcePortSet == this._isSyn
this._sourcePortSet == this._isSynSet
this._sourcePortSet == this._isFin
this._sourcePortSet == this._isFinSet
this._sourcePortSet == orig(this._sourcePortSet)
this._sourcePortSet == orig(this._destinationPortSet)
this._sourcePortSet == orig(this._sequenceNumberSet)
this._sourcePortSet == orig(this._acknowledgmentNumberSet)
this._sourcePortSet == orig(this._windowSizeSet)
this._sourcePortSet == orig(this._tcpChecksumSet)
this._sourcePortSet == orig(this._urgentPointerSet)
this._sourcePortSet == orig(this._allFlagsSet)
this._sourcePortSet == orig(this._isUrg)
this._sourcePortSet == orig(this._isUrgSet)
this._sourcePortSet == orig(this._isAck)
this._sourcePortSet == orig(this._isAckSet)
this._sourcePortSet == orig(this._isPsh)
this._sourcePortSet == orig(this._isPshSet)
this._sourcePortSet == orig(this._isRst)
this._sourcePortSet == orig(this._isRstSet)
this._sourcePortSet == orig(this._isSyn)
this._sourcePortSet == orig(this._isSynSet)
this._sourcePortSet == orig(this._isFin)
this._sourcePortSet == orig(this._isFinSet)
this._tcpHeaderBytes == this._tcpDataBytes
this._tcpHeaderBytes == orig(this._tcpHeaderBytes)
this._tcpHeaderBytes == orig(this._tcpDataBytes)
this._sourcePort == 0
this._sourcePortSet == false
this._rcsid has only one value
this._rcsid != null
==========
net.sourceforge.jpcap.util.ArrayHelper:::CLASS
net.sourceforge.jpcap.util.ArrayHelper._rcsid has only one value
net.sourceforge.jpcap.util.ArrayHelper._rcsid != null
net.sourceforge.jpcap.util.ArrayHelper._rcsid.toString == "$Id: ArrayHelper.java,v 1.8 2002/09/04 19:36:47 pcharles Exp
$"
==========
net.sourceforge.jpcap.util.ArrayHelper.extractInteger(byte[], int, int):::ENTER
bytes != null
cnt one of 1, 2, 4
pos > cnt
==========
net.sourceforge.jpcap.util.ArrayHelper.extractInteger(byte[], int, int):::EXIT
net.sourceforge.jpcap.util.ArrayHelper._rcsid == orig(net.sourceforge.jpcap.util.ArrayHelper._rcsid)
bytes[] == orig(bytes[])

return != 0
net.sourceforge.jpcap.util.ArrayHelper._rcsid.toString == orig(net.sourceforge.jpcap.util.ArrayHelper._rcsid.toString)
return != orig(pos)
return != orig(cnt)
==========
net.sourceforge.jpcap.util.Timeval:::OBJECT
this.seconds >= 0
this.microseconds >= 0
this._rcsid.toString == "$Id: Timeval.java,v 1.2 2003/06/24 23:10:07 pcharles Exp $"
this.seconds >= this.microseconds
==========
net.sourceforge.jpcap.util.Timeval.Timeval(long, int):::ENTER
seconds > microseconds
==========
net.sourceforge.jpcap.util.Timeval.Timeval(long, int):::EXIT
this.seconds == orig(seconds)
this.microseconds == orig(microseconds)
this._rcsid has only one value
this._rcsid != null
this.seconds > this.microseconds
==========
net.sourceforge.jpcap.util.Timeval.getDate():::ENTER
this._rcsid has only one value
this._rcsid != null
this.seconds > this.microseconds
==========
net.sourceforge.jpcap.util.Timeval.getDate():::EXIT
this.seconds == orig(this.seconds)
this.microseconds == orig(this.microseconds)
this._rcsid == orig(this._rcsid)
this._rcsid has only one value
this._rcsid != null
return != null
this.seconds > this.microseconds
this._rcsid.toString == orig(this._rcsid.toString)
==========
net.sourceforge.jpcap.util.Timeval.this():::ENTER
this.seconds == this.microseconds
this.seconds == 0
this._rcsid == null
==========
net.sourceforge.jpcap.util.Timeval.this():::EXIT
this.seconds == this.microseconds
this.seconds == orig(this.seconds)
this.seconds == orig(this.microseconds)
this.seconds == 0
this._rcsid has only one value
this._rcsid != null
==========
pairs:::OBJECT
this.value >= 0.0
this.index >= 0
==========
pairs.pairs(float, int):::ENTER
arg0 >= 0.0
arg1 >= 0
==========
pairs.pairs(float, int):::EXIT
this.value == orig(arg0)
this.index == orig(arg1)

11.4 SVM

==========

weka.classifiers.functions.SMO$BinarySMO:::CLASS
==========
weka.classifiers.functions.SMO$BinarySMO:::OBJECT
this.m_alpha[] elements <= 1.0
this.m_alpha[] elements >= 0.0
this.m_iLow >= -1
this.m_iUp >= -1
this.m_sparseIndices[] elements >= 0
this.m_sparseIndices[] sorted by <
this.m_kernel != null
this.m_kernel.getClass() == weka.classifiers.functions.supportVector.PolyKernel.class
this.m_class[] elements one of -1.0, 1.0
this.m_class[] elements != 0
this.m_logistic == null
weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < this.m_iLow
weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < this.m_iUp
this.m_sparseIndices[] elements > weka.classifiers.functions.SMO$BinarySMO.serialVersionUID
this.m_alpha[] elements >= this.m_b
this.m_alpha[]
this.m_alpha[] == this.m_sparseWeights[]**2 (elementwise)
this.m_alpha[] is the reverse of this.m_sparseWeights[]
this.m_sparseWeights[]
this.m_sparseWeights[] == this.m_alpha[]**2 (elementwise)
this.m_alpha[] elements < this.m_sumOfWeights
this.m_b in this.m_weights[]
this.m_b <= this.m_sumOfWeights
this.m_bLow in this.m_errors[]
this.m_bLow <= this.m_sumOfWeights
this.m_bUp in this.m_errors[]
this.m_bUp <= this.m_sumOfWeights
this.m_iLow < weka.classifiers.functions.Logistic.serialVersionUID
this.m_iUp < weka.classifiers.functions.Logistic.serialVersionUID
this.m_sparseWeights[]
this.m_sparseWeights[] == this.m_weights[]**2 (elementwise)
this.m_weights[]
this.m_weights[] == this.m_sparseWeights[]**2 (elementwise)
this.m_weights[] is the reverse of this.m_sparseWeights[]
this.m_weights[] elements < this.m_sumOfWeights
this.m_class[] == this.m_sparseWeights[]**2 (elementwise)
this.m_sparseWeights[] == this.m_class[]**2 (elementwise)
this.m_sparseWeights[] is the reverse of this.m_class[]
this.m_errors[]
this.m_errors[] == this.m_sparseWeights[]**2 (elementwise)
this.m_sparseWeights[]
this.m_sparseWeights[] == this.m_errors[]**2 (elementwise)
this.m_sparseWeights[] is the reverse of this.m_errors[]
this.m_sparseWeights[] elements < this.m_sumOfWeights
this.m_sparseIndices[] elements < weka.classifiers.functions.Logistic.serialVersionUID
this.m_class[] elements < this.m_sumOfWeights
this.m_errors[] elements < this.m_sumOfWeights
==========
weka.classifiers.functions.SMO$BinarySMO.SVMOutput(int, weka.core.Instance):::ENTER
this.m_bLow != 0
this.m_bUp != 0
arg0 >= -1
arg1 != null
weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < arg0
this.m_b < this.m_sumOfWeights
this.m_bLow < this.m_sumOfWeights
this.m_bUp < this.m_sumOfWeights
this.m_sparseIndices[] elements > arg0
weka.classifiers.functions.Logistic.serialVersionUID > arg0
==========

weka.classifiers.functions.SMO$BinarySMO.SVMOutput(int, weka.core.Instance):::EXIT
this.m_alpha == orig(this.m_alpha)
this.m_alpha[] == orig(this.m_alpha[])
this.m_b == orig(this.m_b)
this.m_bLow == orig(this.m_bLow)
this.m_bUp == orig(this.m_bUp)
this.m_iLow == orig(this.m_iLow)
this.m_iUp == orig(this.m_iUp)
this.m_data == orig(this.m_data)
this.m_weights == orig(this.m_weights)
this.m_weights[] == orig(this.m_weights[])
this.m_sparseWeights == orig(this.m_sparseWeights)
this.m_sparseWeights[] == orig(this.m_sparseWeights[])
this.m_sparseIndices == orig(this.m_sparseIndices)
this.m_sparseIndices[] == orig(this.m_sparseIndices[])
this.m_kernel == orig(this.m_kernel)
this.m_class == orig(this.m_class)
this.m_class[] == orig(this.m_class[])
this.m_errors == orig(this.m_errors)
this.m_errors[] == orig(this.m_errors[])
this.m_I0 == orig(this.m_I0)
this.m_I1 == orig(this.m_I1)
this.m_I2 == orig(this.m_I2)
this.m_I3 == orig(this.m_I3)
this.m_I4 == orig(this.m_I4)
this.m_supportVectors == orig(this.m_supportVectors)
this.m_logistic == orig(this.m_logistic)
this.m_sumOfWeights == orig(this.m_sumOfWeights)
this.m_bLow != 0
this.m_bUp != 0
this.m_b < this.m_sumOfWeights
this.m_bLow < this.m_sumOfWeights
this.m_bUp < this.m_sumOfWeights
this.m_sparseIndices[] elements > orig(arg0)
this.m_kernel.getClass() == orig(this.m_kernel.getClass())
this.m_sumOfWeights > return
==========
weka.classifiers.functions.SMO$BinarySMO.buildClassifier(weka.core.Instances, int, int, boolean, int, int):::ENTER
this.m_alpha == this.m_weights
this.m_alpha == this.m_sparseWeights
this.m_alpha == this.m_class
this.m_alpha == this.m_errors
this.m_b == this.m_bLow
this.m_b == this.m_bUp
this.m_b == this.m_sumOfWeights
this.m_iLow == this.m_iUp
this.m_I0 == this.m_I1
this.m_I0 == this.m_I2
this.m_I0 == this.m_I3
this.m_I0 == this.m_I4
this.m_I0 == this.m_supportVectors
this.m_alpha == null
this.m_b == 0.0
this.m_iLow == 0
this.m_data == null
this.m_sparseIndices == null
this.m_I0 == null
arg0 != null
arg1 >= 0
arg2 >= 1
arg3 == false
arg4 == -1
arg5 == 1

weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < arg1
weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < arg2
this.m_iLow <= arg1
this.m_iLow < arg2
weka.classifiers.functions.Logistic.serialVersionUID > arg1
weka.classifiers.functions.Logistic.serialVersionUID > arg2
arg1 < arg2
arg1 > arg4
arg2 > arg4
arg2 >= arg5
---
weka.classifiers.functions.SMO$BinarySMO.buildClassifier(weka.core.Instances, int, int, boolean, int, int):::EXIT2
this.m_data == orig(this.m_data)
this.m_b one of -1.0, 1.0
this.m_bLow == 1.0
this.m_bUp == -1.0
this.m_data == null
this.m_sparseWeights[] == []
this.m_sparseIndices[] == []
this.m_sumOfWeights
this.m_iLow != this.m_iUp
this.m_iLow != orig(this.m_iLow)
this.m_iLow != orig(arg1)
this.m_iLow != orig(arg2)
this.m_iLow != orig(arg5)
this.m_iUp != orig(this.m_iLow)
this.m_iUp != orig(arg1)
this.m_iUp != orig(arg2)
this.m_iUp != orig(arg5)
orig(arg2) > orig(arg5)
---
weka.classifiers.functions.SMO$BinarySMO.buildClassifier(weka.core.Instances, int, int, boolean, int, int):::EXIT3
this.m_iLow >= 0
this.m_iUp >= 0
this.m_data != null
this.m_iLow >= orig(this.m_iLow)
this.m_iLow > orig(arg4)
this.m_iUp >= orig(this.m_iLow)
this.m_iUp > orig(arg4)
orig(this.m_iLow) in this.m_sparseIndices[]
orig(arg1) in this.m_sparseIndices[]
orig(arg2) in this.m_sparseIndices[]
orig(arg5) in this.m_sparseIndices[]
---
weka.classifiers.functions.SMO$BinarySMO.buildClassifier(weka.core.Instances, int, int, boolean, int, int):::EXIT
this.m_alpha == this.m_weights
this.m_alpha == this.m_class
this.m_alpha == this.m_errors
this.m_alpha == orig(this.m_alpha)
this.m_alpha == orig(this.m_weights)
this.m_alpha == orig(this.m_sparseWeights)
this.m_alpha == orig(this.m_class)
this.m_alpha == orig(this.m_errors)
this.m_kernel == orig(this.m_kernel)
this.m_I0 == this.m_I1
this.m_I0 == this.m_I2
this.m_I0 == this.m_I3
this.m_I0 == this.m_I4
this.m_I0 == this.m_supportVectors
this.m_I0 == orig(this.m_I0)
this.m_I0 == orig(this.m_I1)
this.m_I0 == orig(this.m_I2)
this.m_I0 == orig(this.m_I3)

this.m_I0 == orig(this.m_I4)
this.m_I0 == orig(this.m_supportVectors)
this.m_logistic == orig(this.m_logistic)
this.m_alpha == null
this.m_b != 0
this.m_bLow != 0
this.m_bUp != 0
this.m_sparseWeights != null
this.m_sparseIndices != null
this.m_I0 == null
this.m_b <= this.m_bLow
this.m_b >= this.m_bUp
this.m_b < this.m_sumOfWeights
this.m_b != orig(this.m_b)
this.m_bLow >= this.m_bUp
this.m_bLow < this.m_sumOfWeights
this.m_bLow != orig(this.m_b)
this.m_bUp < this.m_sumOfWeights
this.m_bUp != orig(this.m_b)
this.m_iLow >= orig(arg4)
this.m_iUp >= orig(arg4)
this.m_sparseIndices[] elements >= orig(this.m_iLow)
this.m_sparseIndices[] elements > orig(arg4)
this.m_kernel.getClass() == orig(this.m_kernel.getClass())
this.m_sumOfWeights > orig(this.m_b)
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::ENTER
this.m_alpha != null
this.m_b == 0.0
this.m_bLow != 0
this.m_bUp != 0
this.m_iLow >= 0
this.m_iUp >= 0
this.m_data != null
this.m_weights != null
this.m_sparseWeights == null
this.m_sparseIndices == null
this.m_class != null
this.m_errors != null
this.m_I0 != null
this.m_I1 != null
this.m_I2 != null
this.m_I3 != null
this.m_I4 != null
this.m_supportVectors != null
arg0 >= 0
weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < arg0
this.m_b != this.m_bLow
this.m_b != this.m_bUp
this.m_b < this.m_sumOfWeights
this.m_bLow < this.m_sumOfWeights
this.m_bUp < this.m_sumOfWeights
weka.classifiers.functions.Logistic.serialVersionUID > arg0
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT1
this.m_alpha[] == orig(this.m_alpha[])
this.m_weights[] == orig(this.m_weights[])
return == false
this.m_bLow >= orig(this.m_bLow)
this.m_bUp <= orig(this.m_bUp)
orig(this.m_bLow) in this.m_errors[]
orig(this.m_bUp) in this.m_errors[]
==========

weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT1;condition="not(return == true)"
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT2
(return == false) ==> (orig(this.m_bLow) >= orig(this.m_bUp))
(return == false) ==> (orig(this.m_bLow) in this.m_errors[])
(return == false) ==> (orig(this.m_bUp) in this.m_errors[])
(return == false) ==> (orig(this.m_errors[]) elements != 0)
(return == false) ==> (this.m_alpha[] == orig(this.m_alpha[]))
(return == false) ==> (this.m_b in orig(this.m_alpha[]))
(return == false) ==> (this.m_b in this.m_alpha[])
(return == false) ==> (this.m_bLow > this.m_bUp)
(return == false) ==> (this.m_bLow >= orig(this.m_bLow))
(return == false) ==> (this.m_bLow >= orig(this.m_bUp))
(return == false) ==> (this.m_bUp <= orig(this.m_bLow))
(return == false) ==> (this.m_bUp <= orig(this.m_bUp))
(return == false) ==> (this.m_errors[] elements != 0)
(return == false) ==> (this.m_iLow != this.m_iUp)
(return == false) ==> (this.m_weights[] == orig(this.m_weights[]))
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT2;condition="return == true"
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT2;condition="not(return == true)"
this.m_errors[] elements != 0
orig(this.m_errors[]) elements != 0
this.m_b in this.m_alpha[]
this.m_bLow > this.m_bUp
this.m_bLow >= orig(this.m_bUp)
this.m_bUp <= orig(this.m_bLow)
this.m_iLow != this.m_iUp
orig(this.m_bLow) >= orig(this.m_bUp)
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT
this.m_alpha == orig(this.m_alpha)
this.m_b == orig(this.m_b)
this.m_data == orig(this.m_data)
this.m_weights == orig(this.m_weights)
this.m_sparseWeights == orig(this.m_sparseWeights)
this.m_sparseIndices == orig(this.m_sparseIndices)
this.m_kernel == orig(this.m_kernel)
this.m_class == orig(this.m_class)
this.m_class[] == orig(this.m_class[])
this.m_errors == orig(this.m_errors)
this.m_I0 == orig(this.m_I0)
this.m_I1 == orig(this.m_I1)
this.m_I2 == orig(this.m_I2)
this.m_I3 == orig(this.m_I3)
this.m_I4 == orig(this.m_I4)
this.m_supportVectors == orig(this.m_supportVectors)
this.m_logistic == orig(this.m_logistic)
this.m_sumOfWeights == orig(this.m_sumOfWeights)
(return == false) ==> (orig(this.m_bLow) in this.m_errors[])
(return == false) ==> (orig(this.m_bUp) in this.m_errors[])
(return == false) ==> (this.m_alpha[] == orig(this.m_alpha[]))
(return == false) ==> (this.m_bLow >= orig(this.m_bLow))
(return == false) ==> (this.m_bUp <= orig(this.m_bUp))
(return == false) ==> (this.m_weights[] == orig(this.m_weights[]))
this.m_alpha != null
this.m_b == 0.0
this.m_bLow != 0
this.m_bUp != 0
this.m_iLow >= 0
this.m_iUp >= 0
this.m_data != null

this.m_weights != null
this.m_sparseWeights == null
this.m_sparseIndices == null
this.m_class != null
this.m_errors != null
this.m_I0 != null
this.m_I1 != null
this.m_I2 != null
this.m_I3 != null
this.m_I4 != null
this.m_supportVectors != null
this.m_b != this.m_bLow
this.m_b != this.m_bUp
this.m_b < this.m_sumOfWeights
orig(this.m_alpha[]) elements >= this.m_b
this.m_b != orig(this.m_bLow)
this.m_b != orig(this.m_bUp)
this.m_b in orig(this.m_weights[])
this.m_bLow < this.m_sumOfWeights
this.m_bUp < this.m_sumOfWeights
this.m_kernel.getClass() == orig(this.m_kernel.getClass())
weka.classifiers.functions.Logistic.serialVersionUID > orig(this.m_iLow)
weka.classifiers.functions.Logistic.serialVersionUID > orig(this.m_iUp)
orig(this.m_alpha[]) elements < this.m_sumOfWeights
this.m_sumOfWeights > orig(this.m_bLow)
this.m_sumOfWeights > orig(this.m_bUp)
orig(this.m_weights[]) elements < this.m_sumOfWeights
orig(this.m_errors[]) elements < this.m_sumOfWeights
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT;condition="return == true"
return == true
==========
weka.classifiers.functions.SMO$BinarySMO.examineExample(int):::EXIT;condition="not(return == true)"
this.m_alpha[] == orig(this.m_alpha[])
this.m_weights[] == orig(this.m_weights[])
return == false
this.m_bLow >= orig(this.m_bLow)
this.m_bUp <= orig(this.m_bUp)
orig(this.m_bLow) in this.m_errors[]
orig(this.m_bUp) in this.m_errors[]
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::ENTER
this.m_alpha != null
this.m_b == 0.0
this.m_bLow != 0
this.m_bUp != 0
this.m_iLow >= 0
this.m_iUp >= 0
this.m_data != null
this.m_weights != null
this.m_sparseWeights == null
this.m_sparseIndices == null
this.m_class != null
this.m_errors != null
this.m_I0 != null
this.m_I1 != null
this.m_I2 != null
this.m_I3 != null
this.m_I4 != null
this.m_supportVectors != null
arg0 >= 0
arg1 >= 0
arg2 != 0

weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < arg0
weka.classifiers.functions.SMO$BinarySMO.serialVersionUID < arg1
this.m_b != this.m_bLow
this.m_b != this.m_bUp
this.m_b < this.m_sumOfWeights
this.m_b != arg2
this.m_bLow > this.m_bUp
this.m_bLow < this.m_sumOfWeights
this.m_bLow >= arg2
this.m_bUp < this.m_sumOfWeights
this.m_bUp <= arg2
this.m_iLow != this.m_iUp
arg2 in this.m_errors[]
weka.classifiers.functions.Logistic.serialVersionUID > arg0
weka.classifiers.functions.Logistic.serialVersionUID > arg1
this.m_sumOfWeights > arg2
arg0 != arg1
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT2
this.m_alpha[] == orig(this.m_alpha[])
this.m_bLow == orig(this.m_bLow)
this.m_bUp == orig(this.m_bUp)
this.m_iLow == orig(this.m_iLow)
this.m_iUp == orig(this.m_iUp)
this.m_weights[] == orig(this.m_weights[])
this.m_errors[] == orig(this.m_errors[])
this.m_errors[] elements != 0
return == false
this.m_b in this.m_alpha[]
this.m_bLow > this.m_bUp
this.m_bLow >= orig(arg2)
this.m_bUp <= orig(arg2)
this.m_iLow != this.m_iUp
orig(arg2) in this.m_errors[]
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT2;condition="not(return == true)"
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT3
this.m_alpha[] == orig(this.m_alpha[])
this.m_bLow == orig(this.m_bLow)
this.m_bUp == orig(this.m_bUp)
this.m_iLow == orig(this.m_iLow)
this.m_iUp == orig(this.m_iUp)
this.m_weights[] == orig(this.m_weights[])
this.m_errors[] == orig(this.m_errors[])
this.m_errors[] elements != 0
return == false
this.m_b in this.m_alpha[]
this.m_bLow > this.m_bUp
this.m_bLow >= orig(arg2)
this.m_bUp <= orig(arg2)
this.m_iLow != this.m_iUp
orig(arg2) in this.m_errors[]
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT3;condition="not(return == true)"
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT4
return == true
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT4;condition="return == true"
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT
this.m_alpha == orig(this.m_alpha)

this.m_b == orig(this.m_b)
this.m_data == orig(this.m_data)
this.m_weights == orig(this.m_weights)
this.m_sparseWeights == orig(this.m_sparseWeights)
this.m_sparseIndices == orig(this.m_sparseIndices)
this.m_kernel == orig(this.m_kernel)
this.m_class == orig(this.m_class)
this.m_class[] == orig(this.m_class[])
this.m_errors == orig(this.m_errors)
this.m_I0 == orig(this.m_I0)
this.m_I1 == orig(this.m_I1)
this.m_I2 == orig(this.m_I2)
this.m_I3 == orig(this.m_I3)
this.m_I4 == orig(this.m_I4)
this.m_supportVectors == orig(this.m_supportVectors)
this.m_logistic == orig(this.m_logistic)
this.m_sumOfWeights == orig(this.m_sumOfWeights)
(return == false) ==> (orig(arg2) in this.m_errors[])
(return == false) ==> (orig(this.m_bLow) in this.m_errors[])
(return == false) ==> (orig(this.m_bUp) in this.m_errors[])
(return == false) ==> (orig(this.m_errors[]) elements != 0)
(return == false) ==> (this.m_alpha[] == orig(this.m_alpha[]))
(return == false) ==> (this.m_b in orig(this.m_alpha[]))
(return == false) ==> (this.m_b in this.m_alpha[])
(return == false) ==> (this.m_bLow == orig(this.m_bLow))
(return == false) ==> (this.m_bLow > orig(this.m_bUp))
(return == false) ==> (this.m_bLow > this.m_bUp)
(return == false) ==> (this.m_bLow >= orig(arg2))
(return == false) ==> (this.m_bLow in orig(this.m_errors[]))
(return == false) ==> (this.m_bUp < orig(this.m_bLow))
(return == false) ==> (this.m_bUp <= orig(arg2))
(return == false) ==> (this.m_bUp == orig(this.m_bUp))
(return == false) ==> (this.m_bUp in orig(this.m_errors[]))
(return == false) ==> (this.m_errors[] == orig(this.m_errors[]))
(return == false) ==> (this.m_errors[] elements != 0)
(return == false) ==> (this.m_iLow != orig(this.m_iUp))
(return == false) ==> (this.m_iLow != this.m_iUp)
(return == false) ==> (this.m_iLow == orig(this.m_iLow))
(return == false) ==> (this.m_iUp != orig(this.m_iLow))
(return == false) ==> (this.m_iUp == orig(this.m_iUp))
(return == false) ==> (this.m_weights[] == orig(this.m_weights[]))
this.m_alpha != null
this.m_b == 0.0
this.m_bLow != 0
this.m_bUp != 0
this.m_iLow >= 0
this.m_iUp >= 0
this.m_data != null
this.m_weights != null
this.m_sparseWeights == null
this.m_sparseIndices == null
this.m_class != null
this.m_errors != null
this.m_I0 != null
this.m_I1 != null
this.m_I2 != null
this.m_I3 != null
this.m_I4 != null
this.m_supportVectors != null
this.m_b != this.m_bLow
this.m_b != this.m_bUp
this.m_b < this.m_sumOfWeights
orig(this.m_alpha[]) elements >= this.m_b

this.m_b != orig(this.m_bLow)
this.m_b != orig(this.m_bUp)
this.m_b in orig(this.m_weights[])
this.m_b != orig(arg2)
this.m_bLow < this.m_sumOfWeights
this.m_bUp < this.m_sumOfWeights
this.m_kernel.getClass() == orig(this.m_kernel.getClass())
weka.classifiers.functions.Logistic.serialVersionUID > orig(this.m_iLow)
weka.classifiers.functions.Logistic.serialVersionUID > orig(this.m_iUp)
orig(this.m_alpha[]) elements < this.m_sumOfWeights
this.m_sumOfWeights > orig(this.m_bLow)
this.m_sumOfWeights > orig(this.m_bUp)
orig(this.m_weights[]) elements < this.m_sumOfWeights
orig(this.m_errors[]) elements < this.m_sumOfWeights
this.m_sumOfWeights > orig(arg2)
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT;condition="return == true"
return == true
==========
weka.classifiers.functions.SMO$BinarySMO.takeStep(int, int, double):::EXIT;condition="not(return == true)"
this.m_alpha[] == orig(this.m_alpha[])
this.m_bLow == orig(this.m_bLow)
this.m_bUp == orig(this.m_bUp)
this.m_iLow == orig(this.m_iLow)
this.m_iUp == orig(this.m_iUp)
this.m_weights[] == orig(this.m_weights[])
this.m_errors[] == orig(this.m_errors[])
this.m_errors[] elements != 0
return == false
this.m_b in this.m_alpha[]
this.m_bLow > this.m_bUp
this.m_bLow >= orig(arg2)
this.m_bUp <= orig(arg2)
this.m_iLow != this.m_iUp
orig(arg2) in this.m_errors[]
==========
weka.classifiers.functions.SMO:::CLASS
weka.classifiers.functions.SMO.TAGS_FILTER != null
weka.classifiers.functions.SMO.TAGS_FILTER.getClass() == weka.core.Tag[].class
weka.classifiers.functions.SMO.TAGS_FILTER[] elements != null
weka.classifiers.functions.SMO.TAGS_FILTER[].getClass() == [weka.core.Tag, weka.core.Tag, weka.core.Tag]
weka.classifiers.functions.SMO.TAGS_FILTER[].getClass() elements == weka.core.Tag.class
weka.classifiers.functions.SMO.m_Del == 4.94E-321
==========
weka.classifiers.functions.SMO:::OBJECT
weka.classifiers.functions.SMO.FILTER_NORMALIZE == this.m_filterType
weka.classifiers.functions.SMO.FILTER_STANDARDIZE == this.m_randomSeed
this.m_checksTurnedOff == this.m_fitLogisticModels
this.m_classifiers.getClass() == weka.classifiers.functions.SMO$BinarySMO[][].class
this.m_classifiers[] elements != null
this.m_C == 1.0
this.m_eps == 1.0E-12
this.m_tol == 0.0010
this.m_Filter.getClass() == weka.filters.unsupervised.attribute.Normalize.class
this.m_classIndex != 0
this.m_checksTurnedOff == false
this.m_numFolds == -1
this.m_kernel != null
this.m_kernel.getClass() == weka.classifiers.functions.supportVector.PolyKernel.class
weka.classifiers.functions.SMO.serialVersionUID < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_NORMALIZE != this.m_classIndex
weka.classifiers.functions.SMO.FILTER_STANDARDIZE != this.m_classIndex
weka.classifiers.functions.SMO.FILTER_NONE != this.m_classIndex

weka.classifiers.functions.SMO.TAGS_FILTER.getClass() != this.m_classifiers.getClass()
weka.classifiers.functions.SMO.TAGS_FILTER.getClass() != this.m_Filter.getClass()
weka.classifiers.functions.SMO.TAGS_FILTER.getClass() != this.m_kernel.getClass()
this.m_classifiers.getClass() != this.m_Filter.getClass()
this.m_classifiers.getClass() != this.m_kernel.getClass()
this.m_Filter.getClass() != this.m_kernel.getClass()
this.m_classIndex >= this.m_numFolds
==========
weka.classifiers.functions.SMO.buildClassifier(weka.core.Instances):::ENTER
this.m_classIndex == this.m_numFolds
this.m_KernelIsLinear == this.m_checksTurnedOff
this.m_KernelIsLinear == this.m_fitLogisticModels
this.m_classifiers == null
this.m_NominalToBinary == null
this.m_Filter == null
this.m_Missing == null
this.m_classIndex == -1
this.m_classAttribute == null
this.m_KernelIsLinear == false
arg0 != null
==========
weka.classifiers.functions.SMO.buildClassifier(weka.core.Instances):::EXIT
weka.classifiers.functions.SMO.TAGS_FILTER == orig(weka.classifiers.functions.SMO.TAGS_FILTER)
weka.classifiers.functions.SMO.TAGS_FILTER[] == orig(weka.classifiers.functions.SMO.TAGS_FILTER[])
this.m_C == orig(this.m_C)
this.m_eps == orig(this.m_eps)
this.m_tol == orig(this.m_tol)
this.m_checksTurnedOff == orig(this.m_KernelIsLinear)
this.m_checksTurnedOff == orig(this.m_checksTurnedOff)
this.m_checksTurnedOff == orig(this.m_fitLogisticModels)
weka.classifiers.functions.SMO.m_Del == orig(weka.classifiers.functions.SMO.m_Del)
this.m_numFolds == orig(this.m_classIndex)
this.m_numFolds == orig(this.m_numFolds)
this.m_kernel == orig(this.m_kernel)
this.m_classifiers != null
this.m_Filter != null
this.m_Missing != null
this.m_classAttribute != null
this.m_KernelIsLinear == true
weka.classifiers.functions.SMO.FILTER_NORMALIZE < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_STANDARDIZE < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_NONE < this.m_classIndex
weka.classifiers.functions.SMO.TAGS_FILTER.getClass() == orig(weka.classifiers.functions.SMO.TAGS_FILTER.getClass())
this.m_classIndex > this.m_numFolds
this.m_kernel.getClass() == orig(this.m_kernel.getClass())
==========
weka.classifiers.functions.SMO.distributionForInstance(weka.core.Instance):::ENTER
this.m_classifiers != null
this.m_Filter != null
this.m_Missing != null
this.m_classAttribute != null
this.m_KernelIsLinear == true
arg0 != null
weka.classifiers.functions.SMO.FILTER_NORMALIZE < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_STANDARDIZE < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_NONE < this.m_classIndex
this.m_classIndex > this.m_numFolds
==========
weka.classifiers.functions.SMO.distributionForInstance(weka.core.Instance):::EXIT1
==========
weka.classifiers.functions.SMO.distributionForInstance(weka.core.Instance):::EXIT
weka.classifiers.functions.SMO.TAGS_FILTER == orig(weka.classifiers.functions.SMO.TAGS_FILTER)
weka.classifiers.functions.SMO.TAGS_FILTER[] == orig(weka.classifiers.functions.SMO.TAGS_FILTER[])

this.m_classifiers == orig(this.m_classifiers)
this.m_classifiers[] == orig(this.m_classifiers[])
this.m_C == orig(this.m_C)
this.m_eps == orig(this.m_eps)
this.m_tol == orig(this.m_tol)
this.m_NominalToBinary == orig(this.m_NominalToBinary)
this.m_Filter == orig(this.m_Filter)
this.m_Missing == orig(this.m_Missing)
this.m_classIndex == orig(this.m_classIndex)
this.m_classAttribute == orig(this.m_classAttribute)
this.m_KernelIsLinear == orig(this.m_KernelIsLinear)
this.m_checksTurnedOff == orig(this.m_checksTurnedOff)
this.m_checksTurnedOff == orig(this.m_fitLogisticModels)
weka.classifiers.functions.SMO.m_Del == orig(weka.classifiers.functions.SMO.m_Del)
this.m_numFolds == orig(this.m_numFolds)
this.m_kernel == orig(this.m_kernel)
this.m_classifiers != null
this.m_Filter != null
this.m_Missing != null
this.m_classAttribute != null
this.m_KernelIsLinear == true
return != null
return[] elements >= 0.0
weka.classifiers.functions.SMO.FILTER_NORMALIZE < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_STANDARDIZE < this.m_classIndex
weka.classifiers.functions.SMO.FILTER_NONE < this.m_classIndex
weka.classifiers.functions.SMO.TAGS_FILTER.getClass() == orig(weka.classifiers.functions.SMO.TAGS_FILTER.getClass())
this.m_classifiers.getClass() == orig(this.m_classifiers.getClass())
return[] elements <= this.m_C
return[] elements >= this.m_eps
this.m_Filter.getClass() == orig(this.m_Filter.getClass())
this.m_classIndex > this.m_numFolds
return[] elements >= weka.classifiers.functions.SMO.m_Del
this.m_kernel.getClass() == orig(this.m_kernel.getClass())

11.5 gaffitter

11.6 Lucene

```
==========================================================================
org.apache.lucene.search.DefaultSimilarity:::OBJECT
==========================================================================
org.apache.lucene.search.DefaultSimilarity.DefaultSimilarity():::EXIT
==========================================================================
org.apache.lucene.search.DefaultSimilarity.coord(int, int):::ENTER
overlap one of  0, 1, 2
maxOverlap one of  1, 2
overlap <= maxOverlap
==========================================================================
org.apache.lucene.search.DefaultSimilarity.coord(int, int):::EXIT
return one of  0.0, 0.5, 1.0
==========================================================================
org.apache.lucene.search.DefaultSimilarity.idf(int, int):::ENTER
docFreq one of  0, 18, 19
numDocs == 24
docFreq < numDocs
==========================================================================
org.apache.lucene.search.DefaultSimilarity.idf(int, int):::EXIT
return one of  1.1823215, 1.2336148, 4.178054
==========================================================================
org.apache.lucene.search.DefaultSimilarity.queryNorm(float):::ENTER
sumOfSquaredWeights >= 1.3978842
==========================================================================
org.apache.lucene.search.DefaultSimilarity.queryNorm(float):::EXIT
```

return < orig(sumOfSquaredWeights)
==========
org.apache.lucene.search.DefaultSimilarity.tf(float):::ENTER
freq >= 0.0
==========
org.apache.lucene.search.DefaultSimilarity.tf(float):::EXIT
orig(freq) == return**2
return <= orig(freq)
==========
org.apache.lucene.search.TermScorer:::CLASS
==========
org.apache.lucene.search.TermScorer:::OBJECT
this.docs[] <= this.freqs[] (lexically)
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE == size(this.docs[])
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE == size(this.freqs[])
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE == size(this.scoreCache[])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CAC
1]
this.weight != null
this.weight.getClass() == org.apache.lucene.search.TermQuery$TermWeight.class
this.termDocs != null
this.termDocs.getClass() == org.apache.lucene.index.SegmentTermDocs.class
this.norms != null
this.norms[] == [105, 100, 102, 103, 100, 102, 103, 102, 104, 103, 101, 104, 102, 106, 100, 104, 105, 106, 101, 106, 101, 106,
105, 101]
this.doc >= 0
this.docs != null
this.docs[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 2, 3, 4, 5, 6, 7, 8,
10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21,
22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.docs[] elements >= 0
this.freqs != null
this.freqs[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [7, 20, 25, 15, 25, 4,
13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6,
31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] elements >= 0
this.pointer >= 0
this.pointerMax one of 0, 18, 19
this.scoreCache != null
this.scoreCache[] elements >= 0.0
this.scoreCache[] sorted by <
size(this.norms[]) == 24
this.norms[this.pointerMax] one of 101, 105, 106
this.docs[this.pointerMax] one of 0, 2
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] == 0
this.freqs[this.pointerMax] one of 0, 7, 14
this.weight.getClass() != this.termDocs.getClass()
this.norms[] elements > this.pointer
this.norms[] elements > this.pointerMax
this.norms[] elements > org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE
this.norms[] elements > size(this.norms[])
this.norms[] elements > this.docs[this.pointer]
this.norms[] elements > this.docs[this.pointerMax]
this.norms[] elements > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.norms[] elements > this.freqs[this.pointer]
this.norms[] elements > this.freqs[this.pointerMax]
this.weightValue in this.scoreCache[]
this.doc >= this.pointer
this.doc != org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE
this.doc != size(this.norms[])
this.doc != size(this.docs[])-1
this.doc != this.norms[this.pointer]
this.doc != this.norms[this.pointerMax]

this.doc >= this.docs[this.pointer]
this.doc >= this.docs[this.pointerMax]
this.doc >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.doc >= this.freqs[this.pointerMax]
this.pointerMax in this.docs[]
this.docs[] elements <= size(this.norms[])-1
this.docs[] elements < size(this.docs[])-1
this.docs[] elements < this.norms[this.pointer]
this.docs[] elements < this.norms[this.pointerMax]
this.docs[] elements >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.freqs[this.pointerMax] in this.docs[]
this.freqs[] elements < this.norms[this.pointer]
this.freqs[] elements < this.norms[this.pointerMax]
this.freqs[] elements >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointer < size(this.norms[])-1
this.pointer < size(this.docs[])-1
this.pointer < this.norms[this.pointer]
this.pointer < this.norms[this.pointerMax]
this.pointer >= this.docs[this.pointerMax]
this.pointer >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointer >= this.freqs[this.pointerMax]
this.pointerMax < size(this.norms[])-1
this.pointerMax < size(this.docs[])-1
this.pointerMax < this.norms[this.pointer]
this.pointerMax < this.norms[this.pointerMax]
this.pointerMax >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointer]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointer]
size(this.norms[]) < this.norms[this.pointer]
size(this.norms[]) < this.norms[this.pointerMax]
size(this.norms[]) != this.freqs[this.pointer]
size(this.norms[])-1 >= this.docs[this.pointer]
size(this.norms[])-1 > this.docs[this.pointerMax]
size(this.norms[])-1 != this.freqs[this.pointer]
size(this.norms[])-1 > this.freqs[this.pointerMax]
size(this.docs[])-1 > this.docs[this.pointer]
size(this.docs[])-1 > this.docs[this.pointerMax]
size(this.docs[])-1 > this.freqs[this.pointerMax]
this.norms[this.pointer] > this.docs[this.pointer]
this.norms[this.pointer] > this.docs[this.pointerMax]
this.norms[this.pointer] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.norms[this.pointer] > this.freqs[this.pointer]
this.norms[this.pointer] > this.freqs[this.pointerMax]
this.norms[this.pointerMax] > this.docs[this.pointer]
this.norms[this.pointerMax] > this.docs[this.pointerMax]
this.norms[this.pointerMax] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.norms[this.pointerMax] > this.freqs[this.pointer]
this.norms[this.pointerMax] > this.freqs[this.pointerMax]
this.docs[this.pointer] >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] <= this.freqs[this.pointerMax]
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.freqs[this.pointer]
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.freqs[this.pointerMax]
==========
org.apache.lucene.search.TermScorer.TermScorer(org.apache.lucene.search.Weight, org.apache.lucene.index.TermDocs, org.apache.lucene.s
byte[]):::ENTER
weight != null
weight.getClass() == org.apache.lucene.search.TermQuery$TermWeight.class
td != null
td.getClass() == org.apache.lucene.index.SegmentTermDocs.class
similarity != null
similarity.getClass() == org.apache.lucene.search.DefaultSimilarity.class

norms != null
norms[] == [105, 100, 102, 103, 100, 102, 103, 102, 104, 103, 101, 104, 102, 106, 100, 104, 105, 106, 101, 106, 101, 106, 105, 101]
size(norms[]) == 24
weight.getClass() != td.getClass()
weight.getClass() != similarity.getClass()
td.getClass() != similarity.getClass()
norms[] elements > size(norms[])
==========
org.apache.lucene.search.TermScorer.TermScorer(org.apache.lucene.search.Weight, org.apache.lucene.index.TermDocs, org.apache.lucene.s byte[]):::EXIT
this.weight == orig(weight)
this.termDocs == orig(td)
this.norms == orig(norms)
this.norms[] == norms[]
this.doc == this.pointer
this.doc == this.pointerMax
this.doc == this.docs[this.doc]
this.doc == this.freqs[this.doc]
this.doc == this.docs[this.pointer]
this.doc == this.docs[this.pointerMax]
this.doc == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.doc == this.freqs[this.pointer]
this.doc == this.freqs[this.pointerMax]
this.doc == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[] == this.freqs[]
norms[] == orig(norms[])
this.doc == 0
this.docs[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.docs[] elements == 0
norms[] == [105, 100, 102, 103, 100, 102, 103, 102, 104, 103, 101, 104, 102, 106, 100, 104, 105, 106, 101, 106, 101, 106, 105, 101]
norms[this.doc] == 105
this.weight.getClass() == orig(weight.getClass())
this.weight.getClass() != orig(similarity.getClass())
this.termDocs.getClass() == orig(td.getClass())
this.termDocs.getClass() != orig(similarity.getClass())
this.docs[] elements == this.doc
norms[] elements > this.doc
norms[] elements > org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE
norms[] elements > size(this.norms[])
==========
org.apache.lucene.search.TermScorer.doc():::ENTER
this.doc == this.docs[this.pointer]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.norms[] elements > this.doc
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > this.freqs[this.pointerMax-1]
this.doc <= size(this.norms[])-1

this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc != this.freqs[this.doc]
this.doc < this.freqs[this.pointerMax-1]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.pointerMax-1]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.pointerMax-1]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
this.pointer < this.pointerMax
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.pointerMax-1]
this.pointer != this.freqs[this.pointer]
this.pointer < this.freqs[this.pointerMax-1]
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax < this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.docs[this.doc]
this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]

```
================================================================================
org.apache.lucene.search.TermScorer.doc():::EXIT
this.weight == orig(this.weight)
this.termDocs == orig(this.termDocs)
this.norms == orig(this.norms)
this.norms[] == orig(this.norms[])
this.weightValue == orig(this.weightValue)
this.doc == return
this.docs == orig(this.docs)
this.docs[] == orig(this.docs[])
this.freqs == orig(this.freqs)
this.freqs[] == orig(this.freqs[])
this.pointer == orig(this.pointer)
this.pointerMax == orig(this.pointerMax)
this.scoreCache == orig(this.scoreCache)
this.scoreCache[] == orig(this.scoreCache[])
return == orig(this.doc)
return == this.docs[this.pointer]
return == this.docs[orig(this.pointer)]
return == orig(this.docs[post(this.pointer)])
return == orig(this.docs[this.pointer])
size(this.norms[])-1 == this.docs[this.pointerMax-1]
size(this.norms[])-1 == this.docs[orig(this.pointerMax)-1]
size(this.norms[])-1 == orig(this.docs[post(this.pointerMax)-1])
size(this.norms[])-1 == orig(this.docs[this.pointerMax-1])
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[orig(this.pointerMax)]
this.docs[this.pointerMax] == orig(this.freqs[post(this.pointerMax)])
this.docs[this.pointerMax] == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[this.pointerMax])
this.docs[] one of  [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7,
9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of  [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19,
16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of  18, 19
this.norms[this.pointerMax] one of  101, 106
this.norms[this.pointerMax-1] one of  101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of  31, 40
this.weight.getClass() == orig(this.weight.getClass())
this.termDocs.getClass() == orig(this.termDocs.getClass())
this.norms[] elements > return
this.norms[] elements > this.docs[return]
this.norms[] elements > this.freqs[this.pointerMax-1]
this.norms[] elements > this.freqs[return]
this.docs[] elements < this.norms[this.pointerMax-1]
this.docs[] elements < this.norms[return]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
this.freqs[] elements < this.norms[this.pointerMax-1]
this.freqs[] elements < this.norms[return]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
this.pointer < this.pointerMax
this.pointer <= return
this.pointer < this.norms[this.pointerMax-1]
this.pointer < this.norms[return]
this.pointer != this.freqs[this.pointer]
this.pointer < this.freqs[this.pointerMax-1]
```

this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax < this.norms[return]
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax < this.freqs[this.pointerMax-1]
this.pointerMax != this.freqs[return]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[return]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[return]
return <= size(this.norms[])-1
return < size(this.docs[])-1
return < this.norms[this.pointer]
return < this.norms[this.pointerMax]
return < this.norms[this.pointerMax-1]
return < this.norms[return]
return >= this.docs[this.pointerMax]
return < this.freqs[this.pointerMax-1]
return != this.freqs[return]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) < this.norms[return]
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[]) != this.freqs[return]
size(this.norms[])-1 >= this.docs[return]
size(this.norms[])-1 != this.freqs[return]
size(this.docs[])-1 > this.docs[return]
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.docs[return]
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.freqs[return]
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[return]
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] > this.freqs[return]
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.docs[return]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.freqs[return]
this.norms[return] > this.docs[this.pointerMax]
this.norms[return] > this.docs[return]
this.norms[return] > this.freqs[this.pointer]
this.norms[return] > this.freqs[this.pointerMax-1]
this.norms[return] > this.freqs[return]
this.docs[this.pointerMax] <= this.docs[return]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] <= this.freqs[return]
this.docs[return] != this.freqs[this.pointer]
this.docs[return] < this.freqs[this.pointerMax-1]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
this.freqs[this.pointerMax-1] >= this.freqs[return]
==========
org.apache.lucene.search.TermScorer.next():::ENTER
this.doc == this.docs[this.pointer]
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == 0
this.norms[] elements > this.doc
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.doc <= size(this.norms[])-1

this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements >= this.docs[this.pointerMax]
this.pointer <= this.pointerMax
this.pointer < this.norms[this.doc]
this.pointerMax < this.norms[this.doc]
this.pointerMax >= this.docs[this.pointerMax]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.docs[])-1 > this.docs[this.doc]
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.docs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.pointerMax] <= this.freqs[this.pointer]
==========
org.apache.lucene.search.TermScorer.next():::EXIT112
this.docs[] == orig(this.docs[])
this.freqs[] == orig(this.freqs[])
this.pointerMax == this.docs[this.pointer]
this.pointerMax == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == this.docs[orig(this.doc)]
this.pointerMax == this.docs[orig(this.pointerMax)]
this.pointerMax == this.freqs[this.pointer]
this.pointerMax == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == this.freqs[orig(this.doc)]
this.pointerMax == this.freqs[orig(this.pointerMax)]
this.pointerMax == orig(this.docs[post(this.pointer)])
this.pointerMax == orig(this.freqs[post(this.pointer)])
this.pointerMax == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.docs[this.doc])
this.pointerMax == orig(this.freqs[this.doc])
this.pointerMax == orig(this.docs[this.pointerMax])
this.pointerMax == orig(this.freqs[this.pointerMax])
orig(this.doc) == this.docs[this.pointer-1]
orig(this.doc) == this.docs[orig(this.pointer)]
orig(this.doc) == orig(this.docs[post(this.pointer)-1])
this.norms[this.pointer-1] == this.norms[orig(this.pointer)]
this.norms[this.pointer-1] == orig(this.norms[this.pointer])
this.freqs[this.pointer-1] == this.freqs[orig(this.pointer)]
this.freqs[this.pointer-1] == orig(this.freqs[this.pointer])
this.doc == 2147483647
this.pointer one of 1, 18, 19
this.pointerMax == 0
return == false
orig(this.doc) one of 0, 23
orig(this.pointer) one of 0, 17, 18

this.norms[this.pointer] one of 100, 101, 106
this.norms[this.pointer-1] one of 101, 105, 106
this.norms[this.pointerMax] == 105
this.norms[orig(this.doc)] one of 101, 105
this.freqs[this.pointer-1] one of 0, 31, 40
this.norms[] elements < this.doc
this.norms[] elements > this.freqs[this.pointer-1]
this.docs[] elements < this.doc
this.freqs[] elements < this.doc
this.doc > this.pointer
this.doc > orig(this.doc)
this.doc > orig(this.pointer)
this.doc > orig(this.pointerMax)
this.doc > this.norms[this.pointer]
this.doc > this.norms[this.pointer-1]
this.doc > this.norms[orig(this.doc)]
this.doc > this.norms[orig(this.pointerMax)]
this.doc > this.docs[this.pointerMax]
this.doc > this.freqs[this.pointer-1]
this.doc > this.freqs[this.pointerMax]
this.docs[] elements >= this.pointerMax
this.docs[] elements <= orig(this.doc)
orig(this.pointer) in this.docs[]
this.docs[] elements < this.norms[this.pointer-1]
this.docs[] elements <= this.freqs[this.pointer-1]
this.freqs[] elements >= this.pointerMax
this.freqs[] elements < this.norms[this.pointer-1]
this.freqs[] elements <= this.freqs[this.pointer-1]
this.pointer > this.pointerMax
this.pointer != orig(this.doc)
this.pointer > orig(this.pointer)
orig(this.pointerMax)
this.pointer >= orig(this.pointerMax)
this.pointer < this.norms[this.pointer-1]
this.pointer > this.docs[this.pointerMax]
this.pointer != this.freqs[this.pointer-1]
this.pointer > this.freqs[this.pointerMax]
this.pointerMax <= orig(this.doc)
this.pointerMax <= orig(this.pointer)
this.pointerMax <= orig(this.pointerMax)
this.pointerMax < this.norms[this.pointer-1]
this.pointerMax <= this.docs[this.pointerMax]
this.pointerMax <= this.freqs[this.pointer-1]
this.pointerMax <= this.freqs[this.pointerMax]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointer-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointer-1]
orig(this.doc) >= orig(this.pointerMax)
orig(this.doc) < this.norms[this.pointer-1]
orig(this.doc) <= this.freqs[this.pointer-1]
orig(this.pointer) < this.norms[this.pointer-1]
orig(this.pointer) <= this.freqs[this.pointer-1]
orig(this.pointerMax) < this.norms[this.pointer-1]
orig(this.pointerMax) <= this.freqs[this.pointer-1]
size(this.norms[]) < this.norms[this.pointer-1]
size(this.norms[]) != this.freqs[this.pointer-1]
size(this.norms[])-1 != this.freqs[this.pointer-1]
this.norms[this.pointer] != this.norms[this.pointer-1]
this.norms[this.pointer] != this.norms[this.pointerMax]
this.norms[this.pointer] <= this.norms[orig(this.pointerMax)]
this.norms[this.pointer] > this.freqs[this.pointer-1]
this.norms[this.pointer-1] >= this.norms[orig(this.doc)]
this.norms[this.pointer-1] > this.docs[this.pointerMax]
this.norms[this.pointer-1] > this.freqs[this.pointer-1]

this.norms[this.pointer-1] > this.freqs[this.pointerMax]
this.norms[this.pointerMax] >= this.norms[orig(this.doc)]
this.norms[this.pointerMax] > this.freqs[this.pointer-1]
this.norms[orig(this.doc)] <= this.norms[orig(this.pointerMax)]
this.norms[orig(this.doc)] > this.freqs[this.pointer-1]
this.norms[orig(this.pointerMax)] > this.freqs[this.pointer-1]
this.docs[this.pointerMax] <= this.freqs[this.pointer-1]
this.freqs[this.pointer-1] >= this.freqs[this.pointerMax]
==========
org.apache.lucene.search.TermScorer.next():::EXIT112;condition="not(return == true)"
==========
org.apache.lucene.search.TermScorer.next():::EXIT116
this.doc == this.docs[this.pointer]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == orig(this.freqs[post(this.pointerMax)])
this.docs[this.pointerMax] == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.docs[this.pointerMax])
this.docs[this.pointerMax] == orig(this.freqs[this.pointerMax])
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
return == true
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.freqs[orig(this.pointer)] >= 1
orig(this.docs[post(this.pointerMax)-1]) one of 0, 23
orig(this.freqs[post(this.pointerMax)-1]) one of 0, 31, 40
this.norms[] elements > this.doc
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > orig(this.docs[post(this.doc)])
this.norms[] elements > orig(this.freqs[post(this.doc)])
this.norms[] elements > this.freqs[this.pointerMax-1]
this.norms[] elements > orig(this.docs[post(this.pointerMax)-1])
this.norms[] elements > orig(this.freqs[post(this.pointerMax)-1])
this.doc <= size(this.norms[])-1
this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc < this.norms[orig(this.doc)]
this.doc < this.norms[orig(this.pointer)]
this.doc < this.norms[orig(this.pointerMax)]
this.doc != this.freqs[this.doc]
this.doc < this.freqs[this.pointerMax-1]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.pointerMax-1]
orig(this.docs[post(this.doc)]) in this.docs[]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
orig(this.docs[post(this.pointerMax)-1]) in this.docs[]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.pointerMax-1]

orig(this.freqs[post(this.doc)]) in this.freqs[]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
orig(this.freqs[post(this.pointerMax)-1]) in this.freqs[]
this.pointer < this.pointerMax
this.pointer <= orig(this.pointerMax)
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.pointerMax-1]
this.pointer < this.freqs[this.pointerMax-1]
this.pointer <= orig(this.docs[post(this.pointer)])
this.pointer <= orig(this.docs[post(this.pointerMax)-1])
this.pointer <= orig(this.freqs[post(this.pointerMax)-1])
this.pointerMax > orig(this.pointer)
orig(this.pointerMax)
this.pointerMax >= orig(this.pointerMax)
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax != orig(this.freqs[post(this.doc)])
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax > this.docs[orig(this.pointerMax)]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax < this.freqs[this.pointerMax-1]
this.pointerMax != this.freqs[orig(this.doc)]
this.pointerMax != this.freqs[orig(this.pointer)]
this.pointerMax > this.freqs[orig(this.pointerMax)]
this.pointerMax != orig(this.freqs[post(this.pointer)])
this.pointerMax != orig(this.docs[post(this.pointerMax)-1])
this.pointerMax != orig(this.freqs[post(this.pointerMax)-1])
this.pointerMax != orig(this.freqs[this.doc])
this.pointerMax != orig(this.freqs[this.pointer])
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.doc)])
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.pointerMax)-1])
orig(this.doc) < size(this.norms[])-1
orig(this.doc) < this.norms[this.doc]
orig(this.doc) < this.norms[this.pointerMax-1]
orig(this.doc) < this.freqs[this.pointerMax-1]
orig(this.doc) != this.freqs[orig(this.doc)]
orig(this.doc) <= orig(this.docs[post(this.pointer)])
orig(this.doc) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.doc) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.docs[]) elements < this.norms[this.doc]
orig(this.docs[]) elements < this.norms[this.pointerMax-1]
orig(this.docs[]) elements >= this.docs[this.pointerMax]
orig(this.docs[]) elements < this.freqs[this.pointerMax-1]
orig(this.docs[]) elements <= orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[]) elements <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.freqs[]) elements < this.norms[this.doc]
orig(this.freqs[]) elements < this.norms[this.pointerMax-1]
orig(this.freqs[]) elements >= this.docs[this.pointerMax]
orig(this.freqs[]) elements <= this.freqs[this.pointerMax-1]
orig(this.freqs[]) elements <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.pointer) < this.norms[this.doc]
orig(this.pointer) < this.norms[this.pointerMax-1]
orig(this.pointer) < this.freqs[this.pointerMax-1]
orig(this.pointer) != this.freqs[orig(this.pointer)]
orig(this.pointer) <= orig(this.docs[post(this.pointer)])
orig(this.pointer) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.pointer) <= orig(this.freqs[post(this.pointerMax)-1])

orig(this.pointerMax) < this.norms[this.doc]
orig(this.pointerMax) < this.norms[this.pointerMax-1]
orig(this.pointerMax) != this.freqs[this.doc]
orig(this.pointerMax) != this.freqs[this.pointer]
orig(this.pointerMax) < this.freqs[this.pointerMax-1]
orig(this.pointerMax) != this.freqs[orig(this.doc)]
orig(this.pointerMax) != this.freqs[orig(this.pointer)]
orig(this.pointerMax) != this.freqs[orig(this.pointerMax)]
orig(this.pointerMax) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.pointerMax) <= orig(this.freqs[post(this.pointerMax)-1])
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) != orig(this.freqs[post(this.doc)])
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[]) != orig(this.freqs[post(this.pointerMax)-1])
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.norms[])-1 >= orig(this.docs[post(this.doc)])
size(this.norms[])-1 != orig(this.freqs[post(this.doc)])
size(this.norms[])-1 > this.docs[orig(this.pointer)]
size(this.norms[])-1 != orig(this.freqs[post(this.pointerMax)-1])
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 > orig(this.docs[post(this.doc)])
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
size(this.docs[])-1 > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > orig(this.docs[post(this.doc)])
this.norms[this.doc] > orig(this.freqs[post(this.doc)])
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.docs[orig(this.doc)]
this.norms[this.doc] > this.docs[orig(this.pointer)]
this.norms[this.doc] > this.docs[orig(this.pointerMax)]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.doc] > this.freqs[orig(this.doc)]
this.norms[this.doc] > this.freqs[orig(this.pointer)]
this.norms[this.doc] > this.freqs[orig(this.pointerMax)]
this.norms[this.doc] > orig(this.docs[post(this.pointer)])
this.norms[this.doc] > orig(this.freqs[post(this.pointer)])
this.norms[this.doc] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.doc] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.doc] > orig(this.docs[this.doc])
this.norms[this.doc] > orig(this.freqs[this.doc])
this.norms[this.doc] > orig(this.freqs[this.pointer])
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > orig(this.docs[post(this.doc)])
this.norms[this.pointer] > orig(this.freqs[post(this.doc)])
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.pointer] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > orig(this.docs[post(this.doc)])
this.norms[this.pointerMax] > orig(this.freqs[post(this.doc)])
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.pointerMax] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.pointerMax-1] != this.norms[orig(this.pointerMax)]
this.norms[this.pointerMax-1] > this.docs[this.doc]

this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > orig(this.docs[post(this.doc)])
this.norms[this.pointerMax-1] > orig(this.freqs[post(this.doc)])
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.docs[orig(this.doc)]
this.norms[this.pointerMax-1] > this.docs[orig(this.pointer)]
this.norms[this.pointerMax-1] > this.docs[orig(this.pointerMax)]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.freqs[orig(this.doc)]
this.norms[this.pointerMax-1] > this.freqs[orig(this.pointer)]
this.norms[this.pointerMax-1] > this.freqs[orig(this.pointerMax)]
this.norms[this.pointerMax-1] > orig(this.docs[post(this.pointer)])
this.norms[this.pointerMax-1] > orig(this.freqs[post(this.pointer)])
this.norms[this.pointerMax-1] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.pointerMax-1] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.pointerMax-1] > orig(this.docs[this.doc])
this.norms[this.pointerMax-1] > orig(this.freqs[this.doc])
this.norms[this.pointerMax-1] > orig(this.freqs[this.pointer])
this.norms[orig(this.doc)] > this.docs[this.doc]
this.norms[orig(this.doc)] > this.freqs[this.doc]
this.norms[orig(this.doc)] > orig(this.docs[post(this.doc)])
this.norms[orig(this.doc)] > orig(this.freqs[post(this.doc)])
this.norms[orig(this.doc)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.doc)] > orig(this.docs[post(this.pointerMax)-1])
this.norms[orig(this.doc)] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[orig(this.pointer)] > this.docs[this.doc]
this.norms[orig(this.pointer)] > this.freqs[this.doc]
this.norms[orig(this.pointer)] > orig(this.docs[post(this.doc)])
this.norms[orig(this.pointer)] > orig(this.freqs[post(this.doc)])
this.norms[orig(this.pointer)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.pointer)] > orig(this.docs[post(this.pointerMax)-1])
this.norms[orig(this.pointer)] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[orig(this.pointerMax)] > this.docs[this.doc]
this.norms[orig(this.pointerMax)] > this.freqs[this.doc]
this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.doc)])
this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.doc)])
this.norms[orig(this.pointerMax)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.pointerMax)-1])
this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.pointerMax)-1])
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] >= this.docs[orig(this.pointerMax)]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] >= this.docs[orig(this.pointerMax)]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.freqs[orig(this.pointerMax)]
this.freqs[this.doc] != orig(this.docs[post(this.pointer)])
this.freqs[this.doc] != orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[post(this.doc)]) >= this.docs[this.pointerMax]
orig(this.docs[post(this.doc)]) != this.freqs[this.pointer]
orig(this.docs[post(this.doc)]) < this.freqs[this.pointerMax-1]
orig(this.docs[post(this.doc)]) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[post(this.doc)]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.freqs[post(this.doc)]) >= this.docs[this.pointerMax]
orig(this.freqs[post(this.doc)]) <= this.freqs[this.pointerMax-1]
orig(this.freqs[post(this.doc)]) <= orig(this.freqs[post(this.pointerMax)-1])
this.docs[this.pointerMax] <= this.docs[orig(this.doc)]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] <= this.freqs[orig(this.doc)]
this.docs[this.pointerMax] < this.freqs[orig(this.pointer)]

this.docs[this.pointerMax] <= orig(this.docs[post(this.pointer)])
this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointer)])
this.docs[this.pointerMax] <= orig(this.docs[post(this.pointerMax)-1])
this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointerMax)-1])
this.docs[this.pointerMax] <= orig(this.docs[this.doc])
this.docs[this.pointerMax] <= orig(this.freqs[this.doc])
this.docs[orig(this.doc)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.doc)] != this.freqs[orig(this.pointer)]
this.docs[orig(this.pointer)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.pointer)] != this.freqs[orig(this.doc)]
this.docs[orig(this.pointerMax)] < this.freqs[this.pointer]
this.docs[orig(this.pointerMax)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.pointerMax)] < this.freqs[orig(this.pointer)]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
this.freqs[orig(this.pointerMax)]
orig(this.freqs[post(this.pointer)])
this.freqs[this.pointer] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[this.pointerMax-1] >= this.freqs[orig(this.doc)]
this.freqs[this.pointerMax-1] >= this.freqs[orig(this.pointer)]
this.freqs[this.pointerMax-1] > this.freqs[orig(this.pointerMax)]
this.freqs[this.pointerMax-1] > orig(this.docs[post(this.pointer)])
this.freqs[this.pointerMax-1] >= orig(this.freqs[post(this.pointer)])
this.freqs[this.pointerMax-1] > orig(this.docs[post(this.pointerMax)-1])
orig(this.freqs[post(this.pointerMax)-1])
this.freqs[this.pointerMax-1] > orig(this.docs[this.doc])
this.freqs[this.pointerMax-1] >= orig(this.freqs[this.doc])
this.freqs[this.pointerMax-1] >= orig(this.freqs[this.pointer])
this.freqs[orig(this.doc)] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[orig(this.pointerMax)]
this.freqs[orig(this.pointer)] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[orig(this.pointer)] != orig(this.docs[this.doc])
orig(this.freqs[this.pointer])
this.freqs[orig(this.pointerMax)] != orig(this.docs[post(this.pointer)])
this.freqs[orig(this.pointerMax)] != orig(this.freqs[post(this.pointer)])
this.freqs[orig(this.pointerMax)] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[orig(this.pointerMax)] != orig(this.freqs[post(this.pointerMax)-1])
this.freqs[orig(this.pointerMax)] != orig(this.freqs[this.pointer])
orig(this.docs[post(this.pointer)]) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[post(this.pointer)]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.freqs[post(this.pointer)]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.docs[post(this.pointerMax)-1]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.docs[post(this.pointerMax)-1]) >= orig(this.docs[this.doc])
orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.docs[this.doc])
orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.freqs[this.doc])
orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.freqs[this.pointer])
this.doc - this.docs[orig(this.pointerMax)] - orig(this.docs[post(this.pointer)]) == 0
23 * this.pointer - 23 * orig(this.pointer) - orig(this.docs[post(this.pointerMax)-1]) == 0
2 * this.docs[this.doc] - 2 * orig(this.docs[post(this.doc)]) - 5 * this.docs[orig(this.pointerMax)] == 0
==========
org.apache.lucene.search.TermScorer.next():::EXIT116;condition="return == true"
==========
org.apache.lucene.search.TermScorer.next():::EXIT
this.weight == orig(this.weight)
this.termDocs == orig(this.termDocs)
this.norms == orig(this.norms)
this.norms[] == orig(this.norms[])
this.weightValue == orig(this.weightValue)
this.docs == orig(this.docs)
this.freqs == orig(this.freqs)
this.freqs[] >= orig(this.docs[]) (lexically)
this.scoreCache == orig(this.scoreCache)
this.scoreCache[] == orig(this.scoreCache[])
this.docs[this.pointerMax] == orig(this.docs[post(this.pointerMax)])

this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] == orig(this.docs[post(org.apache.lucene.search.TermScorer.SC
1])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SC
1])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] == orig(this.docs[this.pointerMax])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] == orig(this.freqs[this.pointerMax])
this.freqs[this.pointerMax] == orig(this.freqs[post(this.pointerMax)])
(this.doc == 2147483647) <==> (return == false)
(this.doc == 2147483647) <==> (this.doc > org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)
(this.doc == 2147483647) <==> (this.doc > size(this.norms[]))
(this.doc == 2147483647) <==> (this.doc > this.docs[this.pointer])
(this.doc == 2147483647) <==> (this.doc > this.norms[orig(this.doc)])
(this.doc == 2147483647) <==> (this.doc > this.norms[orig(this.pointer)])
(this.doc == 2147483647) <==> (this.doc > this.norms[orig(this.pointerMax)])
(this.doc == 2147483647) <==> (this.doc > this.norms[this.pointerMax])
(this.doc == 2147483647) <==> (this.doc > this.norms[this.pointer])
(this.doc == 2147483647) <==> (this.docs[this.pointerMax] >= this.freqs[this.pointer])
(this.doc == 2147483647) <==> (this.freqs[this.pointer] <= this.freqs[this.pointerMax])
(this.doc == 2147483647) <==> (this.freqs[this.pointer] == 0)
(this.doc == 2147483647) <==> (this.norms[this.pointerMax] == 105)
(this.doc == 2147483647) <==> (this.pointer > orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) <==> (this.pointer > this.docs[this.pointer])
(this.doc == 2147483647) <==> (this.pointer > this.pointerMax)
(this.doc == 2147483647) <==> (this.pointerMax <= orig(this.pointer))
(this.doc == 2147483647) <==> (this.pointerMax <= this.docs[this.pointerMax])
(this.doc == 2147483647) <==> (this.pointerMax <= this.freqs[this.pointerMax])
(this.doc == 2147483647) <==> (this.pointerMax == 0)
(this.doc == 2147483647) <==> (this.pointerMax == orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) <==> (this.pointerMax == orig(this.freqs[this.doc]))
(this.doc == 2147483647) <==> (this.pointerMax == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-
1])
(this.doc == 2147483647) <==> (this.pointerMax == this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) <==> (this.pointerMax == this.freqs[orig(this.doc)])
(this.doc == 2147483647) <==> (this.pointerMax == this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) <==> (this.pointerMax == this.freqs[this.pointer])
(this.doc == 2147483647) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.pointer)-
1]))
(this.doc == 2147483647) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < orig(this.norms[post(this.pointer)-
1]))
(this.doc == 2147483647) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.doc) < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.doc) < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.doc) <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.doc) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.doc) <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.doc) <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.doc) == orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.doc) == this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.doc) == this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.doc) >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.doc) >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.doc) >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.doc) >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.doc) >= orig(this.pointerMax))
(this.doc == 2147483647) ==> (orig(this.doc) >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.doc) >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.doc) >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.doc) >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.doc) >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.doc) >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.doc) one of 0, 23)
(this.doc == 2147483647) ==> (orig(this.docs[]) elements < orig(this.norms[post(this.pointer)-1]))

(this.doc == 2147483647) ==> (orig(this.docs[]) elements < this.doc)
(this.doc == 2147483647) ==> (orig(this.docs[]) elements < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= orig(this.doc))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.docs[]) elements >= this.pointerMax)
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)-1]) <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)-1]) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)-1]) >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)-1]) >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)-1]) >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)-1]) one of 0, 23)
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)]) <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)]) <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)]) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.docs[post(this.pointer)]) == 0)
(this.doc == 2147483647) ==> (orig(this.docs[this.doc]) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.docs[this.doc]) == 0)
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements < this.doc)
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.freqs[]) elements >= this.pointerMax)
(this.doc == 2147483647) ==> (orig(this.freqs[post(this.pointer)-1]) >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.freqs[post(this.pointer)-1]) >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.freqs[post(this.pointer)-1]) one of 0, 31, 40)
(this.doc == 2147483647) ==> (orig(this.freqs[post(this.pointer)]) <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.freqs[post(this.pointer)]) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.freqs[post(this.pointer)]) == 0)
(this.doc == 2147483647) ==> (orig(this.freqs[this.doc]) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.freqs[this.doc]) == 0)
(this.doc == 2147483647) ==> (orig(this.freqs[this.pointer]) one of 0, 31, 40)
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.docs[this.doc]))

(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) > orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.norms[post(this.pointer)-1]) one of 101, 105, 106)
(this.doc == 2147483647) ==> (orig(this.pointer) < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.pointer) < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.pointer) <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.pointer) <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.pointer) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.pointer) <= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.pointer) <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.pointer) <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.pointer) >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.pointer) >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.pointer) >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.pointer) >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.pointer) >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.pointer) >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.pointer) >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.pointer) >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.pointer) >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.pointer) >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.pointer) in orig(this.docs[]))
(this.doc == 2147483647) ==> (orig(this.pointer) in this.docs[])
(this.doc == 2147483647) ==> (orig(this.pointer) one of 0, 17, 18)
(this.doc == 2147483647) ==> (orig(this.pointerMax)
(this.doc == 2147483647) ==> (orig(this.pointerMax) < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (orig(this.pointerMax) <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (orig(this.pointerMax) >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (size(this.docs[])-1 > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (size(this.docs[])-1 > orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (size(this.docs[])-1 > orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (size(this.docs[])-1 > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (size(this.docs[])-1 > this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (size(this.docs[])-1 > this.freqs[this.pointer])
(this.doc == 2147483647) ==> (size(this.norms[]) != orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (size(this.norms[]) != this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (size(this.norms[]) < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (size(this.norms[]) < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (size(this.norms[])-1 != orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (size(this.norms[])-1 != this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (size(this.norms[])-1 > orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (size(this.norms[])-1 > orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (size(this.norms[])-1 > orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (size(this.norms[])-1 > orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (size(this.norms[])-1 > this.docs[orig(this.doc)])

(this.doc == 2147483647) ==> (size(this.norms[])-1 > this.docs[this.pointer])
(this.doc == 2147483647) ==> (size(this.norms[])-1 > this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (size(this.norms[])-1 > this.freqs[this.pointer])
(this.doc == 2147483647) ==> (size(this.norms[])-1 >= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (size(this.norms[])-1 >= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.doc > orig(this.doc))
(this.doc == 2147483647) ==> (this.doc > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.doc > orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.doc > orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.doc > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.doc > orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.doc > orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.doc > orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.doc > orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.doc > orig(this.pointer))
(this.doc == 2147483647) ==> (this.doc > orig(this.pointerMax))
(this.doc == 2147483647) ==> (this.doc > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1])
(this.doc == 2147483647) ==> (this.doc > this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.doc > this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.doc > this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.doc > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.doc > this.docs[this.pointerMax])
(this.doc == 2147483647) ==> (this.doc > this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.doc > this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.doc > this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.doc > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.doc > this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.doc > this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.doc > this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (this.doc > this.pointer)
(this.doc == 2147483647) ==> (this.doc > this.pointerMax)
(this.doc == 2147483647) ==> (this.docs[] <= orig(this.freqs[]) (lexically))
(this.doc == 2147483647) ==> (this.docs[] == orig(this.docs[]))
(this.doc == 2147483647) ==> (this.docs[] elements < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[] elements < this.doc)
(this.doc == 2147483647) ==> (this.docs[] elements < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[] elements <= orig(this.doc))
(this.doc == 2147483647) ==> (this.docs[] elements <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[] elements <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[] elements <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.docs[] elements <= this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[] elements <= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[] elements <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[] elements <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[] elements >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[] elements >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[] elements >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[] elements >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[] elements >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[] elements >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.docs[] elements >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (this.docs[] elements >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[] elements >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.docs[] elements >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.docs[] elements >= this.pointerMax)
(this.doc == 2147483647) ==> (this.docs[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
(this.doc == 2147483647) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] < orig(this.norms[post(this.poi
1]))
(this.doc == 2147483647) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.docs[post(this.poi
1]))
(this.doc == 2147483647) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.freqs[post(this.poi

1]))
(this.doc == 2147483647) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] <= this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.docs[orig(this.doc)] == 0)
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointer)] one of 0, 23)
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] <= this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.docs[orig(this.pointerMax)] == 0)
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.docs[this.pointerMax])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.docs[this.pointer-1] one of 0, 23)
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.docs[this.pointerMax] >= this.freqs[orig(this.pointerMax)])

(this.doc == 2147483647) ==> (this.docs[this.pointerMax] one of 0, 2)
(this.doc == 2147483647) ==> (this.docs[this.pointer] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= this.docs[this.pointerMax])
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.docs[this.pointer] <= this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.docs[this.pointer] == 0)
(this.doc == 2147483647) ==> (this.freqs[] == orig(this.freqs[]))
(this.doc == 2147483647) ==> (this.freqs[] elements < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[] elements < this.doc)
(this.doc == 2147483647) ==> (this.freqs[] elements < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (this.freqs[] elements <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[] elements <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.freqs[] elements <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.freqs[] elements <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.freqs[] elements >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[] elements >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[] elements >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[] elements >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.docs[this.pointer])
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.freqs[] elements >= this.pointerMax)
(this.doc == 2147483647) ==> (this.freqs[] one of [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
(this.doc == 2147483647) ==> (this.freqs[orig(this.doc)] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.doc)] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.doc)] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.doc)] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.doc)] <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.freqs[orig(this.doc)] == 0)
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] >= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointer)] one of 0, 31, 40)
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointerMax)] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointerMax)] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointerMax)] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointerMax)] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.freqs[orig(this.pointerMax)] == 0)
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] == orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] == this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] >= this.freqs[this.pointerMax])

(this.doc == 2147483647) ==> (this.freqs[this.pointer-1] one of 0, 31, 40)
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] >= orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] >= orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] >= orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] >= orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] >= this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] >= this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.freqs[this.pointerMax] one of 0, 7, 14)
(this.doc == 2147483647) ==> (this.freqs[this.pointer] < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer] <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer] <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer] <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.freqs[this.pointer] <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.freqs[this.pointer] <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[] elements < this.doc)
(this.doc == 2147483647) ==> (this.norms[] elements > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[] elements > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[] elements > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[] elements > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] <= orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] <= this.norms[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] <= this.norms[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[orig(this.doc)] one of 101, 105)
(this.doc == 2147483647) ==> (this.norms[orig(this.pointer)] > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.pointer)] > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.pointer)] > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[orig(this.pointer)] > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[orig(this.pointer)] one of 101, 105, 106)
(this.doc == 2147483647) ==> (this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[orig(this.pointerMax)] > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[orig(this.pointerMax)] > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] == this.norms[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[this.pointerMax])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.docs[this.pointer])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] > this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] >= this.norms[orig(this.doc)])
(this.doc == 2147483647) ==> (this.norms[this.pointer-1] one of 101, 105, 106)

(this.doc == 2147483647) ==> (this.norms[this.pointerMax] > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointerMax] > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointerMax] > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointerMax] > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointerMax] >= this.norms[orig(this.doc)])
(this.doc == 2147483647) ==> (this.norms[this.pointer] != orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointer] != this.norms[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.norms[this.pointer] != this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer] != this.norms[this.pointerMax])
(this.doc == 2147483647) ==> (this.norms[this.pointer] <= this.norms[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.norms[this.pointer] > orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointer] > orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.norms[this.pointer] > this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer] > this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.norms[this.pointer] one of 100, 101, 106)
(this.doc == 2147483647) ==> (this.pointer != orig(this.doc))
(this.doc == 2147483647) ==> (this.pointer != orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.pointer != orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.pointer != orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.pointer != this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.pointer != this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.pointer != this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.pointer != this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.pointer < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.pointer < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (this.pointer > orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.pointer > orig(this.freqs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.pointer > orig(this.freqs[this.doc]))
(this.doc == 2147483647) ==> (this.pointer > orig(this.pointer))
(this.doc == 2147483647) ==> (this.pointer > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1])
(this.doc == 2147483647) ==> (this.pointer > this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.pointer > this.docs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.pointer > this.docs[this.pointerMax])
(this.doc == 2147483647) ==> (this.pointer > this.freqs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.pointer > this.freqs[orig(this.pointerMax)])
(this.doc == 2147483647) ==> (this.pointer > this.freqs[this.pointerMax])
(this.doc == 2147483647) ==> (this.pointer > this.freqs[this.pointer])
(this.doc == 2147483647) ==> (this.pointer >= orig(this.pointerMax))
(this.doc == 2147483647) ==> (this.pointer one of 1, 18, 19)
(this.doc == 2147483647) ==> (this.pointerMax < orig(this.norms[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.pointerMax < this.norms[this.pointer-1])
(this.doc == 2147483647) ==> (this.pointerMax <= orig(this.doc))
(this.doc == 2147483647) ==> (this.pointerMax <= orig(this.docs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.pointerMax <= orig(this.freqs[post(this.pointer)-1]))
(this.doc == 2147483647) ==> (this.pointerMax <= orig(this.freqs[this.pointer]))
(this.doc == 2147483647) ==> (this.pointerMax <= orig(this.pointerMax))
(this.doc == 2147483647) ==> (this.pointerMax <= this.docs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.pointerMax <= this.docs[this.pointer-1])
(this.doc == 2147483647) ==> (this.pointerMax <= this.freqs[orig(this.pointer)])
(this.doc == 2147483647) ==> (this.pointerMax <= this.freqs[this.pointer-1])
(this.doc == 2147483647) ==> (this.pointerMax == orig(this.docs[post(this.pointer)]))
(this.doc == 2147483647) ==> (this.pointerMax == orig(this.docs[this.doc]))
(this.doc == 2147483647) ==> (this.pointerMax == this.docs[orig(this.doc)])
(this.doc == 2147483647) ==> (this.pointerMax == this.docs[this.pointer])
(this.pointerMax one of 18, 19) <==> (return == true)
(this.pointerMax one of 18, 19) <==> (this.doc < size(this.docs[])-1)
(this.pointerMax one of 18, 19) <==> (this.doc < this.norms[orig(this.doc)])
(this.pointerMax one of 18, 19) <==> (this.doc < this.norms[orig(this.pointer)])
(this.pointerMax one of 18, 19) <==> (this.doc < this.norms[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) <==> (this.doc < this.norms[this.pointerMax])
(this.pointerMax one of 18, 19) <==> (this.doc < this.norms[this.pointer])
(this.pointerMax one of 18, 19) <==> (this.doc <= size(this.norms[])-1)
(this.pointerMax one of 18, 19) <==> (this.doc == this.docs[this.pointer])

(this.pointerMax one of 18, 19) <==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] < this.freqs[this.pointer]
(this.pointerMax one of 18, 19) <==> (this.docs[orig(this.pointerMax)] < this.freqs[this.pointer])
(this.pointerMax one of 18, 19) <==> (this.docs[this.pointerMax] < this.freqs[this.pointer])
(this.pointerMax one of 18, 19) <==> (this.docs[this.pointer] != this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) <==> (this.freqs[orig(this.pointerMax)] != orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) <==> (this.freqs[orig(this.pointerMax)] != orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) <==> (this.freqs[this.pointer] > this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) <==> (this.freqs[this.pointer] >= 1)
(this.pointerMax one of 18, 19) <==> (this.norms[this.pointerMax] one of 101, 106)
(this.pointerMax one of 18, 19) <==> (this.pointer < this.pointerMax)
(this.pointerMax one of 18, 19) <==> (this.pointer <= orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) <==> (this.pointer <= this.docs[this.pointer])
(this.pointerMax one of 18, 19) <==> (this.pointerMax != orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) <==> (this.pointerMax != orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) <==> (this.pointerMax != this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) <==> (this.pointerMax != this.freqs[this.pointer])
(this.pointerMax one of 18, 19) <==> (this.pointerMax > orig(this.pointer))
(this.pointerMax one of 18, 19) <==> (this.pointerMax > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-
1])
(this.pointerMax one of 18, 19) <==> (this.pointerMax > this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) <==> (this.pointerMax > this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) <==> (this.pointerMax > this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) <==> (this.pointerMax > this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (2 * this.docs[this.doc] - 2 * orig(this.docs[post(this.doc)]) - 5 * this.docs[orig(this.pointerMax)]
== 0)
(this.pointerMax one of 18, 19) ==> (23 * this.pointer - 23 * orig(this.pointer) - orig(this.docs[post(this.pointerMax)-1])
== 0)
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.pointerMa
1]))
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-
1])
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < orig(this.norms[post(this.pointerMa
1]))
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-
1])
(this.pointerMax one of 18, 19) ==> (orig(this.doc) != this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (orig(this.doc) - this.docs[orig(this.pointer)] + this.docs[orig(this.pointerMax)] ==
0)
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < size(this.norms[])-1)
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (orig(this.doc) < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.doc) <= orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (orig(this.doc) <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.doc) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.doc) <= this.docs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements <= this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[]) elements >= this.freqs[this.pointerMax])

(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) != this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) <= this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_
1])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.doc)]) in this.docs[])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointer)]) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointer)]) <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointer)]) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointerMax)-1]) <= orig(this.freqs[post(this.pointerMax)-
1]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointerMax)-1]) >= orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointerMax)-1]) in this.docs[])
(this.pointerMax one of 18, 19) ==> (orig(this.docs[post(this.pointerMax)-1]) one of 0, 23)
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements <= this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[]) elements >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) != this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) <= this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_
1])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.doc)]) in this.freqs[])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointer)])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointer)]) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointer)]) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointerMax)-1])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointerMax)-1]) in this.freqs[])
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[post(this.pointerMax)-1]) one of 0, 31, 40)
(this.pointerMax one of 18, 19) ==> (orig(this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_
1])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.docs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.docs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.docs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.freqs[orig(this.doc)])

(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.doc)]) > this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.pointerMax)-1]) > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.pointerMax)-1]) > orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.pointerMax)-1]) > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.pointerMax)-1]) > orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.pointerMax)-1]) > orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (orig(this.norms[post(this.pointerMax)-1]) one of 101, 106)
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) != this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) < this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) <= orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointer) <= this.docs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) != this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) != this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) != this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) != this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) != this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax)
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) < this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (orig(this.pointerMax) <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (size(this.docs[])-1 <= this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (size(this.docs[])-1 > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (size(this.docs[])-1 > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (size(this.docs[])-1 > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) != orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) != orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) != this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (size(this.norms[]) < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 != orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 != orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 != this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 == this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 > this.docs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 >= orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (size(this.norms[])-1 >= this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.doc != this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.doc - this.docs[orig(this.pointerMax)] - orig(this.docs[post(this.pointer)]) == 0)
(this.pointerMax one of 18, 19) ==> (this.doc < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.doc < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.doc < this.freqs[this.pointerMax-1])

(this.pointerMax one of 18, 19) ==> (this.doc < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (this.doc < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.doc <= this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[] elements < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.docs[] elements < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[] elements < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[] elements < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (this.docs[] elements < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[] elements <= this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[] elements >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[] elements >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
(this.pointerMax one of 18, 19) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] < orig(this.norms[post(th
1]))
(this.pointerMax one of 18, 19) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] < this.freqs[orig(this.poin
(this.pointerMax one of 18, 19) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] < this.freqs[this.pointerM
1])
(this.pointerMax one of 18, 19) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.docs[post(th
1]))
(this.pointerMax one of 18, 19) ==> (this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.freqs[post(th
1]))
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.doc)] != this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.doc)] - this.docs[orig(this.pointerMax)] - orig(this.docs[this.doc])
== 0)
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.doc)] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.doc)] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.doc)] >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointer)] != this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointer)] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointer)] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointerMax)] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointerMax)] < this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointerMax)] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[orig(this.pointerMax)] one of 0, 2)
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] != this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] <= this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-
1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] >= this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[this.doc] >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] != this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZ
1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] > this.docs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] > this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] > this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] > this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] >= orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] >= orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax-1] >= this.docs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] < this.docs[this.pointerMax-1])

(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] < this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= this.docs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] <= this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] == 0)
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZ
1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointerMax] == this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] - this.docs[orig(this.pointerMax)] - orig(this.docs[post(this.pointer)])
== 0)
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] <= this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] >= this.docs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.docs[this.pointer] >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements <= this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.freqs[] elements >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.doc)] != orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.doc)] - this.freqs[orig(this.pointerMax)] - orig(this.freqs[this.doc])
== 0)
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.doc)] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointer)] != orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointer)] != orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointer)] - this.freqs[orig(this.pointerMax)] - orig(this.freqs[this.pointer])
== 0)
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointer)] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointer)] >= 1)
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)] != orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)] != orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)] != orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)]
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)]
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[orig(this.pointerMax)] one of 0, 7, 14)
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] != orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] != orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] != this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] != this.docs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] <= this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-
1])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] >= this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] >= this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] >= this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.doc] >= this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] > orig(this.docs[post(this.pointer)]))

(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] > orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] > this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] >= orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] >= orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] >= orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] >= this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] >= this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax-1] one of 31, 40)
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] < this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] <= this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointerMax] == 0)
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointer] != orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointer] - this.freqs[orig(this.pointerMax)] - orig(this.freqs[post(this.pointer)])
== 0)
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointer] < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.freqs[this.pointer] <= this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > this.doc)
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[] elements > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.doc)] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointer)] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] != orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[orig(this.pointerMax)] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.freqs[this.doc]))

(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.docs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.norms[this.doc] > this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] != this.norms[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.docs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.docs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.freqs[post(this.pointer)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.freqs[this.doc]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_S1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.docs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[orig(this.doc)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[orig(this.pointerMax)])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[this.pointerMax])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] > this.freqs[this.pointer])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax-1] one of 101, 106)
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] != orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] != this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointerMax] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > orig(this.docs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > this.docs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.norms[this.pointer] > this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointer < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.pointer < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.pointer < this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointer < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointer < this.norms[this.doc])

(this.pointerMax one of 18, 19) ==> (this.pointer < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointer <= orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.pointer <= orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.pointer <= orig(this.pointerMax))
(this.pointerMax one of 18, 19) ==> (this.pointerMax != orig(this.docs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.pointerMax != orig(this.freqs[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.pointerMax != orig(this.freqs[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.pointerMax != orig(this.freqs[this.pointer]))
(this.pointerMax one of 18, 19) ==> (this.pointerMax != this.freqs[orig(this.pointer)])
(this.pointerMax one of 18, 19) ==> (this.pointerMax != this.freqs[this.doc])
(this.pointerMax one of 18, 19) ==> (this.pointerMax < orig(this.norms[post(this.doc)]))
(this.pointerMax one of 18, 19) ==> (this.pointerMax < orig(this.norms[post(this.pointerMax)-1]))
(this.pointerMax one of 18, 19) ==> (this.pointerMax < this.docs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointerMax < this.freqs[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointerMax < this.norms[this.doc])
(this.pointerMax one of 18, 19) ==> (this.pointerMax < this.norms[this.pointerMax-1])
(this.pointerMax one of 18, 19) ==> (this.pointerMax >= orig(this.pointerMax))
this.norms[orig(this.pointerMax)] one of 101, 105, 106
this.docs[orig(this.pointerMax)] one of 0, 2
this.freqs[orig(this.pointerMax)] one of 0, 7, 14
this.weight.getClass() == orig(this.weight.getClass())
this.termDocs.getClass() == orig(this.termDocs.getClass())
this.norms[] elements > orig(this.doc)
this.norms[] elements > orig(this.pointer)
this.norms[] elements > orig(this.pointerMax)
this.norms[] elements > this.docs[orig(this.doc)]
this.norms[] elements > this.docs[orig(this.pointer)]
this.norms[] elements > this.docs[orig(this.pointerMax)]
this.norms[] elements > this.freqs[orig(this.doc)]
this.norms[] elements > this.freqs[orig(this.pointer)]
this.norms[] elements > this.freqs[orig(this.pointerMax)]
this.norms[] elements > orig(this.docs[post(this.pointer)])
this.norms[] elements > orig(this.freqs[post(this.pointer)])
this.norms[] elements > orig(this.docs[this.doc])
this.norms[] elements > orig(this.freqs[this.doc])
this.norms[] elements > orig(this.freqs[this.pointer])
this.doc >= orig(this.doc)
this.doc >= orig(this.pointer)
this.doc != this.norms[orig(this.doc)]
this.doc != this.norms[orig(this.pointer)]
this.doc != this.norms[orig(this.pointerMax)]
this.doc >= this.docs[orig(this.pointer)]
this.doc >= this.docs[orig(this.pointerMax)]
this.doc != this.freqs[orig(this.pointerMax)]
this.doc >= orig(this.docs[post(this.pointer)])
orig(this.doc) in this.docs[]
this.docs[] >= orig(this.docs[]) (elementwise)
orig(this.pointerMax) in this.docs[]
this.docs[] elements < this.norms[orig(this.doc)]
this.docs[] elements < this.norms[orig(this.pointer)]
this.docs[] elements < this.norms[orig(this.pointerMax)]
this.freqs[orig(this.pointerMax)] in this.docs[]
orig(this.docs[post(this.pointer)]) in this.docs[]
orig(this.docs[this.doc]) in this.docs[]
this.freqs[] >= orig(this.freqs[]) (elementwise)
this.freqs[] elements < this.norms[orig(this.doc)]
this.freqs[] elements < this.norms[orig(this.pointer)]
this.freqs[] elements < this.norms[orig(this.pointerMax)]
orig(this.freqs[post(this.pointer)]) in this.freqs[]
orig(this.freqs[this.doc]) in this.freqs[]
orig(this.freqs[this.pointer]) in this.freqs[]
this.pointer != this.pointerMax
this.pointer >= orig(this.pointer)

this.pointer < this.norms[orig(this.doc)]
this.pointer < this.norms[orig(this.pointer)]
this.pointer < this.norms[orig(this.pointerMax)]
this.pointer != this.freqs[this.pointer]
this.pointer != this.freqs[orig(this.doc)]
this.pointer != this.freqs[orig(this.pointerMax)]
this.pointerMax < this.norms[orig(this.doc)]
this.pointerMax < this.norms[orig(this.pointer)]
this.pointerMax < this.norms[orig(this.pointerMax)]
this.pointerMax >= this.docs[orig(this.pointerMax)]
this.pointerMax >= this.freqs[orig(this.pointerMax)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(this.doc)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(this.pointer)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(this.pointerMax)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[orig(this.doc)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[orig(this.pointer)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.pointer)])
orig(this.doc) <= size(this.norms[])-1
orig(this.doc) < size(this.docs[])-1
orig(this.doc) < this.norms[this.pointer]
orig(this.doc) < this.norms[this.pointerMax]
orig(this.doc) < this.norms[orig(this.doc)]
orig(this.doc) < this.norms[orig(this.pointer)]
orig(this.doc) < this.norms[orig(this.pointerMax)]
orig(this.doc) >= this.docs[this.pointerMax]
orig(this.doc) >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
orig(this.doc) >= this.freqs[this.pointerMax]
orig(this.docs[]) elements <= size(this.norms[])-1
orig(this.docs[]) elements < size(this.docs[])-1
orig(this.docs[]) elements < this.norms[this.pointer]
orig(this.docs[]) elements < this.norms[this.pointerMax]
orig(this.docs[]) elements < this.norms[orig(this.doc)]
orig(this.docs[]) elements < this.norms[orig(this.pointer)]
orig(this.docs[]) elements < this.norms[orig(this.pointerMax)]
orig(this.docs[]) elements >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.freqs[this.pointerMax] in orig(this.docs[])
orig(this.freqs[]) elements < this.norms[this.pointer]
orig(this.freqs[]) elements < this.norms[this.pointerMax]
orig(this.freqs[]) elements < this.norms[orig(this.doc)]
orig(this.freqs[]) elements < this.norms[orig(this.pointer)]
orig(this.freqs[]) elements < this.norms[orig(this.pointerMax)]
orig(this.freqs[]) elements >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
orig(this.pointer) < size(this.norms[])-1
orig(this.pointer) < size(this.docs[])-1
orig(this.pointer) < this.norms[this.pointer]
orig(this.pointer) < this.norms[this.pointerMax]
orig(this.pointer) < this.norms[orig(this.doc)]
orig(this.pointer) < this.norms[orig(this.pointer)]
orig(this.pointer) < this.norms[orig(this.pointerMax)]
orig(this.pointer) >= this.docs[this.pointerMax]
orig(this.pointer) >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
orig(this.pointer) <= this.docs[orig(this.pointer)]
orig(this.pointer) >= this.freqs[this.pointerMax]
orig(this.pointerMax) < size(this.norms[])-1
orig(this.pointerMax) < size(this.docs[])-1
orig(this.pointerMax) < this.norms[this.pointer]
orig(this.pointerMax) < this.norms[this.pointerMax]
orig(this.pointerMax) < this.norms[orig(this.doc)]
orig(this.pointerMax) < this.norms[orig(this.pointer)]
orig(this.pointerMax) < this.norms[orig(this.pointerMax)]
orig(this.pointerMax) >= this.docs[this.pointerMax]
orig(this.pointerMax) >= this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
orig(this.pointerMax) >= this.freqs[this.pointerMax]

size(this.norms[]) < this.norms[orig(this.doc)]
size(this.norms[]) < this.norms[orig(this.pointer)]
size(this.norms[]) < this.norms[orig(this.pointerMax)]
size(this.norms[]) != this.freqs[orig(this.doc)]
size(this.norms[]) != this.freqs[orig(this.pointer)]
size(this.norms[]) != orig(this.freqs[post(this.pointer)])
size(this.norms[]) != orig(this.freqs[this.doc])
size(this.norms[]) != orig(this.freqs[this.pointer])
size(this.norms[])-1 >= this.docs[orig(this.doc)]
size(this.norms[])-1 >= this.docs[orig(this.pointer)]
size(this.norms[])-1 > this.docs[orig(this.pointerMax)]
size(this.norms[])-1 != this.freqs[orig(this.doc)]
size(this.norms[])-1 != this.freqs[orig(this.pointer)]
size(this.norms[])-1 > this.freqs[orig(this.pointerMax)]
size(this.norms[])-1 >= orig(this.docs[post(this.pointer)])
size(this.norms[])-1 != orig(this.freqs[post(this.pointer)])
size(this.norms[])-1 >= orig(this.docs[this.doc])
size(this.norms[])-1 != orig(this.freqs[this.doc])
size(this.norms[])-1 != orig(this.freqs[this.pointer])
size(this.docs[])-1 > this.docs[orig(this.doc)]
size(this.docs[])-1 > this.docs[orig(this.pointer)]
size(this.docs[])-1 > this.docs[orig(this.pointerMax)]
size(this.docs[])-1 > this.freqs[orig(this.pointerMax)]
size(this.docs[])-1 > orig(this.docs[post(this.pointer)])
size(this.docs[])-1 > orig(this.docs[this.doc])
this.norms[this.pointer] > this.docs[orig(this.doc)]
this.norms[this.pointer] > this.docs[orig(this.pointer)]
this.norms[this.pointer] > this.docs[orig(this.pointerMax)]
this.norms[this.pointer] > this.freqs[orig(this.doc)]
this.norms[this.pointer] > this.freqs[orig(this.pointer)]
this.norms[this.pointer] > this.freqs[orig(this.pointerMax)]
this.norms[this.pointer] > orig(this.docs[post(this.pointer)])
this.norms[this.pointer] > orig(this.freqs[post(this.pointer)])
this.norms[this.pointer] > orig(this.docs[this.doc])
this.norms[this.pointer] > orig(this.freqs[this.doc])
this.norms[this.pointer] > orig(this.freqs[this.pointer])
this.norms[this.pointerMax] > this.docs[orig(this.doc)]
this.norms[this.pointerMax] > this.docs[orig(this.pointer)]
this.norms[this.pointerMax] > this.docs[orig(this.pointerMax)]
this.norms[this.pointerMax] > this.freqs[orig(this.doc)]
this.norms[this.pointerMax] > this.freqs[orig(this.pointer)]
this.norms[this.pointerMax] > this.freqs[orig(this.pointerMax)]
this.norms[this.pointerMax] > orig(this.docs[post(this.pointer)])
this.norms[this.pointerMax] > orig(this.freqs[post(this.pointer)])
this.norms[this.pointerMax] > orig(this.docs[this.doc])
this.norms[this.pointerMax] > orig(this.freqs[this.doc])
this.norms[this.pointerMax] > orig(this.freqs[this.pointer])
this.norms[orig(this.doc)] > this.docs[this.pointer]
this.norms[orig(this.doc)] > this.docs[this.pointerMax]
this.norms[orig(this.doc)] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.norms[orig(this.doc)] > this.docs[orig(this.doc)]
this.norms[orig(this.doc)] > this.docs[orig(this.pointer)]
this.norms[orig(this.doc)] > this.docs[orig(this.pointerMax)]
this.norms[orig(this.doc)] > this.freqs[this.pointer]
this.norms[orig(this.doc)] > this.freqs[this.pointerMax]
this.norms[orig(this.doc)] > this.freqs[orig(this.doc)]
this.norms[orig(this.doc)] > this.freqs[orig(this.pointer)]
this.norms[orig(this.doc)] > this.freqs[orig(this.pointerMax)]
this.norms[orig(this.doc)] > orig(this.docs[post(this.pointer)])
this.norms[orig(this.doc)] > orig(this.freqs[post(this.pointer)])
this.norms[orig(this.doc)] > orig(this.docs[this.doc])
this.norms[orig(this.doc)] > orig(this.freqs[this.doc])
this.norms[orig(this.doc)] > orig(this.freqs[this.pointer])

this.norms[orig(this.pointer)] > this.docs[this.pointer]
this.norms[orig(this.pointer)] > this.docs[this.pointerMax]
this.norms[orig(this.pointer)] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.norms[orig(this.pointer)] > this.docs[orig(this.doc)]
this.norms[orig(this.pointer)] > this.docs[orig(this.pointer)]
this.norms[orig(this.pointer)] > this.docs[orig(this.pointerMax)]
this.norms[orig(this.pointer)] > this.freqs[this.pointer]
this.norms[orig(this.pointer)] > this.freqs[this.pointerMax]
this.norms[orig(this.pointer)] > this.freqs[orig(this.doc)]
this.norms[orig(this.pointer)] > this.freqs[orig(this.pointer)]
this.norms[orig(this.pointer)] > this.freqs[orig(this.pointerMax)]
this.norms[orig(this.pointer)] > orig(this.docs[post(this.pointer)])
this.norms[orig(this.pointer)] > orig(this.freqs[post(this.pointer)])
this.norms[orig(this.pointer)] > orig(this.docs[this.doc])
this.norms[orig(this.pointer)] > orig(this.freqs[this.doc])
this.norms[orig(this.pointer)] > orig(this.freqs[this.pointer])
this.norms[orig(this.pointerMax)] > this.docs[this.pointer]
this.norms[orig(this.pointerMax)] > this.docs[this.pointerMax]
this.norms[orig(this.pointerMax)] > this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.norms[orig(this.pointerMax)] > this.docs[orig(this.doc)]
this.norms[orig(this.pointerMax)] > this.docs[orig(this.pointer)]
this.norms[orig(this.pointerMax)] > this.docs[orig(this.pointerMax)]
this.norms[orig(this.pointerMax)] > this.freqs[this.pointer]
this.norms[orig(this.pointerMax)] > this.freqs[this.pointerMax]
this.norms[orig(this.pointerMax)] > this.freqs[orig(this.doc)]
this.norms[orig(this.pointerMax)] > this.freqs[orig(this.pointer)]
this.norms[orig(this.pointerMax)] > this.freqs[orig(this.pointerMax)]
this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.pointer)])
this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.pointer)])
this.norms[orig(this.pointerMax)] > orig(this.docs[this.doc])
this.norms[orig(this.pointerMax)] > orig(this.freqs[this.doc])
this.norms[orig(this.pointerMax)] > orig(this.freqs[this.pointer])
this.docs[this.pointer] >= this.docs[orig(this.pointerMax)]
this.docs[this.pointerMax] <= this.docs[orig(this.pointer)]
this.docs[this.pointerMax] <= this.freqs[orig(this.pointer)]
this.docs[this.pointerMax] <= orig(this.freqs[this.pointer])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.docs[orig(this.doc)]
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.docs[orig(this.pointer)]
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.freqs[orig(this.doc)]
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= this.freqs[orig(this.pointer)]
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.docs[post(this.pointer)])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.freqs[post(this.pointer)])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.docs[this.doc])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.freqs[this.doc])
this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1] <= orig(this.freqs[this.pointer])
this.docs[orig(this.doc)] >= this.docs[orig(this.pointerMax)]
this.docs[orig(this.pointer)] >= this.docs[orig(this.pointerMax)]
this.docs[orig(this.pointer)] >= this.freqs[this.pointerMax]
this.docs[orig(this.pointerMax)] <= this.freqs[this.pointer]
this.docs[orig(this.pointerMax)] <= this.freqs[orig(this.doc)]
this.docs[orig(this.pointerMax)] <= this.freqs[orig(this.pointer)]
this.docs[orig(this.pointerMax)] <= this.freqs[orig(this.pointerMax)]
this.freqs[this.pointer] >= this.freqs[orig(this.pointerMax)]
this.freqs[this.pointerMax] <= this.freqs[orig(this.pointer)]
this.freqs[this.pointerMax] <= orig(this.freqs[this.pointer])
this.freqs[orig(this.doc)] >= this.freqs[orig(this.pointerMax)]
this.freqs[orig(this.pointer)] >= this.freqs[orig(this.pointerMax)]
orig(this.doc) - this.docs[orig(this.pointer)] + this.docs[orig(this.pointerMax)] == 0
this.docs[this.pointer] - this.docs[orig(this.pointerMax)] - orig(this.docs[post(this.pointer)]) == 0
this.docs[orig(this.doc)] - this.docs[orig(this.pointerMax)] - orig(this.docs[this.doc]) == 0
this.freqs[this.pointer] - this.freqs[orig(this.pointerMax)] - orig(this.freqs[post(this.pointer)]) == 0
this.freqs[orig(this.doc)] - this.freqs[orig(this.pointerMax)] - orig(this.freqs[this.doc]) == 0
this.freqs[orig(this.pointer)] - this.freqs[orig(this.pointerMax)] - orig(this.freqs[this.pointer]) == 0

==========
org.apache.lucene.search.TermScorer.next():::EXIT;condition="return == true"
this.doc == this.docs[this.pointer]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == orig(this.freqs[post(this.pointerMax)])
this.docs[this.pointerMax] == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.docs[this.pointerMax])
this.docs[this.pointerMax] == orig(this.freqs[this.pointerMax])
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
return == true
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.freqs[orig(this.pointer)] >= 1
orig(this.docs[post(this.pointerMax)-1]) one of 0, 23
orig(this.freqs[post(this.pointerMax)-1]) one of 0, 31, 40
this.norms[] elements > this.doc
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > orig(this.docs[post(this.doc)])
this.norms[] elements > orig(this.freqs[post(this.doc)])
this.norms[] elements > this.freqs[this.pointerMax-1]
this.norms[] elements > orig(this.docs[post(this.pointerMax)-1])
this.norms[] elements > orig(this.freqs[post(this.pointerMax)-1])
this.doc <= size(this.norms[])-1
this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc < this.norms[orig(this.doc)]
this.doc < this.norms[orig(this.pointer)]
this.doc < this.norms[orig(this.pointerMax)]
this.doc != this.freqs[this.doc]
this.doc < this.freqs[this.pointerMax-1]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.pointerMax-1]
orig(this.docs[post(this.doc)]) in this.docs[]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
orig(this.docs[post(this.pointerMax)-1]) in this.docs[]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.pointerMax-1]
orig(this.freqs[post(this.doc)]) in this.freqs[]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
orig(this.freqs[post(this.pointerMax)-1]) in this.freqs[]
this.pointer < this.pointerMax
this.pointer <= orig(this.pointerMax)
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.pointerMax-1]
this.pointer < this.freqs[this.pointerMax-1]
this.pointer <= orig(this.docs[post(this.pointer)])

this.pointer <= orig(this.docs[post(this.pointerMax)-1])
this.pointer <= orig(this.freqs[post(this.pointerMax)-1])
this.pointerMax > orig(this.pointer)
orig(this.pointerMax)
this.pointerMax >= orig(this.pointerMax)
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax != orig(this.freqs[post(this.doc)])
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax > this.docs[orig(this.pointerMax)]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax < this.freqs[this.pointerMax-1]
this.pointerMax != this.freqs[orig(this.doc)]
this.pointerMax != this.freqs[orig(this.pointer)]
this.pointerMax > this.freqs[orig(this.pointerMax)]
this.pointerMax != orig(this.freqs[post(this.pointer)])
this.pointerMax != orig(this.docs[post(this.pointerMax)-1])
this.pointerMax != orig(this.freqs[post(this.pointerMax)-1])
this.pointerMax != orig(this.freqs[this.doc])
this.pointerMax != orig(this.freqs[this.pointer])
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.doc)])
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != orig(this.freqs[post(this.pointerMax)-1])
orig(this.doc) < size(this.norms[])-1
orig(this.doc) < this.norms[this.doc]
orig(this.doc) < this.norms[this.pointerMax-1]
orig(this.doc) < this.freqs[this.pointerMax-1]
orig(this.doc) != this.freqs[orig(this.doc)]
orig(this.doc) <= orig(this.docs[post(this.pointer)])
orig(this.doc) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.doc) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.docs[]) elements < this.norms[this.doc]
orig(this.docs[]) elements < this.norms[this.pointerMax-1]
orig(this.docs[]) elements >= this.docs[this.pointerMax]
orig(this.docs[]) elements < this.freqs[this.pointerMax-1]
orig(this.docs[]) elements <= orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[]) elements <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.freqs[]) elements < this.norms[this.doc]
orig(this.freqs[]) elements < this.norms[this.pointerMax-1]
orig(this.freqs[]) elements >= this.docs[this.pointerMax]
orig(this.freqs[]) elements <= this.freqs[this.pointerMax-1]
orig(this.freqs[]) elements <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.pointer) < this.norms[this.doc]
orig(this.pointer) < this.norms[this.pointerMax-1]
orig(this.pointer) < this.freqs[this.pointerMax-1]
orig(this.pointer) != this.freqs[orig(this.pointer)]
orig(this.pointer) <= orig(this.docs[post(this.pointer)])
orig(this.pointer) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.pointer) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.pointerMax) < this.norms[this.doc]
orig(this.pointerMax) < this.norms[this.pointerMax-1]
orig(this.pointerMax) != this.freqs[this.doc]
orig(this.pointerMax) != this.freqs[this.pointer]
orig(this.pointerMax) < this.freqs[this.pointerMax-1]
orig(this.pointerMax) != this.freqs[orig(this.doc)]
orig(this.pointerMax) != this.freqs[orig(this.pointer)]
orig(this.pointerMax) != this.freqs[orig(this.pointerMax)]
orig(this.pointerMax) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.pointerMax) <= orig(this.freqs[post(this.pointerMax)-1])

size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) != orig(this.freqs[post(this.doc)])
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[]) != orig(this.freqs[post(this.pointerMax)-1])
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.norms[])-1 >= orig(this.docs[post(this.doc)])
size(this.norms[])-1 != orig(this.freqs[post(this.doc)])
size(this.norms[])-1 > this.docs[orig(this.pointer)]
size(this.norms[])-1 != orig(this.freqs[post(this.pointerMax)-1])
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 > orig(this.docs[post(this.doc)])
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
size(this.docs[])-1 > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > orig(this.docs[post(this.doc)])
this.norms[this.doc] > orig(this.freqs[post(this.doc)])
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.docs[orig(this.doc)]
this.norms[this.doc] > this.docs[orig(this.pointer)]
this.norms[this.doc] > this.docs[orig(this.pointerMax)]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.doc] > this.freqs[orig(this.doc)]
this.norms[this.doc] > this.freqs[orig(this.pointer)]
this.norms[this.doc] > this.freqs[orig(this.pointerMax)]
this.norms[this.doc] > orig(this.docs[post(this.pointer)])
this.norms[this.doc] > orig(this.freqs[post(this.pointer)])
this.norms[this.doc] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.doc] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.doc] > orig(this.docs[this.doc])
this.norms[this.doc] > orig(this.freqs[this.doc])
this.norms[this.doc] > orig(this.freqs[this.pointer])
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > orig(this.docs[post(this.doc)])
this.norms[this.pointer] > orig(this.freqs[post(this.doc)])
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.pointer] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > orig(this.docs[post(this.doc)])
this.norms[this.pointerMax] > orig(this.freqs[post(this.doc)])
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.pointerMax] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.pointerMax-1] != this.norms[orig(this.pointerMax)]
this.norms[this.pointerMax-1] > this.docs[this.doc]
this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > orig(this.docs[post(this.doc)])
this.norms[this.pointerMax-1] > orig(this.freqs[post(this.doc)])
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.docs[orig(this.doc)]
this.norms[this.pointerMax-1] > this.docs[orig(this.pointer)]
this.norms[this.pointerMax-1] > this.docs[orig(this.pointerMax)]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.freqs[orig(this.doc)]

this.norms[this.pointerMax-1] > this.freqs[orig(this.pointer)]
this.norms[this.pointerMax-1] > this.freqs[orig(this.pointerMax)]
this.norms[this.pointerMax-1] > orig(this.docs[post(this.pointer)])
this.norms[this.pointerMax-1] > orig(this.freqs[post(this.pointer)])
this.norms[this.pointerMax-1] > orig(this.docs[post(this.pointerMax)-1])
this.norms[this.pointerMax-1] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[this.pointerMax-1] > orig(this.docs[this.doc])
this.norms[this.pointerMax-1] > orig(this.freqs[this.doc])
this.norms[this.pointerMax-1] > orig(this.freqs[this.pointer])
this.norms[orig(this.doc)] > this.docs[this.doc]
this.norms[orig(this.doc)] > this.freqs[this.doc]
this.norms[orig(this.doc)] > orig(this.docs[post(this.doc)])
this.norms[orig(this.doc)] > orig(this.freqs[post(this.doc)])
this.norms[orig(this.doc)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.doc)] > orig(this.docs[post(this.pointerMax)-1])
this.norms[orig(this.doc)] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[orig(this.pointer)] > this.docs[this.doc]
this.norms[orig(this.pointer)] > this.freqs[this.doc]
this.norms[orig(this.pointer)] > orig(this.docs[post(this.doc)])
this.norms[orig(this.pointer)] > orig(this.freqs[post(this.doc)])
this.norms[orig(this.pointer)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.pointer)] > orig(this.docs[post(this.pointerMax)-1])
this.norms[orig(this.pointer)] > orig(this.freqs[post(this.pointerMax)-1])
this.norms[orig(this.pointerMax)] > this.docs[this.doc]
this.norms[orig(this.pointerMax)] > this.freqs[this.doc]
this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.doc)])
this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.doc)])
this.norms[orig(this.pointerMax)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.pointerMax)] > orig(this.docs[post(this.pointerMax)-1])
this.norms[orig(this.pointerMax)] > orig(this.freqs[post(this.pointerMax)-1])
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] >= this.docs[orig(this.pointerMax)]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] >= this.docs[orig(this.pointerMax)]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.freqs[orig(this.pointerMax)]
this.freqs[this.doc] != orig(this.docs[post(this.pointer)])
this.freqs[this.doc] != orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[post(this.doc)]) >= this.docs[this.pointerMax]
orig(this.docs[post(this.doc)]) != this.freqs[this.pointer]
orig(this.docs[post(this.doc)]) < this.freqs[this.pointerMax-1]
orig(this.docs[post(this.doc)]) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[post(this.doc)]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.freqs[post(this.doc)]) >= this.docs[this.pointerMax]
orig(this.freqs[post(this.doc)]) <= this.freqs[this.pointerMax-1]
orig(this.freqs[post(this.doc)]) <= orig(this.freqs[post(this.pointerMax)-1])
this.docs[this.pointerMax] <= this.docs[orig(this.doc)]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] <= this.freqs[orig(this.doc)]
this.docs[this.pointerMax] < this.freqs[orig(this.pointer)]
this.docs[this.pointerMax] <= orig(this.docs[post(this.pointer)])
this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointer)])
this.docs[this.pointerMax] <= orig(this.docs[post(this.pointerMax)-1])
this.docs[this.pointerMax] <= orig(this.freqs[post(this.pointerMax)-1])
this.docs[this.pointerMax] <= orig(this.docs[this.doc])
this.docs[this.pointerMax] <= orig(this.freqs[this.doc])
this.docs[orig(this.doc)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.doc)] != this.freqs[orig(this.pointer)]
this.docs[orig(this.pointer)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.pointer)] != this.freqs[orig(this.doc)]

this.docs[orig(this.pointerMax)] < this.freqs[this.pointer]
this.docs[orig(this.pointerMax)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.pointerMax)] < this.freqs[orig(this.pointer)]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
this.freqs[orig(this.pointerMax)]
orig(this.freqs[post(this.pointer)])
this.freqs[this.pointer] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[this.pointerMax-1] >= this.freqs[orig(this.doc)]
this.freqs[this.pointerMax-1] >= this.freqs[orig(this.pointer)]
this.freqs[this.pointerMax-1] > this.freqs[orig(this.pointerMax)]
this.freqs[this.pointerMax-1] > orig(this.docs[post(this.pointer)])
this.freqs[this.pointerMax-1] >= orig(this.freqs[post(this.pointer)])
this.freqs[this.pointerMax-1] > orig(this.docs[post(this.pointerMax)-1])
orig(this.freqs[post(this.pointerMax)-1])
this.freqs[this.pointerMax-1] > orig(this.docs[this.doc])
this.freqs[this.pointerMax-1] >= orig(this.freqs[this.doc])
this.freqs[this.pointerMax-1] >= orig(this.freqs[this.pointer])
this.freqs[orig(this.doc)] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[orig(this.pointerMax)]
this.freqs[orig(this.pointer)] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[orig(this.pointer)] != orig(this.docs[this.doc])
orig(this.freqs[this.pointer])
this.freqs[orig(this.pointerMax)] != orig(this.docs[post(this.pointer)])
this.freqs[orig(this.pointerMax)] != orig(this.freqs[post(this.pointer)])
this.freqs[orig(this.pointerMax)] != orig(this.docs[post(this.pointerMax)-1])
this.freqs[orig(this.pointerMax)] != orig(this.freqs[post(this.pointerMax)-1])
this.freqs[orig(this.pointerMax)] != orig(this.freqs[this.pointer])
orig(this.docs[post(this.pointer)]) <= orig(this.docs[post(this.pointerMax)-1])
orig(this.docs[post(this.pointer)]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.freqs[post(this.pointer)]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.docs[post(this.pointerMax)-1]) <= orig(this.freqs[post(this.pointerMax)-1])
orig(this.docs[post(this.pointerMax)-1]) >= orig(this.docs[this.doc])
orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.docs[this.doc])
orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.freqs[this.doc])
orig(this.freqs[post(this.pointerMax)-1]) >= orig(this.freqs[this.pointer])
this.doc - this.docs[orig(this.pointerMax)] - orig(this.docs[post(this.pointer)]) == 0
23 * this.pointer - 23 * orig(this.pointer) - orig(this.docs[post(this.pointerMax)-1]) == 0
2 * this.docs[this.doc] - 2 * orig(this.docs[post(this.doc)]) - 5 * this.docs[orig(this.pointerMax)] == 0
==========
org.apache.lucene.search.TermScorer.next():::EXIT;condition="not(return == true)"
this.docs[] == orig(this.docs[])
this.freqs[] == orig(this.freqs[])
this.pointerMax == this.docs[this.pointer]
this.pointerMax == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == this.docs[orig(this.doc)]
this.pointerMax == this.docs[orig(this.pointerMax)]
this.pointerMax == this.freqs[this.pointer]
this.pointerMax == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == this.freqs[orig(this.doc)]
this.pointerMax == this.freqs[orig(this.pointerMax)]
this.pointerMax == orig(this.docs[post(this.pointer)])
this.pointerMax == orig(this.freqs[post(this.pointer)])
this.pointerMax == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.docs[this.doc])
this.pointerMax == orig(this.freqs[this.doc])
this.pointerMax == orig(this.docs[this.pointerMax])
this.pointerMax == orig(this.freqs[this.pointerMax])
orig(this.doc) == this.docs[this.pointer-1]
orig(this.doc) == this.docs[orig(this.pointer)]
orig(this.doc) == orig(this.docs[post(this.pointer)-1])
this.norms[this.pointer-1] == this.norms[orig(this.pointer)]
this.norms[this.pointer-1] == orig(this.norms[this.pointer])

this.freqs[this.pointer-1] == this.freqs[orig(this.pointer)]
this.freqs[this.pointer-1] == orig(this.freqs[this.pointer])
this.doc == 2147483647
this.pointer one of 1, 18, 19
this.pointerMax == 0
return == false
orig(this.doc) one of 0, 23
orig(this.pointer) one of 0, 17, 18
this.norms[this.pointer] one of 100, 101, 106
this.norms[this.pointer-1] one of 101, 105, 106
this.norms[this.pointerMax] == 105
this.norms[orig(this.doc)] one of 101, 105
this.freqs[this.pointer-1] one of 0, 31, 40
this.norms[] elements < this.doc
this.norms[] elements > this.freqs[this.pointer-1]
this.docs[] elements < this.doc
this.freqs[] elements < this.doc
this.doc > this.pointer
this.doc > orig(this.doc)
this.doc > orig(this.pointer)
this.doc > orig(this.pointerMax)
this.doc > this.norms[this.pointer]
this.doc > this.norms[this.pointer-1]
this.doc > this.norms[orig(this.doc)]
this.doc > this.norms[orig(this.pointerMax)]
this.doc > this.docs[this.pointerMax]
this.doc > this.freqs[this.pointer-1]
this.doc > this.freqs[this.pointerMax]
this.docs[] elements >= this.pointerMax
this.docs[] elements <= orig(this.doc)
orig(this.pointer) in this.docs[]
this.docs[] elements < this.norms[this.pointer-1]
this.docs[] elements <= this.freqs[this.pointer-1]
this.freqs[] elements >= this.pointerMax
this.freqs[] elements < this.norms[this.pointer-1]
this.freqs[] elements <= this.freqs[this.pointer-1]
this.pointer > this.pointerMax
this.pointer != orig(this.doc)
this.pointer > orig(this.pointer)
orig(this.pointerMax)
this.pointer >= orig(this.pointerMax)
this.pointer < this.norms[this.pointer-1]
this.pointer > this.docs[this.pointerMax]
this.pointer != this.freqs[this.pointer-1]
this.pointer > this.freqs[this.pointerMax]
this.pointerMax <= orig(this.doc)
this.pointerMax <= orig(this.pointer)
this.pointerMax <= orig(this.pointerMax)
this.pointerMax < this.norms[this.pointer-1]
this.pointerMax <= this.docs[this.pointerMax]
this.pointerMax <= this.freqs[this.pointer-1]
this.pointerMax <= this.freqs[this.pointerMax]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointer-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointer-1]
orig(this.doc) >= orig(this.pointerMax)
orig(this.doc) < this.norms[this.pointer-1]
orig(this.doc) <= this.freqs[this.pointer-1]
orig(this.pointer) < this.norms[this.pointer-1]
orig(this.pointer) <= this.freqs[this.pointer-1]
orig(this.pointerMax) < this.norms[this.pointer-1]
orig(this.pointerMax) <= this.freqs[this.pointer-1]
size(this.norms[]) < this.norms[this.pointer-1]
size(this.norms[]) != this.freqs[this.pointer-1]

size(this.norms[])-1 != this.freqs[this.pointer-1]
this.norms[this.pointer] != this.norms[this.pointer-1]
this.norms[this.pointer] != this.norms[this.pointerMax]
this.norms[this.pointer] <= this.norms[orig(this.pointerMax)]
this.norms[this.pointer] > this.freqs[this.pointer-1]
this.norms[this.pointer-1] >= this.norms[orig(this.doc)]
this.norms[this.pointer-1] > this.docs[this.pointerMax]
this.norms[this.pointer-1] > this.freqs[this.pointer-1]
this.norms[this.pointer-1] > this.freqs[this.pointerMax]
this.norms[this.pointerMax] >= this.norms[orig(this.doc)]
this.norms[this.pointerMax] > this.freqs[this.pointer-1]
this.norms[orig(this.doc)] <= this.norms[orig(this.pointerMax)]
this.norms[orig(this.doc)] > this.freqs[this.pointer-1]
this.norms[orig(this.pointerMax)] > this.freqs[this.pointer-1]
this.docs[this.pointerMax] <= this.freqs[this.pointer-1]
this.freqs[this.pointer-1] >= this.freqs[this.pointerMax]
==========
org.apache.lucene.search.TermScorer.score():::ENTER
this.doc == this.docs[this.pointer]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.norms[] elements > this.doc
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > this.freqs[this.pointerMax-1]
this.doc <= size(this.norms[])-1
this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc != this.freqs[this.doc]
this.doc < this.freqs[this.pointerMax-1]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.pointerMax-1]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.pointerMax-1]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
this.pointer < this.pointerMax
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.pointerMax-1]
this.pointer != this.freqs[this.pointer]
this.pointer < this.freqs[this.pointerMax-1]
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax != this.freqs[this.pointer]

this.pointerMax < this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.docs[this.doc]
this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
==========
org.apache.lucene.search.TermScorer.score():::EXIT
this.weight == orig(this.weight)
this.termDocs == orig(this.termDocs)
this.norms == orig(this.norms)
this.norms[] == orig(this.norms[])
this.weightValue == orig(this.weightValue)
this.doc == orig(this.doc)
this.doc == this.docs[this.pointer]
this.doc == this.docs[orig(this.pointer)]
this.doc == orig(this.docs[post(this.pointer)])
this.doc == orig(this.docs[this.pointer])
this.docs == orig(this.docs)
this.docs[] == orig(this.docs[])
this.freqs == orig(this.freqs)
this.freqs[] == orig(this.freqs[])
this.pointer == orig(this.pointer)
this.pointerMax == orig(this.pointerMax)
this.scoreCache == orig(this.scoreCache)
this.scoreCache[] == orig(this.scoreCache[])
size(this.norms[])-1 == this.docs[this.pointerMax-1]
size(this.norms[])-1 == this.docs[orig(this.pointerMax)-1]
size(this.norms[])-1 == orig(this.docs[post(this.pointerMax)-1])
size(this.norms[])-1 == orig(this.docs[this.pointerMax-1])
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]

this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[orig(this.pointerMax)]
this.docs[this.pointerMax] == orig(this.freqs[post(this.pointerMax)])
this.docs[this.pointerMax] == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[this.pointerMax])
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.weight.getClass() == orig(this.weight.getClass())
this.termDocs.getClass() == orig(this.termDocs.getClass())
this.norms[] elements > this.doc
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > this.freqs[this.pointerMax-1]
this.weightValue > return
this.doc <= size(this.norms[])-1
this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc != this.freqs[this.doc]
this.doc < this.freqs[this.pointerMax-1]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.pointerMax-1]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.pointerMax-1]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
this.pointer < this.pointerMax
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.pointerMax-1]
this.pointer != this.freqs[this.pointer]
this.pointer < this.freqs[this.pointerMax-1]
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax < this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]

this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.docs[this.doc]
this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector):::ENTER
this.doc == this.pointer
this.doc == this.pointerMax
this.doc == this.docs[this.doc]
this.doc == this.freqs[this.doc]
this.doc == this.docs[this.pointer]
this.doc == this.docs[this.pointerMax]
this.doc == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.doc == this.freqs[this.pointer]
this.doc == this.freqs[this.pointerMax]
this.doc == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[] == this.freqs[]
this.weightValue == 1.1823215
this.doc == 0
this.docs[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.docs[] elements == 0
this.scoreCache[] == [0.0, 1.1823215, 1.6720551, 2.047841, 2.364643, 2.6437514, 2.8960845, 3.1281288, 3.3441103, 3.5469646,
3.7388291, 3.921317, 4.095682, 4.262921, 4.4238424, 4.5791116, 4.729286, 4.8748364, 5.0161653, 5.1536202, 5.287503, 5.418078,
5.54558, 5.670215, 5.792169, 5.9116077, 6.028681, 6.1435227, 6.2562575, 6.3669963, 6.475842, 6.5828876]
hc.getClass() == org.apache.lucene.search.TopDocCollector.class
this.norms[this.doc] == 105
this.docs[] elements == this.doc
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector):::EXIT
this.weight == orig(this.weight)
this.termDocs == orig(this.termDocs)
this.norms == orig(this.norms)
this.norms[] == orig(this.norms[])
this.weightValue == orig(this.weightValue)
this.docs == orig(this.docs)
this.freqs == orig(this.freqs)
this.pointerMax == orig(this.doc)
this.pointerMax == orig(this.pointer)
this.pointerMax == orig(this.pointerMax)
this.pointerMax == this.docs[this.pointer]
this.pointerMax == this.docs[this.pointerMax]
this.pointerMax == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]

this.pointerMax == this.docs[orig(this.doc)]
this.pointerMax == this.docs[orig(this.pointer)]
this.pointerMax == this.docs[orig(this.pointerMax)]
this.pointerMax == this.freqs[this.pointer]
this.pointerMax == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == orig(this.docs[post(this.pointer)])
this.pointerMax == orig(this.docs[post(this.pointer)-1])
this.pointerMax == orig(this.freqs[post(this.pointer)])
this.pointerMax == orig(this.freqs[post(this.pointer)-1])
this.pointerMax == orig(this.docs[post(this.pointerMax)])
this.pointerMax == orig(this.freqs[post(this.pointerMax)])
this.pointerMax == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.docs[this.doc])
this.pointerMax == orig(this.freqs[this.doc])
this.pointerMax == orig(this.docs[this.pointer])
this.pointerMax == orig(this.docs[this.pointerMax])
this.pointerMax == orig(this.freqs[this.pointer])
this.pointerMax == orig(this.freqs[this.pointerMax])
this.scoreCache == orig(this.scoreCache)
this.scoreCache[] == orig(this.scoreCache[])
size(this.norms[])-1 == this.docs[this.pointer-1]
this.weightValue == 1.1823215
this.doc == 2147483647
this.docs[] == [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] == [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointer == 19
this.pointerMax == 0
this.scoreCache[] == [0.0, 1.1823215, 1.6720551, 2.047841, 2.364643, 2.6437514, 2.8960845, 3.1281288, 3.3441103, 3.5469646,
3.7388291, 3.921317, 4.095682, 4.262921, 4.4238424, 4.5791116, 4.729286, 4.8748364, 5.0161653, 5.1536202, 5.287503, 5.418078,
5.54558, 5.670215, 5.792169, 5.9116077, 6.028681, 6.1435227, 6.2562575, 6.3669963, 6.475842, 6.5828876]
this.norms[this.pointer] == 106
this.norms[this.pointer-1] == 101
this.norms[this.pointerMax] == 105
this.freqs[this.pointer-1] == 40
this.freqs[this.pointerMax] == 7
this.weight.getClass() == orig(this.weight.getClass())
this.termDocs.getClass() == orig(this.termDocs.getClass())
this.pointer in this.docs[]
orig(this.docs[]) elements == this.pointerMax
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector, int):::ENTER
this.doc == this.pointer
this.doc == this.docs[this.doc]
this.doc == this.docs[this.pointer]
this.doc == this.docs[this.pointerMax]
this.doc == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.doc == this.freqs[this.pointerMax]
this.doc == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
this.weightValue == 1.1823215
this.doc == 0
this.docs[] == [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] == [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax == 19
this.scoreCache[] == [0.0, 1.1823215, 1.6720551, 2.047841, 2.364643, 2.6437514, 2.8960845, 3.1281288, 3.3441103, 3.5469646,
3.7388291, 3.921317, 4.095682, 4.262921, 4.4238424, 4.5791116, 4.729286, 4.8748364, 5.0161653, 5.1536202, 5.287503, 5.418078,
5.54558, 5.670215, 5.792169, 5.9116077, 6.028681, 6.1435227, 6.2562575, 6.3669963, 6.475842, 6.5828876]
c.getClass() == org.apache.lucene.search.TopDocCollector.class
end == 2147483647
this.norms[this.doc] == 105
this.norms[this.pointerMax] == 106
this.norms[this.pointerMax-1] == 101

this.freqs[this.doc] == 7
this.freqs[this.pointerMax-1] == 40
this.freqs[this.doc] in this.docs[]
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector, int):::EXIT85
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector, int):::EXIT85;condition="not(return == true)"
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector, int):::EXIT
this.weight == orig(this.weight)
this.termDocs == orig(this.termDocs)
this.norms == orig(this.norms)
this.norms[] == orig(this.norms[])
this.weightValue == orig(this.weightValue)
this.doc == orig(end)
this.docs == orig(this.docs)
this.docs[] == orig(this.docs[])
this.freqs == orig(this.freqs)
this.freqs[] == orig(this.freqs[])
this.pointer == orig(this.pointerMax)
this.pointerMax == orig(this.doc)
this.pointerMax == orig(this.pointer)
this.pointerMax == this.docs[this.pointer]
this.pointerMax == this.docs[this.pointerMax]
this.pointerMax == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == this.docs[orig(this.doc)]
this.pointerMax == this.docs[orig(this.pointer)]
this.pointerMax == this.docs[orig(this.pointerMax)]
this.pointerMax == this.freqs[this.pointer]
this.pointerMax == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.pointerMax == this.freqs[orig(this.pointerMax)]
this.pointerMax == orig(this.docs[post(this.pointer)])
this.pointerMax == orig(this.freqs[post(this.pointer)])
this.pointerMax == orig(this.docs[post(this.pointerMax)])
this.pointerMax == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.pointerMax == orig(this.docs[this.doc])
this.pointerMax == orig(this.docs[this.pointer])
this.pointerMax == orig(this.docs[this.pointerMax])
this.pointerMax == orig(this.freqs[this.pointerMax])
this.scoreCache == orig(this.scoreCache)
this.scoreCache[] == orig(this.scoreCache[])
size(this.norms[])-1 == this.docs[this.pointer-1]
size(this.norms[])-1 == this.docs[orig(this.pointerMax)-1]
size(this.norms[])-1 == orig(this.docs[post(this.pointer)-1])
size(this.norms[])-1 == orig(this.docs[this.pointerMax-1])
this.weightValue == 1.1823215
this.doc == 2147483647
this.docs[] == [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] == [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointer == 19
this.pointerMax == 0
this.scoreCache[] == [0.0, 1.1823215, 1.6720551, 2.047841, 2.364643, 2.6437514, 2.8960845, 3.1281288, 3.3441103, 3.5469646, 3.7388291, 3.921317, 4.095682, 4.262921, 4.4238424, 4.5791116, 4.729286, 4.8748364, 5.0161653, 5.1536202, 5.287503, 5.418078, 5.54558, 5.670215, 5.792169, 5.9116077, 6.028681, 6.1435227, 6.2562575, 6.3669963, 6.475842, 6.5828876]
return == false
this.norms[this.pointer] == 106
this.norms[this.pointer-1] == 101
this.norms[this.pointerMax] == 105
this.freqs[this.pointer-1] == 40
this.freqs[this.pointerMax] == 7
this.weight.getClass() == orig(this.weight.getClass())

this.termDocs.getClass() == orig(this.termDocs.getClass())
this.pointer in this.docs[]
==========
org.apache.lucene.search.TermScorer.score(org.apache.lucene.search.HitCollector, int):::EXIT;condition="not(return == true)"
==========
org.apache.lucene.search.TermScorer.skipTo(int):::ENTER
this.doc == this.docs[this.pointer]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.norms[] elements > this.doc
this.norms[] elements > target
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > this.docs[target]
this.norms[] elements > this.docs[target-1]
this.norms[] elements > this.freqs[this.pointerMax-1]
this.norms[] elements > this.freqs[target]
this.norms[] elements > this.freqs[target-1]
this.doc < target
this.doc < size(this.norms[])-1
this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc < this.norms[target]
this.doc < this.norms[target-1]
this.doc != this.freqs[this.doc]
this.doc != this.docs[target]
this.doc < this.freqs[this.pointerMax-1]
this.doc != this.freqs[target-1]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.pointerMax-1]
this.docs[] elements < this.norms[target]
this.docs[] elements < this.norms[target-1]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.pointerMax-1]
this.freqs[] elements < this.norms[target]
this.freqs[] elements < this.norms[target-1]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
this.pointer < this.pointerMax
this.pointer < target
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.pointerMax-1]
this.pointer < this.norms[target]
this.pointer < this.norms[target-1]
this.pointer != this.docs[target]
this.pointer != this.freqs[this.pointer]

this.pointer < this.freqs[this.pointerMax-1]
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax < this.norms[target]
this.pointerMax < this.norms[target-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax < this.freqs[this.pointerMax-1]
this.pointerMax != this.freqs[target]
this.pointerMax != this.freqs[target-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[target]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[target-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[target]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[target-1]
target <= size(this.norms[])-1
target < size(this.docs[])-1
target < this.norms[this.doc]
target < this.norms[this.pointer]
target < this.norms[this.pointerMax]
target < this.norms[this.pointerMax-1]
target < this.norms[target]
target < this.norms[target-1]
target > this.docs[this.pointerMax]
target < this.freqs[this.pointerMax-1]
target != this.freqs[target]
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) < this.norms[target]
size(this.norms[]) < this.norms[target-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[]) != this.freqs[target]
size(this.norms[]) != this.freqs[target-1]
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.norms[])-1 >= this.docs[target]
size(this.norms[])-1 >= this.docs[target-1]
size(this.norms[])-1 != this.freqs[target]
size(this.norms[])-1 != this.freqs[target-1]
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 > this.docs[target]
size(this.docs[])-1 > this.docs[target-1]
size(this.docs[])-1 >= this.freqs[this.pointer]
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
size(this.docs[])-1 != this.freqs[target]
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.freqs[this.doc]
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.docs[target]
this.norms[this.doc] > this.docs[target-1]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.doc] > this.freqs[target]
this.norms[this.doc] > this.freqs[target-1]
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > this.docs[target]
this.norms[this.pointer] > this.docs[target-1]

this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.freqs[target]
this.norms[this.pointer] > this.freqs[target-1]
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > this.docs[target]
this.norms[this.pointerMax] > this.docs[target-1]
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] > this.freqs[target]
this.norms[this.pointerMax] > this.freqs[target-1]
this.norms[this.pointerMax-1] > this.docs[this.doc]
this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.docs[target]
this.norms[this.pointerMax-1] > this.docs[target-1]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.freqs[target]
this.norms[this.pointerMax-1] > this.freqs[target-1]
this.norms[target] != this.norms[target-1]
this.norms[target] > this.docs[this.doc]
this.norms[target] > this.freqs[this.doc]
this.norms[target] > this.docs[this.pointerMax]
this.norms[target] > this.docs[target]
this.norms[target] > this.docs[target-1]
this.norms[target] > this.freqs[this.pointer]
this.norms[target] > this.freqs[this.pointerMax-1]
this.norms[target] > this.freqs[target]
this.norms[target] > this.freqs[target-1]
this.norms[target-1] > this.docs[this.doc]
this.norms[target-1] > this.freqs[this.doc]
this.norms[target-1] > this.docs[this.pointerMax]
this.norms[target-1] > this.docs[target]
this.norms[target-1] > this.docs[target-1]
this.norms[target-1] > this.freqs[this.pointer]
this.norms[target-1] > this.freqs[this.pointerMax-1]
this.norms[target-1] > this.freqs[target]
this.norms[target-1] > this.freqs[target-1]
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] <= this.docs[target-1]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] <= this.docs[target]
this.docs[this.pointerMax] <= this.docs[target-1]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] <= this.freqs[target]
this.docs[this.pointerMax] <= this.freqs[target-1]
this.docs[target] < this.freqs[this.pointerMax-1]
this.docs[target-1] != this.freqs[this.pointer]
this.docs[target-1] < this.freqs[this.pointerMax-1]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
this.freqs[this.pointerMax-1] >= this.freqs[target]
this.freqs[this.pointerMax-1] >= this.freqs[target-1]
==========
org.apache.lucene.search.TermScorer.skipTo(int):::EXIT140
==========
org.apache.lucene.search.TermScorer.skipTo(int):::EXIT140;condition="return == true"
==========
org.apache.lucene.search.TermScorer.skipTo(int):::EXIT

this.weight == orig(this.weight)
this.termDocs == orig(this.termDocs)
this.norms == orig(this.norms)
this.norms[] == orig(this.norms[])
this.weightValue == orig(this.weightValue)
this.doc == this.docs[this.pointer]
this.doc == orig(this.docs[post(this.pointer)])
this.docs == orig(this.docs)
this.docs[] == orig(this.docs[])
this.freqs == orig(this.freqs)
this.freqs[] == orig(this.freqs[])
this.pointerMax == orig(this.pointerMax)
this.scoreCache == orig(this.scoreCache)
this.scoreCache[] == orig(this.scoreCache[])
orig(this.doc) == this.docs[orig(this.pointer)]
size(this.norms[])-1 == this.docs[this.pointerMax-1]
size(this.norms[])-1 == this.docs[orig(this.pointerMax)-1]
size(this.norms[])-1 == orig(this.docs[post(this.pointerMax)-1])
size(this.norms[])-1 == orig(this.docs[this.pointerMax-1])
this.docs[this.pointerMax] == this.docs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[this.pointerMax]
this.docs[this.pointerMax] == this.freqs[org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE-1]
this.docs[this.pointerMax] == this.freqs[orig(this.pointerMax)]
this.docs[this.pointerMax] == orig(this.freqs[post(this.pointerMax)])
this.docs[this.pointerMax] == orig(this.docs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[post(org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE)-1])
this.docs[this.pointerMax] == orig(this.freqs[this.pointerMax])
this.docs[] one of [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 15, 16, 18, 19, 20, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [2, 4, 5, 7, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, 22, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.freqs[] one of [7, 20, 25, 15, 25, 4, 13, 13, 11, 30, 9, 4, 15, 2, 5, 26, 4, 40, 40, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [14, 19, 16, 15, 9, 1, 3, 3, 1, 31, 11, 5, 1, 8, 5, 1, 6, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
this.pointerMax one of 18, 19
return == true
this.norms[this.pointerMax] one of 101, 106
this.norms[this.pointerMax-1] one of 101, 106
this.docs[this.pointerMax] == 0
this.freqs[this.pointer] >= 1
this.freqs[this.pointer-1] >= 1
this.freqs[this.pointerMax-1] one of 31, 40
this.freqs[orig(this.pointer)] >= 1
this.weight.getClass() == orig(this.weight.getClass())
this.termDocs.getClass() == orig(this.termDocs.getClass())
this.norms[] elements > this.doc
this.norms[] elements > orig(this.doc)
this.norms[] elements > orig(this.pointer)
this.norms[] elements > orig(target)
this.norms[] elements > this.docs[this.doc]
this.norms[] elements > this.docs[this.doc-1]
this.norms[] elements > this.freqs[this.doc]
this.norms[] elements > this.freqs[this.doc-1]
this.norms[] elements > this.docs[this.pointer-1]
this.norms[] elements > this.docs[orig(this.doc)]
this.norms[] elements > this.docs[orig(target)]
this.norms[] elements > this.docs[orig(target)-1]
this.norms[] elements > this.freqs[this.pointer-1]
this.norms[] elements > this.freqs[this.pointerMax-1]
this.norms[] elements > this.freqs[orig(this.doc)]
this.norms[] elements > this.freqs[orig(this.pointer)]
this.norms[] elements > this.freqs[orig(target)]
this.norms[] elements > this.freqs[orig(target)-1]
this.doc > orig(this.doc)
this.doc > orig(this.pointer)
this.doc >= orig(target)

this.doc <= size(this.norms[])-1
this.doc < size(this.docs[])-1
this.doc < this.norms[this.doc]
this.doc < this.norms[this.doc-1]
this.doc < this.norms[this.pointer]
this.doc < this.norms[this.pointer-1]
this.doc < this.norms[this.pointerMax]
this.doc < this.norms[this.pointerMax-1]
this.doc < this.norms[orig(this.doc)]
this.doc < this.norms[orig(this.pointer)]
this.doc < this.norms[orig(target)]
this.doc < this.norms[orig(target)-1]
this.doc != this.freqs[this.doc]
this.doc > this.docs[this.pointer-1]
this.doc > this.docs[this.pointerMax]
this.doc != this.freqs[this.pointer]
this.doc < this.freqs[this.pointerMax-1]
this.doc != this.freqs[orig(target)]
this.docs[] elements < this.norms[this.doc]
this.docs[] elements < this.norms[this.doc-1]
this.docs[] elements < this.norms[this.pointer-1]
this.docs[] elements < this.norms[this.pointerMax-1]
this.docs[] elements < this.norms[orig(this.doc)]
this.docs[] elements < this.norms[orig(this.pointer)]
this.docs[] elements < this.norms[orig(target)]
this.docs[] elements < this.norms[orig(target)-1]
this.docs[] elements >= this.docs[this.pointerMax]
this.docs[] elements < this.freqs[this.pointerMax-1]
this.freqs[] elements < this.norms[this.doc]
this.freqs[] elements < this.norms[this.doc-1]
this.freqs[] elements < this.norms[this.pointer-1]
this.freqs[] elements < this.norms[this.pointerMax-1]
this.freqs[] elements < this.norms[orig(this.doc)]
this.freqs[] elements < this.norms[orig(this.pointer)]
this.freqs[] elements < this.norms[orig(target)]
this.freqs[] elements < this.norms[orig(target)-1]
this.freqs[] elements >= this.docs[this.pointerMax]
this.freqs[] elements <= this.freqs[this.pointerMax-1]
this.pointer < this.pointerMax
this.pointer > orig(this.pointer)
this.pointer <= orig(target)
this.pointer < this.norms[this.doc]
this.pointer < this.norms[this.doc-1]
this.pointer < this.norms[this.pointer-1]
this.pointer < this.norms[this.pointerMax-1]
this.pointer < this.norms[orig(this.doc)]
this.pointer < this.norms[orig(this.pointer)]
this.pointer < this.norms[orig(target)]
this.pointer < this.norms[orig(target)-1]
this.pointer != this.docs[this.doc-1]
this.pointer > this.docs[this.pointerMax]
this.pointer != this.docs[orig(this.doc)]
this.pointer != this.docs[orig(target)-1]
this.pointer != this.freqs[this.pointer]
this.pointer < this.freqs[this.pointerMax-1]
this.pointer != this.freqs[orig(this.doc)]
this.pointerMax > orig(this.pointer)
this.pointerMax < this.norms[this.doc]
this.pointerMax < this.norms[this.doc-1]
this.pointerMax < this.norms[this.pointer-1]
this.pointerMax < this.norms[this.pointerMax-1]
this.pointerMax < this.norms[orig(this.doc)]
this.pointerMax < this.norms[orig(this.pointer)]

this.pointerMax < this.norms[orig(target)]
this.pointerMax < this.norms[orig(target)-1]
this.pointerMax != this.freqs[this.doc]
this.pointerMax != this.freqs[this.doc-1]
this.pointerMax > this.docs[this.pointerMax]
this.pointerMax != this.freqs[this.pointer]
this.pointerMax != this.freqs[this.pointer-1]
this.pointerMax < this.freqs[this.pointerMax-1]
this.pointerMax != this.freqs[orig(this.doc)]
this.pointerMax != this.freqs[orig(this.pointer)]
this.pointerMax != this.freqs[orig(target)]
this.pointerMax != this.freqs[orig(target)-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.doc-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointer-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(this.doc)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(this.pointer)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(target)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE < this.norms[orig(target)-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.doc-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointer-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[this.pointerMax-1]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[orig(this.doc)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[orig(target)]
org.apache.lucene.search.TermScorer.SCORE_CACHE_SIZE != this.freqs[orig(target)-1]
orig(this.doc) < size(this.norms[])-1
orig(this.doc) < size(this.docs[])-1
orig(this.doc) < this.norms[this.doc]
orig(this.doc) < this.norms[this.doc-1]
orig(this.doc) < this.norms[this.pointer]
orig(this.doc) < this.norms[this.pointer-1]
orig(this.doc) < this.norms[this.pointerMax]
orig(this.doc) < this.norms[this.pointerMax-1]
orig(this.doc) < this.norms[orig(this.doc)]
orig(this.doc) < this.norms[orig(this.pointer)]
orig(this.doc) < this.norms[orig(target)]
orig(this.doc) < this.norms[orig(target)-1]
orig(this.doc) != this.docs[this.doc]
orig(this.doc) != this.freqs[this.doc-1]
orig(this.doc) <= this.docs[this.pointer-1]
orig(this.doc) >= this.docs[this.pointerMax]
orig(this.doc) != this.docs[orig(target)]
orig(this.doc) < this.freqs[this.pointerMax-1]
orig(this.doc) != this.freqs[orig(this.doc)]
orig(this.doc) != this.freqs[orig(target)-1]
orig(this.pointer) < size(this.norms[])-1
orig(this.pointer) < size(this.docs[])-1
orig(this.pointer) < this.norms[this.doc]
orig(this.pointer) < this.norms[this.doc-1]
orig(this.pointer) < this.norms[this.pointer]
orig(this.pointer) < this.norms[this.pointer-1]
orig(this.pointer) < this.norms[this.pointerMax]
orig(this.pointer) < this.norms[this.pointerMax-1]
orig(this.pointer) < this.norms[orig(this.doc)]
orig(this.pointer) < this.norms[orig(this.pointer)]
orig(this.pointer) < this.norms[orig(target)]
orig(this.pointer) < this.norms[orig(target)-1]
orig(this.pointer) != this.docs[this.doc]
orig(this.pointer) <= this.docs[this.pointer-1]
orig(this.pointer) >= this.docs[this.pointerMax]
orig(this.pointer) != this.docs[orig(target)]

orig(this.pointer) != this.freqs[this.pointer-1]
orig(this.pointer) < this.freqs[this.pointerMax-1]
orig(this.pointer) != this.freqs[orig(this.pointer)]
orig(target) <= size(this.norms[])-1
orig(target) < size(this.docs[])-1
orig(target) < this.norms[this.doc]
orig(target) < this.norms[this.doc-1]
orig(target) < this.norms[this.pointer]
orig(target) < this.norms[this.pointer-1]
orig(target) < this.norms[this.pointerMax]
orig(target) < this.norms[this.pointerMax-1]
orig(target) < this.norms[orig(this.doc)]
orig(target) < this.norms[orig(this.pointer)]
orig(target) < this.norms[orig(target)]
orig(target) < this.norms[orig(target)-1]
orig(target) > this.docs[this.pointer-1]
orig(target) > this.docs[this.pointerMax]
orig(target) != this.freqs[this.pointer]
orig(target) < this.freqs[this.pointerMax-1]
orig(target) != this.freqs[orig(target)]
size(this.norms[]) < this.norms[this.doc]
size(this.norms[]) < this.norms[this.doc-1]
size(this.norms[]) < this.norms[this.pointer-1]
size(this.norms[]) < this.norms[this.pointerMax-1]
size(this.norms[]) < this.norms[orig(this.doc)]
size(this.norms[]) < this.norms[orig(this.pointer)]
size(this.norms[]) < this.norms[orig(target)]
size(this.norms[]) < this.norms[orig(target)-1]
size(this.norms[]) != this.freqs[this.doc]
size(this.norms[]) != this.freqs[this.doc-1]
size(this.norms[]) != this.freqs[this.pointer-1]
size(this.norms[]) < this.freqs[this.pointerMax-1]
size(this.norms[]) != this.freqs[orig(this.doc)]
size(this.norms[]) != this.freqs[orig(this.pointer)]
size(this.norms[]) != this.freqs[orig(target)]
size(this.norms[]) != this.freqs[orig(target)-1]
size(this.norms[])-1 >= this.docs[this.doc]
size(this.norms[])-1 >= this.docs[this.doc-1]
size(this.norms[])-1 != this.freqs[this.doc]
size(this.norms[])-1 != this.freqs[this.doc-1]
size(this.norms[])-1 > this.docs[this.pointer-1]
size(this.norms[])-1 >= this.docs[orig(this.doc)]
size(this.norms[])-1 >= this.docs[orig(target)]
size(this.norms[])-1 >= this.docs[orig(target)-1]
size(this.norms[])-1 != this.freqs[this.pointer-1]
size(this.norms[])-1 != this.freqs[orig(this.doc)]
size(this.norms[])-1 != this.freqs[orig(this.pointer)]
size(this.norms[])-1 != this.freqs[orig(target)]
size(this.norms[])-1 != this.freqs[orig(target)-1]
size(this.docs[])-1 > this.docs[this.doc]
size(this.docs[])-1 > this.docs[this.doc-1]
size(this.docs[])-1 != this.freqs[this.doc]
size(this.docs[])-1 > this.docs[this.pointer-1]
size(this.docs[])-1 > this.docs[orig(this.doc)]
size(this.docs[])-1 > this.docs[orig(target)]
size(this.docs[])-1 > this.docs[orig(target)-1]
size(this.docs[])-1 <= this.freqs[this.pointerMax-1]
size(this.docs[])-1 >= this.freqs[orig(this.pointer)]
size(this.docs[])-1 != this.freqs[orig(target)]
this.norms[this.doc] != this.norms[this.doc-1]
this.norms[this.doc] > this.docs[this.doc]
this.norms[this.doc] > this.docs[this.doc-1]
this.norms[this.doc] > this.freqs[this.doc]

this.norms[this.doc] > this.freqs[this.doc-1]
this.norms[this.doc] > this.docs[this.pointer-1]
this.norms[this.doc] > this.docs[this.pointerMax]
this.norms[this.doc] > this.docs[orig(this.doc)]
this.norms[this.doc] > this.docs[orig(target)]
this.norms[this.doc] > this.docs[orig(target)-1]
this.norms[this.doc] > this.freqs[this.pointer]
this.norms[this.doc] > this.freqs[this.pointer-1]
this.norms[this.doc] > this.freqs[this.pointerMax-1]
this.norms[this.doc] > this.freqs[orig(this.doc)]
this.norms[this.doc] > this.freqs[orig(this.pointer)]
this.norms[this.doc] > this.freqs[orig(target)]
this.norms[this.doc] > this.freqs[orig(target)-1]
this.norms[this.doc-1] > this.docs[this.doc]
this.norms[this.doc-1] > this.docs[this.doc-1]
this.norms[this.doc-1] > this.freqs[this.doc]
this.norms[this.doc-1] > this.freqs[this.doc-1]
this.norms[this.doc-1] > this.docs[this.pointer-1]
this.norms[this.doc-1] > this.docs[this.pointerMax]
this.norms[this.doc-1] > this.docs[orig(this.doc)]
this.norms[this.doc-1] > this.docs[orig(target)]
this.norms[this.doc-1] > this.docs[orig(target)-1]
this.norms[this.doc-1] > this.freqs[this.pointer]
this.norms[this.doc-1] > this.freqs[this.pointer-1]
this.norms[this.doc-1] > this.freqs[this.pointerMax-1]
this.norms[this.doc-1] > this.freqs[orig(this.doc)]
this.norms[this.doc-1] > this.freqs[orig(this.pointer)]
this.norms[this.doc-1] > this.freqs[orig(target)]
this.norms[this.doc-1] > this.freqs[orig(target)-1]
this.norms[this.pointer] != this.norms[this.pointer-1]
this.norms[this.pointer] > this.docs[this.doc]
this.norms[this.pointer] > this.docs[this.doc-1]
this.norms[this.pointer] > this.freqs[this.doc]
this.norms[this.pointer] > this.freqs[this.doc-1]
this.norms[this.pointer] > this.docs[this.pointer-1]
this.norms[this.pointer] > this.docs[orig(this.doc)]
this.norms[this.pointer] > this.docs[orig(target)]
this.norms[this.pointer] > this.docs[orig(target)-1]
this.norms[this.pointer] > this.freqs[this.pointer-1]
this.norms[this.pointer] > this.freqs[this.pointerMax-1]
this.norms[this.pointer] > this.freqs[orig(this.doc)]
this.norms[this.pointer] > this.freqs[orig(this.pointer)]
this.norms[this.pointer] > this.freqs[orig(target)]
this.norms[this.pointer] > this.freqs[orig(target)-1]
this.norms[this.pointer-1] > this.docs[this.doc]
this.norms[this.pointer-1] > this.docs[this.doc-1]
this.norms[this.pointer-1] > this.freqs[this.doc]
this.norms[this.pointer-1] > this.freqs[this.doc-1]
this.norms[this.pointer-1] > this.docs[this.pointer-1]
this.norms[this.pointer-1] > this.docs[this.pointerMax]
this.norms[this.pointer-1] > this.docs[orig(this.doc)]
this.norms[this.pointer-1] > this.docs[orig(target)]
this.norms[this.pointer-1] > this.docs[orig(target)-1]
this.norms[this.pointer-1] > this.freqs[this.pointer]
this.norms[this.pointer-1] > this.freqs[this.pointer-1]
this.norms[this.pointer-1] > this.freqs[this.pointerMax-1]
this.norms[this.pointer-1] > this.freqs[orig(this.doc)]
this.norms[this.pointer-1] > this.freqs[orig(this.pointer)]
this.norms[this.pointer-1] > this.freqs[orig(target)]
this.norms[this.pointer-1] > this.freqs[orig(target)-1]
this.norms[this.pointerMax] != this.norms[this.pointerMax-1]
this.norms[this.pointerMax] > this.docs[this.doc]
this.norms[this.pointerMax] > this.docs[this.doc-1]

this.norms[this.pointerMax] > this.freqs[this.doc]
this.norms[this.pointerMax] > this.freqs[this.doc-1]
this.norms[this.pointerMax] > this.docs[this.pointer-1]
this.norms[this.pointerMax] > this.docs[orig(this.doc)]
this.norms[this.pointerMax] > this.docs[orig(target)]
this.norms[this.pointerMax] > this.docs[orig(target)-1]
this.norms[this.pointerMax] > this.freqs[this.pointer-1]
this.norms[this.pointerMax] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax] > this.freqs[orig(this.doc)]
this.norms[this.pointerMax] > this.freqs[orig(this.pointer)]
this.norms[this.pointerMax] > this.freqs[orig(target)]
this.norms[this.pointerMax] > this.freqs[orig(target)-1]
this.norms[this.pointerMax-1] > this.docs[this.doc]
this.norms[this.pointerMax-1] > this.docs[this.doc-1]
this.norms[this.pointerMax-1] > this.freqs[this.doc]
this.norms[this.pointerMax-1] > this.freqs[this.doc-1]
this.norms[this.pointerMax-1] > this.docs[this.pointer-1]
this.norms[this.pointerMax-1] > this.docs[this.pointerMax]
this.norms[this.pointerMax-1] > this.docs[orig(this.doc)]
this.norms[this.pointerMax-1] > this.docs[orig(target)]
this.norms[this.pointerMax-1] > this.docs[orig(target)-1]
this.norms[this.pointerMax-1] > this.freqs[this.pointer]
this.norms[this.pointerMax-1] > this.freqs[this.pointer-1]
this.norms[this.pointerMax-1] > this.freqs[this.pointerMax-1]
this.norms[this.pointerMax-1] > this.freqs[orig(this.doc)]
this.norms[this.pointerMax-1] > this.freqs[orig(this.pointer)]
this.norms[this.pointerMax-1] > this.freqs[orig(target)]
this.norms[this.pointerMax-1] > this.freqs[orig(target)-1]
this.norms[orig(this.doc)] > this.docs[this.doc]
this.norms[orig(this.doc)] > this.docs[this.doc-1]
this.norms[orig(this.doc)] > this.freqs[this.doc]
this.norms[orig(this.doc)] > this.freqs[this.doc-1]
this.norms[orig(this.doc)] > this.docs[this.pointer-1]
this.norms[orig(this.doc)] > this.docs[this.pointerMax]
this.norms[orig(this.doc)] > this.docs[orig(this.doc)]
this.norms[orig(this.doc)] > this.docs[orig(target)]
this.norms[orig(this.doc)] > this.docs[orig(target)-1]
this.norms[orig(this.doc)] > this.freqs[this.pointer]
this.norms[orig(this.doc)] > this.freqs[this.pointer-1]
this.norms[orig(this.doc)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.doc)] > this.freqs[orig(this.doc)]
this.norms[orig(this.doc)] > this.freqs[orig(this.pointer)]
this.norms[orig(this.doc)] > this.freqs[orig(target)]
this.norms[orig(this.doc)] > this.freqs[orig(target)-1]
this.norms[orig(this.pointer)] > this.docs[this.doc]
this.norms[orig(this.pointer)] > this.docs[this.doc-1]
this.norms[orig(this.pointer)] > this.freqs[this.doc]
this.norms[orig(this.pointer)] > this.freqs[this.doc-1]
this.norms[orig(this.pointer)] > this.docs[this.pointer-1]
this.norms[orig(this.pointer)] > this.docs[this.pointerMax]
this.norms[orig(this.pointer)] > this.docs[orig(this.doc)]
this.norms[orig(this.pointer)] > this.docs[orig(target)]
this.norms[orig(this.pointer)] > this.docs[orig(target)-1]
this.norms[orig(this.pointer)] > this.freqs[this.pointer]
this.norms[orig(this.pointer)] > this.freqs[this.pointer-1]
this.norms[orig(this.pointer)] > this.freqs[this.pointerMax-1]
this.norms[orig(this.pointer)] > this.freqs[orig(this.doc)]
this.norms[orig(this.pointer)] > this.freqs[orig(this.pointer)]
this.norms[orig(this.pointer)] > this.freqs[orig(target)]
this.norms[orig(this.pointer)] > this.freqs[orig(target)-1]
this.norms[orig(target)] != this.norms[orig(target)-1]
this.norms[orig(target)] > this.docs[this.doc]
this.norms[orig(target)] > this.docs[this.doc-1]

this.norms[orig(target)] > this.freqs[this.doc]
this.norms[orig(target)] > this.freqs[this.doc-1]
this.norms[orig(target)] > this.docs[this.pointer-1]
this.norms[orig(target)] > this.docs[this.pointerMax]
this.norms[orig(target)] > this.docs[orig(this.doc)]
this.norms[orig(target)] > this.docs[orig(target)]
this.norms[orig(target)] > this.docs[orig(target)-1]
this.norms[orig(target)] > this.freqs[this.pointer]
this.norms[orig(target)] > this.freqs[this.pointer-1]
this.norms[orig(target)] > this.freqs[this.pointerMax-1]
this.norms[orig(target)] > this.freqs[orig(this.doc)]
this.norms[orig(target)] > this.freqs[orig(this.pointer)]
this.norms[orig(target)] > this.freqs[orig(target)]
this.norms[orig(target)] > this.freqs[orig(target)-1]
this.norms[orig(target)-1] > this.docs[this.doc]
this.norms[orig(target)-1] > this.docs[this.doc-1]
this.norms[orig(target)-1] > this.freqs[this.doc]
this.norms[orig(target)-1] > this.freqs[this.doc-1]
this.norms[orig(target)-1] > this.docs[this.pointer-1]
this.norms[orig(target)-1] > this.docs[this.pointerMax]
this.norms[orig(target)-1] > this.docs[orig(this.doc)]
this.norms[orig(target)-1] > this.docs[orig(target)]
this.norms[orig(target)-1] > this.docs[orig(target)-1]
this.norms[orig(target)-1] > this.freqs[this.pointer]
this.norms[orig(target)-1] > this.freqs[this.pointer-1]
this.norms[orig(target)-1] > this.freqs[this.pointerMax-1]
this.norms[orig(target)-1] > this.freqs[orig(this.doc)]
this.norms[orig(target)-1] > this.freqs[orig(this.pointer)]
this.norms[orig(target)-1] > this.freqs[orig(target)]
this.norms[orig(target)-1] > this.freqs[orig(target)-1]
this.docs[this.doc] != this.docs[this.pointer-1]
this.docs[this.doc] >= this.docs[this.pointerMax]
this.docs[this.doc] >= this.docs[orig(target)]
this.docs[this.doc] != this.freqs[this.pointer]
this.docs[this.doc] < this.freqs[this.pointerMax-1]
this.docs[this.doc-1] >= this.docs[this.pointerMax]
this.docs[this.doc-1] >= this.docs[orig(this.doc)]
this.docs[this.doc-1] >= this.docs[orig(target)-1]
this.docs[this.doc-1] != this.freqs[this.pointer-1]
this.docs[this.doc-1] < this.freqs[this.pointerMax-1]
this.docs[this.doc-1] != this.freqs[orig(this.pointer)]
this.freqs[this.doc] >= this.docs[this.pointerMax]
this.freqs[this.doc] <= this.freqs[this.pointerMax-1]
this.freqs[this.doc] <= this.freqs[orig(target)]
this.freqs[this.doc-1] != this.docs[this.pointer-1]
this.freqs[this.doc-1] >= this.docs[this.pointerMax]
this.freqs[this.doc-1] <= this.freqs[this.pointerMax-1]
this.freqs[this.doc-1] >= this.freqs[orig(target)-1]
this.docs[this.pointer-1] > this.docs[this.pointerMax]
this.docs[this.pointer-1] != this.docs[orig(target)]
this.docs[this.pointer-1] < this.freqs[this.pointerMax-1]
this.docs[this.pointer-1] != this.freqs[orig(this.doc)]
this.docs[this.pointer-1] != this.freqs[orig(target)-1]
this.docs[this.pointerMax] <= this.docs[orig(this.doc)]
this.docs[this.pointerMax] <= this.docs[orig(target)]
this.docs[this.pointerMax] <= this.docs[orig(target)-1]
this.docs[this.pointerMax] < this.freqs[this.pointer]
this.docs[this.pointerMax] < this.freqs[this.pointer-1]
this.docs[this.pointerMax] < this.freqs[this.pointerMax-1]
this.docs[this.pointerMax] <= this.freqs[orig(this.doc)]
this.docs[this.pointerMax] < this.freqs[orig(this.pointer)]
this.docs[this.pointerMax] <= this.freqs[orig(target)]
this.docs[this.pointerMax] <= this.freqs[orig(target)-1]

this.docs[orig(this.doc)] <= this.docs[orig(target)-1]
this.docs[orig(this.doc)] != this.freqs[this.pointer-1]
this.docs[orig(this.doc)] < this.freqs[this.pointerMax-1]
this.docs[orig(this.doc)] != this.freqs[orig(this.pointer)]
this.docs[orig(target)] != this.freqs[this.pointer]
this.docs[orig(target)] < this.freqs[this.pointerMax-1]
this.docs[orig(target)-1] != this.freqs[this.pointer-1]
this.docs[orig(target)-1] < this.freqs[this.pointerMax-1]
this.docs[orig(target)-1] != this.freqs[orig(this.pointer)]
this.freqs[this.pointer] <= this.freqs[this.pointerMax-1]
this.freqs[this.pointer-1] <= this.freqs[this.pointerMax-1]
this.freqs[this.pointerMax-1] >= this.freqs[orig(this.doc)]
this.freqs[this.pointerMax-1] >= this.freqs[orig(this.pointer)]
this.freqs[this.pointerMax-1] >= this.freqs[orig(target)]
this.freqs[this.pointerMax-1] >= this.freqs[orig(target)-1]
==========
org.apache.lucene.search.TermScorer.skipTo(int):::EXIT;condition="return == true"

11.7 JSim

==========
laser.juliette.simulator.ConfigurationFactory:::OBJECT
this.runs == 1
this.oracle == null
this.listeners.getClass() == java.util.ArrayList.class
==========
laser.juliette.simulator.ConfigurationFactory.getOracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.tasks.Ta
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
==========
laser.juliette.simulator.ConfigurationFactory.getOracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.tasks.Ta
==========
laser.juliette.simulator.ConfigurationFactory.getOracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.tasks.Ta
this.runs == orig(this.runs)
this.adjuster == orig(this.adjuster)
this.oracle == orig(this.oracle)
this.trace == orig(this.trace)
this.summarizer == orig(this.summarizer)
this.listeners == orig(this.listeners)
tasks.adjuster == return.tasks.adjuster
tasks.adjuster == orig(tasks.adjuster)
tasks.tasks == return.tasks.tasks
tasks.tasks == orig(tasks.tasks)
return.config == orig(config)
return.tasks == orig(tasks)
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
return.context.getClass() == java.util.HashMap.class
this.listeners.getClass() == orig(this.listeners.getClass())
tasks.adjuster.getClass() == return.tasks.adjuster.getClass()
tasks.adjuster.getClass() == orig(tasks.adjuster.getClass())
==========
laser.juliette.simulator.Simulator:::OBJECT
this.root.getClass() == laser.juliette.simulator.ams.AgendaItem.class
==========
laser.juliette.simulator.Simulator.startProcess(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.jul.JulFile, java.lang.Class
laser.juliette.simulator.tasks.TaskList):::ENTER
this.root == null
classloader.getClass() == java.net.URLClassLoader.class
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
==========
laser.juliette.simulator.Simulator.startProcess(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.jul.JulFile, java.lang.Class
laser.juliette.simulator.tasks.TaskList):::EXIT
tasks.adjuster == orig(tasks.adjuster)
tasks.tasks == orig(tasks.tasks)
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class

tasks.adjuster.getClass() == orig(tasks.adjuster.getClass())
==========
laser.juliette.simulator.extension.stub.SimpleOracle:::OBJECT
==========
laser.juliette.simulator.extension.stub.SimpleOracle.SimpleOracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulat
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
==========
laser.juliette.simulator.extension.stub.SimpleOracle.SimpleOracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulat
tasks.adjuster == orig(tasks.adjuster)
tasks.tasks == orig(tasks.tasks)
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
tasks.adjuster.getClass() == orig(tasks.adjuster.getClass())
==========
laser.juliette.simulator.oracle.CompleteAction:::OBJECT
this.delay != null
this.delay.getClass() == laser.juliette.simulator.oracle.FixedDuration.class
==========
laser.juliette.simulator.oracle.CompleteAction.perform(laser.juliette.ams.AgendaItem, laser.juliette.simulator.tasks.TaskList):::ENTER
item != null
item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
tasks != null
tasks.adjuster != null
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
tasks.tasks != null
this.delay.getClass() != item.getClass()
this.delay.getClass() != tasks.adjuster.getClass()
item.getClass() != tasks.adjuster.getClass()
==========
laser.juliette.simulator.oracle.CompleteAction.perform(laser.juliette.ams.AgendaItem, laser.juliette.simulator.tasks.TaskList):::EXIT
this.delay == orig(this.delay)
tasks.adjuster == orig(tasks.adjuster)
tasks.tasks == orig(tasks.tasks)
tasks.adjuster != null
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
tasks.tasks != null
this.delay.getClass() != tasks.adjuster.getClass()
this.delay.getClass() == orig(this.delay.getClass())
this.delay.getClass() != orig(item.getClass())
tasks.adjuster.getClass() != orig(item.getClass())
tasks.adjuster.getClass() == orig(tasks.adjuster.getClass())
==========
laser.juliette.simulator.oracle.MessageEventFactory:::OBJECT
this.message != null
==========
laser.juliette.simulator.oracle.MessageEventFactory.createTask():::ENTER
==========
laser.juliette.simulator.oracle.MessageEventFactory.createTask():::EXIT
this.message == orig(this.message)
return.delay == return.time
return != null
return.getClass() == laser.juliette.simulator.tasks.MessageTask.class
return.tasks == null
return.assignedAt == 0
return.order >= 0
laser.juliette.simulator.tasks.Task.index >= 1
return.delay > return.assignedAt
return.delay > return.order
return.delay > laser.juliette.simulator.tasks.Task.index
return.assignedAt <= return.order
return.assignedAt < laser.juliette.simulator.tasks.Task.index
return.order - laser.juliette.simulator.tasks.Task.index + 1 == 0
==========
laser.juliette.simulator.oracle.Oracle:::OBJECT

this.config != null
this.tasks != null
this.tasks.adjuster != null
this.tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
this.tasks.tasks != null
this.context != null
this.context.getClass() == java.util.HashMap.class
this.tasks.adjuster.getClass() != this.context.getClass()
==========
laser.juliette.simulator.oracle.Oracle.Oracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.tasks.TaskList):::E
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
==========
laser.juliette.simulator.oracle.Oracle.Oracle(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.tasks.TaskList):::E
this.config == orig(config)
this.tasks == orig(tasks)
this.tasks.adjuster == tasks.adjuster
this.tasks.tasks == tasks.tasks
tasks.adjuster == orig(tasks.adjuster)
tasks.tasks == orig(tasks.tasks)
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
this.tasks.adjuster.getClass() == tasks.adjuster.getClass()
tasks.adjuster.getClass() == orig(tasks.adjuster.getClass())
==========
laser.juliette.simulator.oracle.Oracle.getTaskList():::ENTER
==========
laser.juliette.simulator.oracle.Oracle.getTaskList():::EXIT
this.config == orig(this.config)
this.tasks == return
this.tasks.adjuster == return.adjuster
this.tasks.tasks == return.tasks
this.context == orig(this.context)
return == orig(this.tasks)
return.adjuster == orig(this.tasks.adjuster)
return.tasks == orig(this.tasks.tasks)
return != null
return.adjuster != null
return.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
return.tasks != null
this.tasks.adjuster.getClass() == return.adjuster.getClass()
this.context.getClass() != return.adjuster.getClass()
this.context.getClass() == orig(this.context.getClass())
return.adjuster.getClass() == orig(this.tasks.adjuster.getClass())
==========
laser.juliette.simulator.oracle.StartAction:::OBJECT
this.delay != null
this.delay.getClass() == laser.juliette.simulator.oracle.FixedDuration.class
==========
laser.juliette.simulator.oracle.StartAction.perform(laser.juliette.ams.AgendaItem, laser.juliette.simulator.tasks.TaskList):::ENTER
item != null
item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
tasks != null
tasks.adjuster != null
tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
tasks.tasks != null
this.delay.getClass() != item.getClass()
this.delay.getClass() != tasks.adjuster.getClass()
item.getClass() != tasks.adjuster.getClass()
==========
laser.juliette.simulator.oracle.StartAction.perform(laser.juliette.ams.AgendaItem, laser.juliette.simulator.tasks.TaskList):::EXIT
this.delay == orig(this.delay)
tasks.adjuster == orig(tasks.adjuster)
tasks.tasks == orig(tasks.tasks)
tasks.adjuster != null

tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
tasks.tasks != null
this.delay.getClass() != tasks.adjuster.getClass()
this.delay.getClass() == orig(this.delay.getClass())
this.delay.getClass() != orig(item.getClass())
tasks.adjuster.getClass() != orig(item.getClass())
tasks.adjuster.getClass() == orig(tasks.adjuster.getClass())
===
laser.juliette.simulator.tasks.CompleteTask:::OBJECT
===
laser.juliette.simulator.tasks.CompleteTask.CompleteTask(laser.juliette.ams.AgendaItem, long):::ENTER
item != null
item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
delay >= 1
===
laser.juliette.simulator.tasks.CompleteTask.CompleteTask(laser.juliette.ams.AgendaItem, long):::EXIT
===
laser.juliette.simulator.tasks.CompleteTask.action():::ENTER
===
laser.juliette.simulator.tasks.CompleteTask.action():::EXIT
===
laser.juliette.simulator.tasks.CompleteTask.getTaskKind():::ENTER
===
laser.juliette.simulator.tasks.CompleteTask.getTaskKind():::EXIT
return == laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.COMPLETE
return != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.START != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.OPT_OUT != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.TERMINATE != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.DEADLINE != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.SET != null
===
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind:::CLASS
===
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind:::OBJECT
===
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.ItemTask$ItemTaskKind(java.lang.String, int):::ENTER
arg0 != null
arg1 >= 0
===
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.ItemTask$ItemTaskKind(java.lang.String, int):::EXIT
arg0.toString == orig(arg0.toString)
===
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.values():::ENTER
===
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.values():::EXIT
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.START == orig(laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.START)
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.OPT_OUT == orig(laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.OPT_O
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.COMPLETE == orig(laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.COM
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.TERMINATE == orig(laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.TER
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.DEADLINE == orig(laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.DEAD
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.SET == orig(laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.SET)
return.getClass() == laser.juliette.simulator.tasks.ItemTask$ItemTaskKind[].class
size(return[]) == 6
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.START in return[]
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.OPT_OUT in return[]
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.COMPLETE in return[]
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.TERMINATE in return[]
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.DEADLINE in return[]
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.SET in return[]
===
laser.juliette.simulator.tasks.ItemTask:::OBJECT
this.item != null

this.item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
this.getClass() != this.item.getClass()
==========
laser.juliette.simulator.tasks.ItemTask.ItemTask(laser.juliette.ams.AgendaItem, long):::ENTER
item != null
item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
delay >= 1
==========
laser.juliette.simulator.tasks.ItemTask.ItemTask(laser.juliette.ams.AgendaItem, long):::EXIT
this.item == orig(item)
this.item.getClass() == orig(item.getClass())
==========
laser.juliette.simulator.tasks.ItemTask.getItem():::ENTER
==========
laser.juliette.simulator.tasks.ItemTask.getItem():::EXIT
this.item == return
return == orig(this.item)
return != null
return.getClass() == laser.juliette.simulator.ams.AgendaItem.class
this.item.getClass() == return.getClass()
return.getClass() != orig(this.getClass())
return.getClass() == orig(this.item.getClass())
==========
laser.juliette.simulator.tasks.MessageTask:::OBJECT
this.message != null
==========
laser.juliette.simulator.tasks.MessageTask.MessageTask(long, laser.lj.template.AggregateInstanceDescriptor):::ENTER
message != null
==========
laser.juliette.simulator.tasks.MessageTask.MessageTask(long, laser.lj.template.AggregateInstanceDescriptor):::EXIT
this.message == orig(message)
==========
laser.juliette.simulator.tasks.MessageTask.action():::ENTER
==========
laser.juliette.simulator.tasks.MessageTask.action():::EXIT
this.message == orig(this.message)
==========
laser.juliette.simulator.tasks.StartTask:::OBJECT
==========
laser.juliette.simulator.tasks.StartTask.StartTask(laser.juliette.ams.AgendaItem, long):::ENTER
item != null
item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
delay one of 1, 2
==========
laser.juliette.simulator.tasks.StartTask.StartTask(laser.juliette.ams.AgendaItem, long):::EXIT
==========
laser.juliette.simulator.tasks.StartTask.action():::ENTER
==========
laser.juliette.simulator.tasks.StartTask.action():::EXIT
==========
laser.juliette.simulator.tasks.StartTask.getTaskKind():::ENTER
==========
laser.juliette.simulator.tasks.StartTask.getTaskKind():::EXIT
return == laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.START
return != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.OPT_OUT != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.COMPLETE != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.TERMINATE != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.DEADLINE != null
laser.juliette.simulator.tasks.ItemTask$ItemTaskKind.SET != null
==========
laser.juliette.simulator.tasks.Task:::CLASS
==========

laser.juliette.simulator.tasks.Task:::OBJECT
this.tasks.adjuster != null
this.tasks.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
this.tasks.tasks != null
this.delay >= 1
this.assignedAt >= 0
this.time >= 1
this.order >= 0
this.getClass() != this.tasks.adjuster.getClass()
this.delay != this.assignedAt
this.delay <= this.time
this.assignedAt < this.time
this.order < laser.juliette.simulator.tasks.Task.index
this.delay + this.assignedAt - this.time == 0
==========
laser.juliette.simulator.tasks.Task.Task(long):::ENTER
delay >= 1
==========
laser.juliette.simulator.tasks.Task.Task(long):::EXIT
this.delay == orig(delay)
this.tasks == null
this.delay != laser.juliette.simulator.tasks.Task.index
this.order - laser.juliette.simulator.tasks.Task.index + 1 == 0
==========
laser.juliette.simulator.tasks.Task.advanceClock():::ENTER
this.tasks != null
==========
laser.juliette.simulator.tasks.Task.advanceClock():::EXIT
this.tasks == orig(this.tasks)
this.tasks.adjuster == orig(this.tasks.adjuster)
this.tasks.tasks == orig(this.tasks.tasks)
this.delay == orig(this.delay)
this.assignedAt == orig(this.assignedAt)
this.time == orig(this.time)
this.order == orig(this.order)
laser.juliette.simulator.tasks.Task.index == orig(laser.juliette.simulator.tasks.Task.index)
this.tasks != null
this.tasks.adjuster.getClass() != orig(this.getClass())
this.tasks.adjuster.getClass() == orig(this.tasks.adjuster.getClass())
==========
laser.juliette.simulator.tasks.Task.compareTo(laser.juliette.simulator.tasks.Task):::ENTER
this.tasks == o.tasks
this.tasks.adjuster == o.tasks.adjuster
this.tasks.tasks == o.tasks.tasks
this.tasks != null
o != null
o.delay >= 1
o.assignedAt >= 0
o.time >= 2
o.order >= 0
this.tasks.adjuster.getClass() != o.getClass()
this.tasks.adjuster.getClass() == o.tasks.adjuster.getClass()
this.assignedAt <= o.time
this.time >= o.assignedAt
this.order != o.order
laser.juliette.simulator.tasks.Task.index > o.order
o.delay != o.assignedAt
o.delay <= o.time
o.assignedAt < o.time
o.delay + o.assignedAt - o.time == 0
==========
laser.juliette.simulator.tasks.Task.compareTo(laser.juliette.simulator.tasks.Task):::EXIT63
this.time == o.time

this.time == orig(o.time)
this.delay != this.order
this.delay != laser.juliette.simulator.tasks.Task.index
this.delay != o.assignedAt
this.assignedAt != o.delay
this.assignedAt != o.order
this.time >= o.delay
this.time > o.assignedAt
this.order != o.delay
this.order > return
laser.juliette.simulator.tasks.Task.index > return
this.time - o.delay - o.assignedAt == 0
this.order - o.order - return == 0
==========
laser.juliette.simulator.tasks.Task.compareTo(laser.juliette.simulator.tasks.Task):::EXIT65
this.delay != o.time
this.delay >= return
this.time != o.delay
this.time != o.time
this.time > return
this.time - o.time - return == 0
==========
laser.juliette.simulator.tasks.Task.compareTo(laser.juliette.simulator.tasks.Task):::EXIT
this.tasks == o.tasks
this.tasks == orig(this.tasks)
this.tasks == orig(o.tasks)
this.tasks.adjuster == o.tasks.adjuster
this.tasks.adjuster == orig(this.tasks.adjuster)
this.tasks.adjuster == orig(o.tasks.adjuster)
this.tasks.tasks == o.tasks.tasks
this.tasks.tasks == orig(this.tasks.tasks)
this.tasks.tasks == orig(o.tasks.tasks)
this.delay == orig(this.delay)
this.assignedAt == orig(this.assignedAt)
this.time == orig(this.time)
this.order == orig(this.order)
laser.juliette.simulator.tasks.Task.index == orig(laser.juliette.simulator.tasks.Task.index)
o.delay == orig(o.delay)
o.assignedAt == orig(o.assignedAt)
o.time == orig(o.time)
o.order == orig(o.order)
(this.time != o.time) ==> (o.time >= 2)
(this.time != o.time) ==> (this.delay != o.time)
(this.time != o.time) ==> (this.delay >= return)
(this.time != o.time) ==> (this.time != o.delay)
(this.time != o.time) ==> (this.time - o.time - return == 0)
(this.time != o.time) ==> (this.time > return)
(this.time != o.time) ==> (this.time >= 1)
(this.time == o.time) ==> (laser.juliette.simulator.tasks.Task.index > return)
(this.time == o.time) ==> (this.assignedAt != o.delay)
(this.time == o.time) ==> (this.assignedAt != o.order)
(this.time == o.time) ==> (this.assignedAt < o.time)
(this.time == o.time) ==> (this.delay != laser.juliette.simulator.tasks.Task.index)
(this.time == o.time) ==> (this.delay != o.assignedAt)
(this.time == o.time) ==> (this.delay != this.order)
(this.time == o.time) ==> (this.delay + this.assignedAt - o.time == 0)
(this.time == o.time) ==> (this.delay <= o.time)
(this.time == o.time) ==> (this.order != o.delay)
(this.time == o.time) ==> (this.order - o.order - return == 0)
(this.time == o.time) ==> (this.order > return)
(this.time == o.time) ==> (this.time - o.delay - o.assignedAt == 0)
(this.time == o.time) ==> (this.time > o.assignedAt)
(this.time == o.time) ==> (this.time >= o.delay)

this.tasks != null
o.delay >= 1
o.assignedAt >= 0
o.time >= 2
o.order >= 0
return != 0
this.tasks.adjuster.getClass() == o.tasks.adjuster.getClass()
this.tasks.adjuster.getClass() != orig(this.getClass())
this.tasks.adjuster.getClass() == orig(this.tasks.adjuster.getClass())
this.tasks.adjuster.getClass() != orig(o.getClass())
this.tasks.adjuster.getClass() == orig(o.tasks.adjuster.getClass())
this.assignedAt <= o.time
this.assignedAt != return
this.time >= o.assignedAt
this.order != o.order
laser.juliette.simulator.tasks.Task.index > o.order
o.delay != o.assignedAt
o.delay <= o.time
o.assignedAt < o.time
o.delay + o.assignedAt - o.time == 0
==========
laser.juliette.simulator.tasks.Task.perform():::ENTER
this.tasks != null
==========
laser.juliette.simulator.tasks.Task.perform():::EXIT
this.tasks == orig(this.tasks)
this.tasks.adjuster == orig(this.tasks.adjuster)
this.tasks.tasks == orig(this.tasks.tasks)
this.delay == orig(this.delay)
this.assignedAt == orig(this.assignedAt)
this.time == orig(this.time)
this.order == orig(this.order)
this.tasks != null
this.tasks.adjuster.getClass() != orig(this.getClass())
this.tasks.adjuster.getClass() == orig(this.tasks.adjuster.getClass())
this.order < orig(laser.juliette.simulator.tasks.Task.index)
laser.juliette.simulator.tasks.Task.index >= orig(laser.juliette.simulator.tasks.Task.index)
==========
laser.juliette.simulator.tasks.Task.setTaskList(laser.juliette.simulator.tasks.TaskList):::ENTER
this.tasks == null
list != null
list.adjuster != null
list.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
list.tasks != null
this.getClass() != list.adjuster.getClass()
this.delay != laser.juliette.simulator.tasks.Task.index
==========
laser.juliette.simulator.tasks.Task.setTaskList(laser.juliette.simulator.tasks.TaskList):::EXIT
this.tasks == orig(list)
this.tasks.adjuster == list.adjuster
this.tasks.tasks == list.tasks
this.delay == orig(this.delay)
this.assignedAt == orig(this.assignedAt)
this.time == orig(this.time)
this.order == orig(this.order)
laser.juliette.simulator.tasks.Task.index == orig(laser.juliette.simulator.tasks.Task.index)
list.adjuster == orig(list.adjuster)
list.tasks == orig(list.tasks)
this.tasks != null
list.adjuster != null
list.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
list.tasks != null
this.tasks.adjuster.getClass() == list.adjuster.getClass()

this.delay != laser.juliette.simulator.tasks.Task.index
list.adjuster.getClass() != orig(this.getClass())
list.adjuster.getClass() == orig(list.adjuster.getClass())
==========
laser.juliette.simulator.tasks.TaskList:::OBJECT
this.adjuster != null
this.adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
this.tasks != null
==========
laser.juliette.simulator.tasks.TaskList.TaskList(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.extension.Adjus
adjuster.getClass() == laser.romeo.jsim.TaskAdjuster.class
==========
laser.juliette.simulator.tasks.TaskList.TaskList(laser.juliette.simulator.oracle.OracleConfiguration, laser.juliette.simulator.extension.Adjus
this.adjuster == orig(adjuster)
this.adjuster.getClass() == orig(adjuster.getClass())
==========
laser.juliette.simulator.tasks.TaskList.addTask(laser.juliette.simulator.tasks.Task):::ENTER
task != null
task.tasks == null
task.delay >= 1
task.assignedAt >= 0
task.time >= 1
task.order >= 0
this.adjuster.getClass() != task.getClass()
task.delay != task.assignedAt
task.delay <= task.time
task.delay != laser.juliette.simulator.tasks.Task.index
task.assignedAt < task.time
task.order < laser.juliette.simulator.tasks.Task.index
task.delay + task.assignedAt - task.time == 0
==========
laser.juliette.simulator.tasks.TaskList.addTask(laser.juliette.simulator.tasks.Task):::EXIT
this.adjuster == task.tasks.adjuster
this.adjuster == orig(this.adjuster)
this.tasks == task.tasks.tasks
this.tasks == orig(this.tasks)
task.tasks == orig(this)
task.delay == orig(task.delay)
task.assignedAt == orig(task.assignedAt)
task.time == orig(task.time)
task.order == orig(task.order)
laser.juliette.simulator.tasks.Task.index == orig(laser.juliette.simulator.tasks.Task.index)
task.delay >= 1
task.assignedAt >= 0
task.time >= 1
task.order >= 0
this.adjuster.getClass() == task.tasks.adjuster.getClass()
this.adjuster.getClass() == orig(this.adjuster.getClass())
this.adjuster.getClass() != orig(task.getClass())
task.delay != task.assignedAt
task.delay <= task.time
task.delay != laser.juliette.simulator.tasks.Task.index
task.assignedAt < task.time
task.order < laser.juliette.simulator.tasks.Task.index
task.delay + task.assignedAt - task.time == 0
==========
laser.juliette.simulator.tasks.TaskList.adjust(laser.juliette.simulator.tasks.ItemTask):::ENTER
task != null
task.item != null
task.item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
this.adjuster.getClass() != task.getClass()
this.adjuster.getClass() != task.item.getClass()
task.getClass() != task.item.getClass()

```
==========================================================================
laser.juliette.simulator.tasks.TaskList.adjust(laser.juliette.simulator.tasks.ItemTask):::EXIT
this.adjuster == orig(this.adjuster)
this.tasks == orig(this.tasks)
task.item == orig(task.item)
task.item != null
task.item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
this.adjuster.getClass() != task.item.getClass()
this.adjuster.getClass() == orig(this.adjuster.getClass())
this.adjuster.getClass() != orig(task.getClass())
task.item.getClass() != orig(task.getClass())
task.item.getClass() == orig(task.item.getClass())
==========================================================================
laser.juliette.simulator.tasks.TaskList.getNextTask():::ENTER
==========================================================================
laser.juliette.simulator.tasks.TaskList.getNextTask():::EXIT
this.adjuster == return.tasks.adjuster
this.adjuster == orig(this.adjuster)
this.tasks == return.tasks.tasks
this.tasks == orig(this.tasks)
return.tasks == orig(this)
return != null
return.delay >= 1
return.assignedAt >= 0
return.time >= 1
return.order >= 0
this.adjuster.getClass() != return.getClass()
this.adjuster.getClass() == return.tasks.adjuster.getClass()
this.adjuster.getClass() == orig(this.adjuster.getClass())
return.delay != return.assignedAt
return.delay <= return.time
return.assignedAt < return.time
return.order < laser.juliette.simulator.tasks.Task.index
return.delay + return.assignedAt - return.time == 0
==========================================================================
laser.juliette.simulator.tasks.TaskList.updateTaskList(laser.juliette.simulator.tasks.Task):::ENTER
task != null
task.tasks == null
task.delay >= 1
task.assignedAt >= 0
task.time >= 1
task.order >= 0
this.adjuster.getClass() != task.getClass()
task.delay != task.assignedAt
task.delay <= task.time
task.delay != laser.juliette.simulator.tasks.Task.index
task.assignedAt < task.time
task.order < laser.juliette.simulator.tasks.Task.index
task.delay + task.assignedAt - task.time == 0
==========================================================================
laser.juliette.simulator.tasks.TaskList.updateTaskList(laser.juliette.simulator.tasks.Task):::EXIT
this.adjuster == task.tasks.adjuster
this.adjuster == orig(this.adjuster)
this.tasks == task.tasks.tasks
this.tasks == orig(this.tasks)
task.tasks == orig(this)
task.delay == orig(task.delay)
task.assignedAt == orig(task.assignedAt)
task.time == orig(task.time)
task.order == orig(task.order)
laser.juliette.simulator.tasks.Task.index == orig(laser.juliette.simulator.tasks.Task.index)
task.delay >= 1
task.assignedAt >= 0
```

task.time >= 1
task.order >= 0
this.adjuster.getClass() == task.tasks.adjuster.getClass()
this.adjuster.getClass() == orig(this.adjuster.getClass())
this.adjuster.getClass() != orig(task.getClass())
task.delay != task.assignedAt
task.delay <= task.time
task.delay != laser.juliette.simulator.tasks.Task.index
task.assignedAt < task.time
task.order < laser.juliette.simulator.tasks.Task.index
task.delay + task.assignedAt - task.time == 0
==========
laser.romeo.jsim.TaskAdjuster:::OBJECT
==========
laser.romeo.jsim.TaskAdjuster.TaskAdjuster():::EXIT
==========
laser.romeo.jsim.TaskAdjuster.consider(laser.juliette.simulator.tasks.ItemTask, java.util.Set):::ENTER
task != null
task.item != null
task.item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
related != null
related.getClass() == java.util.HashSet.class
task.getClass() != task.item.getClass()
task.getClass() != related.getClass()
task.item.getClass() != related.getClass()
==========
laser.romeo.jsim.TaskAdjuster.consider(laser.juliette.simulator.tasks.ItemTask, java.util.Set):::EXIT24
==========
laser.romeo.jsim.TaskAdjuster.consider(laser.juliette.simulator.tasks.ItemTask, java.util.Set):::EXIT48
==========
laser.romeo.jsim.TaskAdjuster.consider(laser.juliette.simulator.tasks.ItemTask, java.util.Set):::EXIT
task.item == orig(task.item)
task.item != null
task.item.getClass() == laser.juliette.simulator.ams.AgendaItem.class
task.item.getClass() != orig(task.getClass())
task.item.getClass() == orig(task.item.getClass())
task.item.getClass() != orig(related.getClass())

12. APPENDIX B. INVARIANTS USED IN EXPERIMENT #2

Following are the Daikon invariants (in the Daikon format) for the applications used in the experiment described in Section 4.

12.1 JSim

Daikon version 4.6.0, released July 1, 2009; http://pag.csail.mit.edu/daikon.
Reading declaration files
[3:23:02 PM]:
Processing trace data; reading 10 dtrace files:
[3:23:03 PM]: Finished reading Driver9.dtrace.gz
==========
laser.juliette.simulator.oracle.LinearRangeDuration:::OBJECT
this.min == 1
this.max == 10
==========
laser.juliette.simulator.oracle.LinearRangeDuration.get():::ENTER
==========
laser.juliette.simulator.oracle.LinearRangeDuration.get():::EXIT
this.min == orig(this.min)
this.max == orig(this.max)
return >= 1
this.min <= return
this.max >= return
Exiting Daikon.

12.2 MartiRank

```
==========================================================================
..check_var():::ENTER
::n_ids == num_ids
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::unknown_limit == unknown_limit
::debug == ::exti
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of  0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 
```

0, 0, 0, 0, 0, 0, 0]
::score[] elements == 0
::ignore_set has only one value
::ignore_set != null
::ignore_set[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
::ignore_set[] elements == 0
::unknown_limit == 0.4
::debug == 0
::gargc == 4
::gargv != null
::gargv[] == [./marti-score]
::gargv[] elements == "./marti-score"
::tabula has only one value
::tabula != null
::tabula[] contains no nulls and has only one value, of length 1000
::tabula[] elements != null
::tabula[][0] == [4339.0, 7094.0, 5649.0, 9134.0, NaN, NaN, 6407.0, 3591.0, 3345.0, 7524.0, 8499.0, 255.0, 3868.0, 6849.0, 2687.0, 4037.0, 7926.0, 6679.0, NaN, NaN, 9104.0, 8245.0, 6668.0, 9905.0, 1113.0, 6745.0, 7828.0, 1840.0, 1096.0, 8668.0, 6718.0, 669.0, 6795.0, 1506.0, 3526.0, 7062.0, 1607.0, 6023.0, 615.0, 1708.0, 6033.0, 4168.0, 6423.0, 9043.0, 2.0, 3686.0, NaN, 7497.0, 3165.0, 88.0, 3675.0, 9148.0, 4506.0, 1349.0, 4460.0, 1862.0, 2181.0, 9759.0, 6082.0, 6391.0, 887.0, 4895.0, 5074.0, 66.0, 3768.0, 5624.0, 2879.0, 5570.0, 4253.0, NaN, 1170.0, NaN, 4928.0, 3972.0, 263.0, 6131.0, 3243.0, 3968.0, 5495.0, 8520.0, 1204.0, 9330.0, 3313.0, 5329.0, 8082.0, 6576.0, 9983.0, 5836.0, 7842.0, 2729.0, 9092.0, 1150.0, 5316.0, 4948.0, 5353.0, 7879.0, 5966.0, 3139.0, NaN, 1215.0, 3000.0, NaN, 5402.0, 8314.0, 5692.0, 2236.0, 5140.0, 2773.0, 1293.0, 1891.0, 2479.0, 2690.0, 1426.0, 2784.0, 104.0, 6815.0, 423.0, 7337.0, 8822.0, 9804.0, 1999.0, 8514.0, 7282.0, NaN, 8426.0, 5504.0, 605.0, 9321.0, 9126.0, 5935.0, 3984.0, 9825.0, NaN, NaN, 3610.0, 8321.0, 6730.0, 2323.0, 8243.0, 1454.0, 266.0, 413.0, 3319.0, 6700.0, 3483.0, NaN, 3858.0, NaN, 5279.0, 7489.0, 9326.0, 5429.0, 1108.0, 2881.0, 1177.0, 118.0, 8277.0, 9167.0, 9512.0, 6145.0, 8018.0, 2815.0, 6258.0, 9817.0, 5158.0, 6894.0, 4820.0, NaN, 6810.0, 3065.0, 6442.0, 5859.0, 8450.0, 9175.0, 5911.0, 2241.0, 4057.0, 9222.0, NaN, 2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0,

7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]
::tabula[][0] elements >= 2.0
var_idx >= 0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > var_idx
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > var_idx
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > var_idx
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= var_idx
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
var_idx in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= var_idx
==========
..check_var():::EXIT
::n_ids == orig(::n_ids)
::n_ids == orig(num_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)

::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::unknown_limit == orig(unknown_limit)
::debug == ::exti
::debug == return
::debug == orig(::debug)
::debug == orig(::exti)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

0, 0, 0,

2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > orig(var_idx)
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > orig(var_idx)
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > orig(var_idx)
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= orig(var_idx)
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
orig(var_idx) in ::order[]

::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= orig(var_idx)
==========
..compasc():::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->dir < ::order[] (lexically)
::rounds[].bins->dir <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

0, 0, 0,

2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]
::tabula[][0] elements >= 2.0
::exti >= 0
ap != null
bp != null
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
::gargc != ::exti
==========
..compasc():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)

::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir < ::order[] (lexically)
::rounds[].bins->dir <= ::order[] (lexically)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null

::score[] == [0, 0, 0,

2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
return one of -1, 0, 1
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::n_ids > return
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_names > return
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::n_rounds > return
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].num_bins elements >= return
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc

::rounds[].bins->last elements >= ::exti
::rounds[].bins->last elements > return
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
::gargc != ::exti
::gargc > return
::exti != return
==========
..compdesc()::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

0, 0, 0,

7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
ap != null
bp != null
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug

::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
==========
..compdesc()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0

::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
return one of -1, 0, 1
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::n_ids > return
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_names > return
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti

::n_rounds > return
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].num_bins elements >= return
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->last elements > return
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
::gargc > return
==========
..dleg()::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0

::score has only one value
::score != null
::score[] == [0, 0, 0,

7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)

::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::unknown_limit < a
::unknown_limit < b
::debug <= ::exti
==========
..dleg()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10

::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0

::exti >= 0

return one of -1, 0, 1

::rounds[].num_bins elements < ::n_ids

::rounds[].bins->last elements < ::n_ids

::rounds[].bins->var elements < ::n_ids

::rounds[].bins->dir elements < ::n_ids

::order[] elements < ::n_ids

::n_ids > ::exti

::n_ids > return

::rounds[].num_bins elements < ::n_names

::rounds[].bins->var elements < ::n_names

::n_names in ::order[]

::n_names > ::exti
::n_names > return
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::n_rounds > return
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].num_bins elements >= return
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->last elements > return
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::unknown_limit < orig(a)
::unknown_limit < orig(b)
::debug <= ::exti
::gargc > return
==========
..main()::ENTER
::n_ids == ::n_names
::n_ids == ::n_rounds
::n_ids == ::debug
::n_ids == ::gargc
::n_ids == ::exti
::order == ::score
::order == ::ignore_set
::n_ids == 0
::rounds == null
::order == null
::unknown_limit == 0.4
::gargv == null
::tabula == null
argc == 4
argv != null
argv[] == [./marti-score]
argv[] elements == "./marti-score"
envp != null
envp[] == [LESSOPEN=| /usr/bin/lesspipe
envp[] elements == "LESSOPEN=| /usr/bin/lesspipe
==========
..main()::EXIT
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins < ::score[] (lexically)
::rounds[].num_bins <= ::score[] (lexically)

::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last > ::score[] (lexically)
::rounds[].bins->last >= ::score[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var < ::score[] (lexically)
::rounds[].bins->var <= ::score[] (lexically)
::rounds[].bins->dir < ::order[] (lexically)
::rounds[].bins->dir <= ::order[] (lexically)
::rounds[].bins->dir < ::score[] (lexically)
::rounds[].bins->dir <= ::score[] (lexically)
::order[] < ::score[] (lexically)
::order[] <= ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::score[] > ::ignore_set[] (lexically)
::score[] >= ::ignore_set[] (lexically)
::unknown_limit == orig(::unknown_limit)
::debug == return
::debug == orig(::n_ids)
::debug == orig(::n_names)
::debug == orig(::n_rounds)
::debug == orig(::debug)
::debug == orig(::gargc)
::debug == orig(::exti)
::gargc == orig(argc)
::gargv == orig(argv)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] elements >= 1
::ignore_set has only one value
::ignore_set != null
::ignore_set[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
::ignore_set[] elements == 0
::unknown_limit == 0.4
::debug == 0
::gargc == 4

::gargv != null
::gargv[] == [./marti-score]
::gargv[] elements == "./marti-score"
::tabula has only one value
::tabula != null
::tabula[] contains no nulls and has only one value, of length 1000
::tabula[] elements != null
::tabula[][0] == [4339.0, 7094.0, 5649.0, 9134.0, NaN, NaN, 6407.0, 3591.0, 3345.0, 7524.0, 8499.0, 255.0, 3868.0, 6849.0, 2687.0, 4037.0, 7926.0, 6679.0, NaN, NaN, 9104.0, 8245.0, 6668.0, 9905.0, 1113.0, 6745.0, 7828.0, 1840.0, 1096.0, 8668.0, 6718.0, 669.0, 6795.0, 1506.0, 3526.0, 7062.0, 1607.0, 6023.0, 615.0, 1708.0, 6033.0, 4168.0, 6423.0, 9043.0, 2.0, 3686.0, NaN, 7497.0, 3165.0, 88.0, 3675.0, 9148.0, 4506.0, 1349.0, 4460.0, 1862.0, 2181.0, 9759.0, 6082.0, 6391.0, 887.0, 4895.0, 5074.0, 66.0, 3768.0, 5624.0, 2879.0, 5570.0, 4253.0, NaN, 1170.0, NaN, 4928.0, 3972.0, 263.0, 6131.0, 3243.0, 3968.0, 5495.0, 8520.0, 1204.0, 9330.0, 3313.0, 5329.0, 8082.0, 6576.0, 9983.0, 5836.0, 7842.0, 2729.0, 9092.0, 1150.0, 5316.0, 4948.0, 5353.0, 7879.0, 5966.0, 3139.0, NaN, 1215.0, 3000.0, NaN, 5402.0, 8314.0, 5692.0, 2236.0, 5140.0, 2773.0, 1293.0, 1891.0, 2479.0, 2690.0, 1426.0, 2784.0, 104.0, 6815.0, 423.0, 7337.0, 8822.0, 9804.0, 1999.0, 8514.0, 7282.0, NaN, 8426.0, 5504.0, 605.0, 9321.0, 9126.0, 5935.0, 3984.0, 9825.0, NaN, NaN, 3610.0, 8321.0, 6730.0, 2323.0, 8243.0, 1454.0, 266.0, 413.0, 3319.0, 6700.0, 3483.0, NaN, 3858.0, NaN, 5279.0, 7489.0, 9326.0, 5429.0, 1108.0, 2881.0, 1177.0, 118.0, 8277.0, 9167.0, 9512.0, 6145.0, 8018.0, 2815.0, 6258.0, 9817.0, 5158.0, 6894.0, 4820.0, NaN, 6810.0, 3065.0, 6442.0, 5859.0, 8450.0, 9175.0, 5911.0, 2241.0, 4057.0, 9222.0, NaN, 2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0,

NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]
::tabula[][0] elements >= 2.0
::exti == -1
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids in ::score[]
::score[] elements <= ::n_ids
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names in ::score[]
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds in ::score[]
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].num_bins elements > ::exti
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements > ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::exti in ::rounds[].bins->var
::rounds[].bins->var elements >= ::exti
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::rounds[].bins->dir elements > ::exti
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::order[] elements > ::exti
::score[] elements > ::debug
::gargc in ::score[]
::score[] elements > ::exti
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
==========
..mergesort()::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::n_ids == 1000
::n_names == 11
::n_rounds == 10

::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

7497.0, 3165.0, 88.0, 3675.0, 9148.0, 4506.0, 1349.0, 4460.0, 1862.0, 2181.0, 9759.0, 6082.0, 6391.0, 887.0, 4895.0, 5074.0, 66.0, 3768.0, 5624.0, 2879.0, 5570.0, 4253.0, NaN, 1170.0, NaN, 4928.0, 3972.0, 263.0, 6131.0, 3243.0, 3968.0, 5495.0, 8520.0, 1204.0, 9330.0, 3313.0, 5329.0, 8082.0, 6576.0, 9983.0, 5836.0, 7842.0, 2729.0, 9092.0, 1150.0, 5316.0, 4948.0, 5353.0, 7879.0, 5966.0, 3139.0, NaN, 1215.0, 3000.0, NaN, 5402.0, 8314.0, 5692.0, 2236.0, 5140.0, 2773.0, 1293.0, 1891.0, 2479.0, 2690.0, 1426.0, 2784.0, 104.0, 6815.0, 423.0, 7337.0, 8822.0, 9804.0, 1999.0, 8514.0, 7282.0, NaN, 8426.0, 5504.0, 605.0, 9321.0, 9126.0, 5935.0, 3984.0, 9825.0, NaN, NaN, 3610.0, 8321.0, 6730.0, 2323.0, 8243.0, 1454.0, 266.0, 413.0, 3319.0, 6700.0, 3483.0, NaN, 3858.0, NaN, 5279.0, 7489.0, 9326.0, 5429.0, 1108.0, 2881.0, 1177.0, 118.0, 8277.0, 9167.0, 9512.0, 6145.0, 8018.0, 2815.0, 6258.0, 9817.0, 5158.0, 6894.0, 4820.0, NaN, 6810.0, 3065.0, 6442.0, 5859.0, 8450.0, 9175.0, 5911.0, 2241.0, 4057.0, 9222.0, NaN, 2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0

::exti >= 0

base != null

size == 4

cmp != null

::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
nmemb > size
==========
..mergesort():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])

::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == return
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

::ignore_set has only one value
::ignore_set != null
::ignore_set[] == [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
::ignore_set[] elements == 0
::unknown_limit == 0.4
::debug == 0
::gargc == 4
::gargv != null
::gargv[] == [./marti-score]
::gargv[] elements == "./marti-score"
::tabula has only one value
::tabula != null
::tabula[] contains no nulls and has only one value, of length 1000
::tabula[] elements != null
::tabula[][0] == [4339.0, 7094.0, 5649.0, 9134.0, NaN, NaN, 6407.0, 3591.0, 3345.0, 7524.0, 8499.0, 255.0, 3868.0, 6849.0, 2687.0, 4037.0, 7926.0, 6679.0, NaN, NaN, 9104.0, 8245.0, 6668.0, 9905.0, 1113.0, 6745.0, 7828.0, 1840.0, 1096.0, 8668.0, 6718.0, 669.0, 6795.0, 1506.0, 3526.0, 7062.0, 1607.0, 6023.0, 615.0, 1708.0, 6033.0, 4168.0, 6423.0, 9043.0, 2.0, 3686.0, NaN, 7497.0, 3165.0, 88.0, 3675.0, 9148.0, 4506.0, 1349.0, 4460.0, 1862.0, 2181.0, 9759.0, 6082.0, 6391.0, 887.0, 4895.0, 5074.0, 66.0, 3768.0, 5624.0, 2879.0, 5570.0, 4253.0, NaN, 1170.0, NaN, 4928.0, 3972.0, 263.0, 6131.0, 3243.0, 3968.0, 5495.0, 8520.0, 1204.0, 9330.0, 3313.0, 5329.0, 8082.0, 6576.0, 9983.0, 5836.0, 7842.0, 2729.0, 9092.0, 1150.0, 5316.0, 4948.0, 5353.0, 7879.0, 5966.0, 3139.0, NaN, 1215.0, 3000.0, NaN, 5402.0, 8314.0, 5692.0, 2236.0, 5140.0, 2773.0, 1293.0, 1891.0, 2479.0, 2690.0, 1426.0, 2784.0, 104.0, 6815.0, 423.0, 7337.0, 8822.0, 9804.0, 1999.0, 8514.0, 7282.0, NaN, 8426.0, 5504.0, 605.0, 9321.0, 9126.0, 5935.0, 3984.0, 9825.0, NaN, NaN, 3610.0, 8321.0, 6730.0, 2323.0, 8243.0, 1454.0, 266.0, 413.0, 3319.0, 6700.0, 3483.0, NaN, 3858.0, NaN, 5279.0, 7489.0, 9326.0, 5429.0, 1108.0, 2881.0, 1177.0, 118.0, 8277.0, 9167.0, 9512.0, 6145.0, 8018.0, 2815.0, 6258.0, 9817.0, 5158.0, 6894.0, 4820.0, NaN, 6810.0, 3065.0, 6442.0, 5859.0, 8450.0, 9175.0, 5911.0, 2241.0, 4057.0, 9222.0, NaN, 2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN,

980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]
::tabula[][0] elements >= 2.0
::exti >= 0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
==========
..sort_examples():::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)

::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

::tabula[][0] == [4339.0, 7094.0, 5649.0, 9134.0, NaN, NaN, 6407.0, 3591.0, 3345.0, 7524.0, 8499.0, 255.0, 3868.0, 6849.0, 2687.0, 4037.0, 7926.0, 6679.0, NaN, NaN, 9104.0, 8245.0, 6668.0, 9905.0, 1113.0, 6745.0, 7828.0, 1840.0, 1096.0, 8668.0, 6718.0, 669.0, 6795.0, 1506.0, 3526.0, 7062.0, 1607.0, 6023.0, 615.0, 1708.0, 6033.0, 4168.0, 6423.0, 9043.0, 2.0, 3686.0, NaN, 7497.0, 3165.0, 88.0, 3675.0, 9148.0, 4506.0, 1349.0, 4460.0, 1862.0, 2181.0, 9759.0, 6082.0, 6391.0, 887.0, 4895.0, 5074.0, 66.0, 3768.0, 5624.0, 2879.0, 5570.0, 4253.0, NaN, 1170.0, NaN, 4928.0, 3972.0, 263.0, 6131.0, 3243.0, 3968.0, 5495.0, 8520.0, 1204.0, 9330.0, 3313.0, 5329.0, 8082.0, 6576.0, 9983.0, 5836.0, 7842.0, 2729.0, 9092.0, 1150.0, 5316.0, 4948.0, 5353.0, 7879.0, 5966.0, 3139.0, NaN, 1215.0, 3000.0, NaN, 5402.0, 8314.0, 5692.0, 2236.0, 5140.0, 2773.0, 1293.0, 1891.0, 2479.0, 2690.0, 1426.0, 2784.0, 104.0, 6815.0, 423.0, 7337.0, 8822.0, 9804.0, 1999.0, 8514.0, 7282.0, NaN, 8426.0, 5504.0, 605.0, 9321.0, 9126.0, 5935.0, 3984.0, 9825.0, NaN, NaN, 3610.0, 8321.0, 6730.0, 2323.0, 8243.0, 1454.0, 266.0, 413.0, 3319.0, 6700.0, 3483.0, NaN, 3858.0, NaN, 5279.0, 7489.0, 9326.0, 5429.0, 1108.0, 2881.0, 1177.0, 118.0, 8277.0, 9167.0, 9512.0, 6145.0, 8018.0, 2815.0, 6258.0, 9817.0, 5158.0, 6894.0, 4820.0, NaN, 6810.0, 3065.0, 6442.0, 5859.0, 8450.0, 9175.0, 5911.0, 2241.0, 4057.0, 9222.0, NaN, 2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0

::exti >= 0

base != null
size == 4
cmp != null
unknown_meth == 1
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].num_bins elements >= unknown_meth
unknown_meth in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->last elements > unknown_meth
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
unknown_meth in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
nmemb > size
==========
..sort_examples()::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].num_bins < orig(::order[]) (lexically)
::rounds[].num_bins <= orig(::order[]) (lexically)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)

::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->last > orig(::order[]) (lexically)
::rounds[].bins->last >= orig(::order[]) (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->var < orig(::order[]) (lexically)
::rounds[].bins->var <= orig(::order[]) (lexically)
::rounds[].bins->dir < ::order[] (lexically)
::rounds[].bins->dir <= ::order[] (lexically)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::score == orig(::score)
::score[] < orig(::order[]) (lexically)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] < orig(::order[]) (lexically)
::ignore_set[] <= orig(::order[]) (lexically)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == return
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

0, 0, 0,

2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
orig(::order[]) elements < ::n_ids
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_names in orig(::order[])
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::n_rounds in orig(::order[])
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].num_bins elements >= orig(unknown_meth)
orig(unknown_meth) in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->last elements > orig(unknown_meth)
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)

::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
orig(unknown_meth) in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::score[] <= orig(::order[]) (elementwise)
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
::debug in orig(::order[])
orig(::order[]) elements >= ::debug
::gargc in orig(::order[])
::exti in orig(::order[])
==========
merge.c.insertionsort():::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

0, 0, 0,

1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
a != null
a[] elements >= 0
n one of 4, 5
size == 4
cmp != null
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)

::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
n >= size
==========
merge.c.insertionsort():::EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]

::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
a[] elements >= 0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]

::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
==
merge.c.setup():::ENTER
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <
::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

0, 0, 0,

3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
list1 != null
list1[] elements >= 0
list2 != null
size == 4
cmp != null
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]
::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti

::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti
n > size
==
merge.c.setup()::.EXIT
::n_ids == orig(::n_ids)
::n_names == orig(::n_names)
::n_rounds == orig(::n_rounds)
::rounds == orig(::rounds)
::rounds[] == orig(::rounds[])
::rounds[].num_bins < ::order[] (lexically)
::rounds[].num_bins <= ::order[] (lexically)
::rounds[].num_bins == orig(::rounds[].num_bins)
::rounds[].bins == orig(::rounds[].bins)
::rounds[].bins[0] == orig(::rounds[].bins[0])
::rounds[].bins->first == ::ignore_set[]
::rounds[].bins->last > ::order[] (lexically)
::rounds[].bins->last >= ::order[] (lexically)
::rounds[].bins->last == orig(::rounds[].bins->last)
::rounds[].bins->var < ::order[] (lexically)
::rounds[].bins->var <= ::order[] (lexically)
::rounds[].bins->var == orig(::rounds[].bins->var)
::rounds[].bins->dir == orig(::rounds[].bins->dir)
::order == orig(::order)
::order[] > ::score[] (lexically)
::order[] > ::ignore_set[] (lexically)
::order[] >= ::ignore_set[] (lexically)
::order[] == orig(::order[])
::score == orig(::score)
::score[] == orig(::score[])
::ignore_set == orig(::ignore_set)
::ignore_set[] == orig(::rounds[].bins->first)
::ignore_set[] == orig(::ignore_set[])
::unknown_limit == orig(::unknown_limit)
::debug == orig(::debug)
::gargc == orig(::gargc)
::gargv == orig(::gargv)
::tabula == orig(::tabula)
::tabula[] == orig(::tabula[])
::tabula[][0] == orig(::tabula[][0])
::exti == orig(::exti)
::n_ids == 1000
::n_names == 11
::n_rounds == 10
::rounds has only one value
::rounds != null
::rounds[] contains no nulls and has only one value, of length 10
::rounds[] elements != null
::rounds[].num_bins == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
::rounds[].num_bins elements >= 1
::rounds[].num_bins sorted by <

::rounds[].bins contains no nulls and has only one value, of length 10
::rounds[].bins elements != null
::rounds[].bins[0] contains no nulls and has only one value, of length 10
::rounds[].bins[0] elements != null
::rounds[].bins->last == [999, 289, 139, 79, 19, 19, 19, 19, 19, 9]
::rounds[].bins->last sorted by >=
::rounds[].bins->var == [2, 8, 9, 7, -1, -1, -1, -1, -1, -1]
::rounds[].bins->var elements != 0
::rounds[].bins->var elements >= -1
::rounds[].bins->dir == [100, 97, 97, 97, 0, 0, 0, 0, 0, 0]
::rounds[].bins->dir elements one of 0, 97, 100
::rounds[].bins->dir sorted by >=
::order has only one value
::order != null
::order[] elements >= 0
::score has only one value
::score != null
::score[] == [0, 0, 0, 0, 0,

9817.0, 5158.0, 6894.0, 4820.0, NaN, 6810.0, 3065.0, 6442.0, 5859.0, 8450.0, 9175.0, 5911.0, 2241.0, 4057.0, 9222.0, NaN, 2769.0, 4033.0, 7642.0, NaN, 7648.0, 1340.0, 9660.0, 9260.0, 1912.0, 8445.0, 5973.0, 3012.0, 2849.0, 1609.0, 3986.0, 8581.0, 901.0, 7010.0, 8944.0, 7835.0, 4402.0, 7731.0, 1616.0, NaN, 2764.0, 5885.0, 9817.0, 440.0, 951.0, NaN, 6506.0, NaN, 7902.0, 7733.0, 7364.0, 2205.0, 1046.0, 3140.0, 8692.0, 5005.0, 9315.0, 5998.0, 4757.0, 8673.0, 4303.0, NaN, 9700.0, 9445.0, 8440.0, 3378.0, NaN, 1432.0, NaN, 6487.0, NaN, 770.0, 9433.0, 1424.0, 9813.0, 6191.0, 1928.0, 5365.0, 7623.0, 5178.0, 853.0, 3706.0, 3830.0, 6365.0, 4065.0, 246.0, 8493.0, NaN, NaN, 2401.0, 5145.0, 542.0, 8648.0, 5286.0, 8357.0, 874.0, 6605.0, 4405.0, 1628.0, NaN, 1192.0, 244.0, 8641.0, 3526.0, 8748.0, 2415.0, 2239.0, 5665.0, 2440.0, 6058.0, 6988.0, 1827.0, 1865.0, 9537.0, 6234.0, 3719.0, 2854.0, 480.0, NaN, 1502.0, 2468.0, 8794.0, 3177.0, 2374.0, 6588.0, 4508.0, 739.0, 3632.0, NaN, NaN, NaN, 7849.0, 1651.0, 7381.0, 6737.0, 6158.0, 9295.0, NaN, 4790.0, 4250.0, 7664.0, 4194.0, 9441.0, 2724.0, 6240.0, 9318.0, 8143.0, 6825.0, 8738.0, 7928.0, 5545.0, 4911.0, 741.0, 5847.0, 1763.0, 8145.0, 4154.0, 7513.0, 9625.0, 9248.0, NaN, 3783.0, 4039.0, 8546.0, 735.0, 7536.0, 6898.0, 2841.0, NaN, 859.0, 8739.0, 3061.0, NaN, 1719.0, 6180.0, 3711.0, 9854.0, 3212.0, 445.0, 688.0, 3766.0, 2614.0, 7551.0, 2126.0, 7281.0, 7032.0, 5139.0, 7631.0, 3094.0, 6058.0, NaN, 6668.0, 5336.0, 7198.0, 5944.0, NaN, 9037.0, 8142.0, 2005.0, 6479.0, 1809.0, 5234.0, 1097.0, 3392.0, 3025.0, 3691.0, NaN, 27.0, 3108.0, 4542.0, 4640.0, 9544.0, 4922.0, 843.0, NaN, 2955.0, 5741.0, 6929.0, 8132.0, 9154.0, 8591.0, 7200.0, 6081.0, 2883.0, 2327.0, 7096.0, 4062.0, 5146.0, 6229.0, 7413.0, 5516.0, 8578.0, 8761.0, 4226.0, NaN, 5682.0, 6780.0, 7399.0, 5588.0, 7147.0, 2930.0, 6791.0, 7046.0, NaN, 1035.0, 5180.0, 8011.0, 7526.0, 4629.0, NaN, 2189.0, 4974.0, 8487.0, 393.0, 2717.0, 6157.0, 4303.0, 8597.0, 3801.0, 6747.0, NaN, 6987.0, 6992.0, 8065.0, 7237.0, 5346.0, 5288.0, 8614.0, 5093.0, 2487.0, 4741.0, 1212.0, NaN, 411.0, 2276.0, 3695.0, 4078.0, 7090.0, 7775.0, 1630.0, 2778.0, 5355.0, 1712.0, 8299.0, 5261.0, 5066.0, 4935.0, 1024.0, NaN, NaN, 2405.0, 2772.0, 2103.0, 7503.0, 5455.0, 7037.0, NaN, 3730.0, NaN, 8547.0, 1861.0, 8613.0, 8250.0, 4114.0, 6860.0, 6648.0, 98.0, 7343.0, 1488.0, NaN, 8081.0, 5081.0, 8462.0, NaN, 342.0, 1358.0, 624.0, 387.0, 9566.0, 6439.0, 5154.0, 8716.0, 7264.0, 2962.0, 6130.0, NaN, 7968.0, NaN, NaN, 1692.0, 3567.0, 9796.0, 1879.0, 7822.0, 8102.0, 2952.0, 7647.0, 1386.0, 7187.0, 906.0, 1571.0, 5362.0, 3310.0, 8367.0, 2716.0, 9207.0, 941.0, 2704.0, 806.0, NaN, 8370.0, 781.0, NaN, 3636.0, 6128.0, 4349.0, 2205.0, 1966.0, 852.0, 8597.0, 8115.0, 8490.0, 1028.0, 202.0, 7018.0, 6196.0, 7058.0, 823.0, 244.0, 3545.0, 3612.0, 108.0, 9404.0, 3831.0, 1846.0, 4037.0, 2069.0, 9829.0, 7713.0, 2997.0, 1752.0, 5689.0, 4901.0, 4540.0, 7079.0, 9559.0, 2755.0, 9150.0, 9335.0, 5232.0, 7458.0, 9013.0, 1171.0, 8683.0, 8817.0, 3733.0, 179.0, 8398.0, 3954.0, 7339.0, 9051.0, 3893.0, 3173.0, 6636.0, 2474.0, 6962.0, 4964.0, 5673.0, 3695.0, 6880.0, 2696.0, NaN, 667.0, 7157.0, 6187.0, 3354.0, 784.0, 5641.0, 5287.0, 2047.0, 5180.0, 9748.0, 9334.0, 2366.0, 1504.0, 6118.0, 9166.0, 2448.0, 7366.0, NaN, 1500.0, 8452.0, 8210.0, 4516.0, 7675.0, 2998.0, 4633.0, 5974.0, 8844.0, 4805.0, 1431.0, 8524.0, 4163.0, 2836.0, 3942.0, NaN, 6850.0, 660.0, 5664.0, 4109.0, 9572.0, 7766.0, 3352.0, 5803.0, 5551.0, 5815.0, NaN, 7730.0, 7676.0, 4137.0, 9752.0, NaN, NaN, 3937.0, 5570.0, 8351.0, 6658.0, 6862.0, 3848.0, NaN, 6760.0, 764.0, 9628.0, 9953.0, 6001.0, 1122.0, 6218.0, 722.0, NaN, 9169.0, 2451.0, 4216.0, NaN, 2640.0, 7413.0, 752.0, 726.0, 3974.0, 2052.0, 3630.0, 7934.0, 2315.0, 2584.0, 769.0, 6691.0, 7336.0, 8326.0, 1297.0, 7267.0, 877.0, 4552.0, 4115.0, 8486.0, 3415.0, 1781.0, NaN, 6098.0, 3834.0, 2679.0, 5912.0, 5619.0, 7643.0, 5932.0, 9909.0, 277.0, 8872.0, NaN, 7828.0, 451.0, 1083.0, 4341.0, 208.0, 8689.0, 233.0, 5276.0, 7639.0, 1787.0, 9882.0, 9544.0, 8676.0, 1174.0, 3988.0, 2933.0, 7049.0, 7441.0, 8641.0, 937.0, 4848.0, 1699.0, 5187.0, 9871.0, 7171.0, 9535.0, 9313.0, 6413.0, 2255.0, 4578.0, NaN, 5051.0, 9845.0, 7650.0, 914.0, 1577.0, 1113.0, 2364.0, 4288.0, 8304.0, 2917.0, 8023.0, 9980.0, 2312.0, 8316.0, 5693.0, 2613.0, 1528.0, 5784.0, 4891.0, 7065.0, 3613.0, 7798.0, 6548.0, 4151.0, 9987.0, 9675.0, 9531.0, 1940.0, 6878.0, 4646.0, 8091.0, 5978.0, 5554.0, NaN, 131.0, 7401.0, 2344.0, 82.0, NaN, 1863.0, 220.0, 5453.0, 8674.0, 5978.0, 1350.0, 8931.0, 4488.0, 1412.0, 6821.0, 4361.0, 8575.0, 4660.0, 8497.0, 3272.0, 227.0, 5615.0, 4253.0, 9492.0, 615.0, 2784.0, 8620.0, 7940.0, 8995.0, 7821.0, 4526.0, 1878.0, 8403.0, 5259.0, 8359.0, 7321.0, 5223.0, 8597.0, 2993.0, 9200.0, NaN, 1157.0, 2584.0, 5873.0, 6998.0, 3087.0, 3118.0, 3926.0, 2244.0, NaN, 4932.0, NaN, 5778.0, 694.0, 8616.0, 3821.0, 285.0, 974.0, 5685.0, NaN, 980.0, 7583.0, 4832.0, NaN, 8738.0, 3428.0, 607.0, NaN, 2245.0, 1033.0, 6872.0, 4349.0, 4046.0, 3187.0, 7831.0, 2009.0, 2614.0, 8711.0, 2242.0, 7109.0, NaN, 9103.0, 1664.0, 98.0, 5226.0, 1497.0, NaN, 6411.0, 6271.0, 2524.0, 1043.0, 255.0, 7085.0, 982.0, 3968.0, 4389.0, 2342.0, 6930.0, 8002.0, 5923.0, 6405.0, 4272.0, 9435.0, 2476.0, 1362.0, 3330.0, 2087.0, 3142.0, 5444.0, NaN, 1358.0, NaN, NaN, 1500.0, 314.0, 6367.0, 3333.0, 4131.0, 6629.0, 1398.0, 5603.0, 1457.0, 1267.0, 5320.0, 1100.0, 8764.0, 3665.0, 7699.0, 9875.0, NaN, 3996.0, 4309.0, 3147.0, 8732.0, 503.0, 8464.0, 9471.0, 1600.0, 9886.0, 1826.0, 4955.0, 4210.0, 1946.0, NaN, 3107.0, 7130.0, 9478.0, 2485.0, 4168.0, NaN, 6802.0, NaN, 3787.0, 1067.0, 887.0, NaN, 5404.0, 4228.0, 4507.0, 6619.0, 1957.0, 4558.0, 932.0, NaN, 8661.0, 9360.0, 2348.0, NaN, 2350.0, 4237.0, 9854.0, 4937.0, 1176.0, 7379.0, 2931.0, 103.0, 7893.0, 6099.0, NaN, 3975.0, 5912.0, 2951.0, 7562.0, 4358.0, 7417.0, 5344.0, 5177.0, 341.0, 5345.0, 8827.0, 615.0, 2816.0, NaN, 3367.0, NaN, NaN, 1522.0, 1504.0, 3472.0, 6329.0, 8130.0, 3834.0, 6065.0, 8493.0, 2088.0, 4796.0, 2866.0, 99.0, 5810.0, 4216.0, NaN, 5482.0, 7878.0, 3776.0, 8426.0, 1993.0, 4713.0, 9541.0, 3150.0, 33.0, 9913.0, 6726.0, 2068.0, 5031.0, 7310.0, 9219.0, 5210.0, 3748.0, 6101.0, 2871.0, 1588.0, 9180.0, 6593.0, 243.0, 6713.0, 8838.0, 1448.0, 2680.0, 7937.0, 4161.0, 4155.0, 6253.0, 7379.0, 9409.0, 1885.0, 344.0, 6625.0, 4061.0, 1482.0, 4937.0, 4663.0, 8000.0, 8546.0, 3823.0, 414.0]

::tabula[][0] elements >= 2.0
::exti >= 0
list1[] elements >= 0
::rounds[].num_bins elements < ::n_ids
::rounds[].bins->last elements < ::n_ids
::rounds[].bins->var elements < ::n_ids
::rounds[].bins->dir elements < ::n_ids
::order[] elements < ::n_ids
::n_ids > ::exti
::rounds[].num_bins elements < ::n_names
::rounds[].bins->var elements < ::n_names
::n_names in ::order[]

::n_names > ::exti
::n_rounds in ::rounds[].num_bins
::rounds[].num_bins elements <= ::n_rounds
::rounds[].bins->var elements < ::n_rounds
::n_rounds in ::order[]
::n_rounds > ::exti
::rounds[].num_bins > ::ignore_set[] (elementwise)
::rounds[].num_bins elements > ::debug
::gargc in ::rounds[].num_bins
::rounds[].bins->last > ::rounds[].bins->var (elementwise)
::rounds[].bins->last > ::ignore_set[] (elementwise)
::rounds[].bins->last elements > ::debug
::rounds[].bins->last elements > ::gargc
::rounds[].bins->last elements >= ::exti
::rounds[].bins->var < ::rounds[].bins->dir (elementwise)
::rounds[].bins->dir >= ::ignore_set[] (elementwise)
::debug in ::rounds[].bins->dir
::rounds[].bins->dir elements >= ::debug
::order[] >= ::score[] (elementwise)
::debug in ::order[]
::order[] elements >= ::debug
::gargc in ::order[]
::exti in ::order[]
::debug in ::score[]
::score[] elements == ::debug
::debug in ::ignore_set[]
::ignore_set[] elements == ::debug
::debug <= ::exti