Source-code plagiarism: A UK academic perspective

Georgina Cosma and Mike Joy

May 2006

Department of Computer Science, The University of Warwick

Research Report No. 422.

ABSTRACT

In computing courses, students are often required to complete tutorial and laboratory exercises asking them to produce source-code. Academics may require students to submit source-code produced as part of such exercises in order to monitor their students' understanding of the material taught on that module, and submitted source-code may be checked for similarities in order to identify instances of plagiarism. In exercises that require students to work individually, source-code plagiarism can occur between students or students may plagiarise by copying material from a book or from other sources.

We have conducted a survey of UK academics who teach programming on computing courses, in order to establish what is understood to constitute source-code plagiarism in an undergraduate context. In this report, we analyse the responses received from 59 academics.

This report presents a detailed description of what can constitute source-code plagiarism from the perspective of academics who teach programming on computing courses, based on the responses to the survey.

Keywords: Plagiarism, Source-code plagiarism description, Source-code plagiarism survey.

Contents

FIGURES

TABLES

1.0 Introduction

Plagiarism in programming assignments is an inevitable issue for most academics teaching programming. The Internet, the rising number of essay banks, and text-books are common sources used by students to obtain material, and these facilities make it easier for students to plagiarise. A recent article revealed that some students use the internet to hire expert coders to implement their programming assignments (Gomes 2006).

Bull *et al.* (2001) and Culwin *et al.* (2001) have carried out surveys on academics to determine the prevalence of plagiarism and have evaluated the performance of free-text plagiarism detection software and source-code plagiarism detection software respectively. The surveys have shown that both free-text and source-code plagiarism are significant problems in academic institutions, and the study by Bull *et al.* (2001), indicated that 50% of the 293 academics that participated in their survey felt that in recent years there has been an increase in plagiarism. Many software tools have been developed for detecting source-code plagiarism, the most popular being Plague (Whale, 1990a), YAP3 (Wise, 1996), and JPlag (Prechelt *et al.* 2000, 2002).

A review of the current literature on source-code plagiarism in student assignments reveals that there is no commonly agreed description of what constitutes source-code plagiarism from the perspective of academics who teach programming on computer courses. Some definitions on source-code plagiarism exist, but these appear to be very limited. For example, according to Faidhi and Robinson (1987), plagiarism occurs when programming assignments are *"copied and transformed"* with very little effort from the students, whereas Joy and Luck (1999) define plagiarism as *"unacknowledged copying of documents and programs."*

Furthermore, a book on academic misconduct written by Decoo (2002), discusses various issues surrounding academic plagiarism. Decoo briefly discusses software plagiarism and the level of user-interface, content and source-code.

Sutherland-Smith (2005) carried out a survey to gather the views of 11 academics in the faculty of Business and Law at South-Coast University in Australia. The findings revealed varied perceptions on plagiarism between academics teaching the same subject, and the author suggests that a *"collaborative, cross-disciplinary re-thinking of plagiarism is needed."*

In order to establish what is understood to constitute source-code plagiarism in an undergraduate context we have carried out a survey that comprised questionnaires on this issue. In this report we analyse the responses gathered from the survey and suggest a detailed description on what can constitute source-code plagiarism from the perspective of academics who teach programming on computing courses. We also present findings concerned with factors that could influence the decision of academics as to whether plagiarism occurred between fragments of source-code and similarities that could be considered as evidence of possible plagiarism in source-code.

2.0 History of source-code plagiarism detection systems

The literature classifies source-code plagiarism detection software into two types: attribute-counting systems and structure-metric systems. The first known plagiarism detection system was an attribute counting program developed by Ottenstein (1976a, b) for detecting identical and nearly-identical student work. The program used Halstead's software science metrics to detect plagiarism by counting operators and operands for ANSI-FORTRAN modules. The metrics suggested by Halstead (1972, 1977) were:

- *n1* = number of unique operators
- *n2* = number of unique operands
- *N1* = total number of operators
- *N2* = total number of operands

Program pairs were considered as plagiarisms if all four values were identical.

Robinson and Soffa (1980) developed a plagiarism detection program that combined new metrics with Halstead's metrics in order to improve plagiarism detection. Their system, called ITPAD, consisted of three steps: lexical analysis, analysis of program structure for characteristics, and analysis of the characteristics.

The system by Robinson and Soffa (1980) breaks each program into blocks and builds two graphs to represent the structure of each student's program. It then generates a list of attributes based on the lexical and structural analysis and compares pairs of programs by counting these characteristics.

Structure metric systems were initially proposed by Donaldson *et al.* (1981). These systems use attribute-counting metrics but they also compare the program structure in order to improve plagiarism detection. Donaldson *et al.* (1981) identified simple techniques that novice programming students use to disguise plagiarism - *"The problem of students handing in programs which are not their own…."* These methods are: renaming variables, reordering statements that will not affect the program result, changing format statements, and breaking up statements such as multiple declarations and output statements (Donaldson *et al.* 1981).

The program developed by Donaldson *et al.* (1981) scans source-code files and stores information about certain types of statements. Then, statement types significant in describing the structure are assigned a single code character. Each assignment is then represented as a string of characters. If the string representations are identical or similar, then the pair of programs is returned as similar. The most well known recent structure metric systems are YAP3 (Wise, 1996), Plague (Whale, 1990a), and JPlag (Prechelt *et al.* 2000, 2002).

Plague first creates a sequence of tokens for each file, and structure files comprising of structure metrics. Then the structure profiles are compared and similar profiles are matched. It then finds and compares the token sequences of similar files. Wise (1996) developed a token matching algorithm called Running-Karp-Rabin Greedy-String-Tiling algorithm (RKS-GST) and used it in Yap3. This algorithm was developed mainly to detect breaking of code functions into multiple functions (and vice-versa) and to detect the reordering of independent code segments.

JPlag's (Prechelt *et al.* 2001) comparison algorithm is a token-string based algorithm combined with the "Greedy String Algorithm" that was introduced by Wise (1996). Basically, JPlag parses files, converts them into token strings and then applies the "Greedy String Algorithm" to find similarities between the files (Prechelt *et al.* 2000, 2002).

In most structure metric systems, including the ones mentioned above, the first stage is called tokenisation. At the tokenisation stage different parts of the code are replaced by a predefined and consistent token, for example different types of loops in the source-code may

be replaced by the same token name regardless whether their loop type (e.g. while loop, for loop). Each source-code document is then represented as a series of token strings. The tokens of each document are compared to determine similar source-code segments.

Comparisons of attribute-counting and structure-metric systems have shown that attribute counting methods alone are not adequate enough for detecting plagiarism (Whale, 1990a, 1990b; Verco & Wise 1996a, 1996b).

3.0 Survey Methodology and Ethics

An on-line questionnaire was distributed to a list of academics supplied by the Higher Education Academy Subject Centre for Information and Computing Sciences (HEA-ICS). The mailing list consisted of 120 names, many of whom can be assumed to have expertise in teaching programming. The people on the list were contacted in November 2005 by e-mail asking them to complete the questionnaire. Furthermore, the instructions for the survey specified that only academics who are currently teaching (or have previously taught) at least one programming subject should respond.

The survey was anonymous, but included a section in which the academics could optionally provide personal information. Of 59 responses, the 43 who provided the name of their academic institution were employed at 37 departments in 34 different institutions, of which 31 were English universities and three were Scottish universities.

The questionnaire contained mostly closed questions requiring multiple-choice responses. The questions were in the form of small scenarios describing various ways students have obtained, used, and acknowledged material. The respondents were required to select from a choice of responses the type of academic offence (if any) that in their opinion applied to each scenario. A comments box was placed below each question in order for academics to provide any comments they have about issues surrounding the question asked. It was very important to gather the comments of academics on the various issues regarding plagiarism due to the variety of academic regulations and the academics' opinions on such a sensitive issue.

It was not the purpose of this survey to address in depth subjective issues, such as plagiarism intent and plagiarism penalties, that could depend on student circumstances and university policies.

The instructions for completing the survey emphasised that all the information provided by the respondents is to be kept confidential. Further, neither the respondents nor their institution will be identified in the thesis or in any report or publication based on this research. The survey is included in Appendix A.

The SPSS (Statistical Package for the Social Sciences) and Microsoft Excel software were both used for creating charts. The SPSS software was used to perform statistical tests on the data collected from the survey.

4.0 Survey Part One

Part one of the report is concerned with an analysis of the issue of what constitutes source-code plagiarism from the perspective of academics. The questions analysed in this section are on the subject of source-code use and acknowledgement. These questions were in the form of small scenarios describing different ways students have used and acknowledged material from sources such as books. The respondents were required to select the type of academic offence (from a choice of responses) that in their opinion applies to each scenario.

4.1 Question 1

The process of proving whether source-code plagiarism occurred may involve looking at other parts of a programming assignment since in some circumstances source-code alone may not be sufficient for identifying and proving plagiarism. A programming assignment may include design diagrams, source-code and other documentation. Question 1 is shown in *Figure 1*.

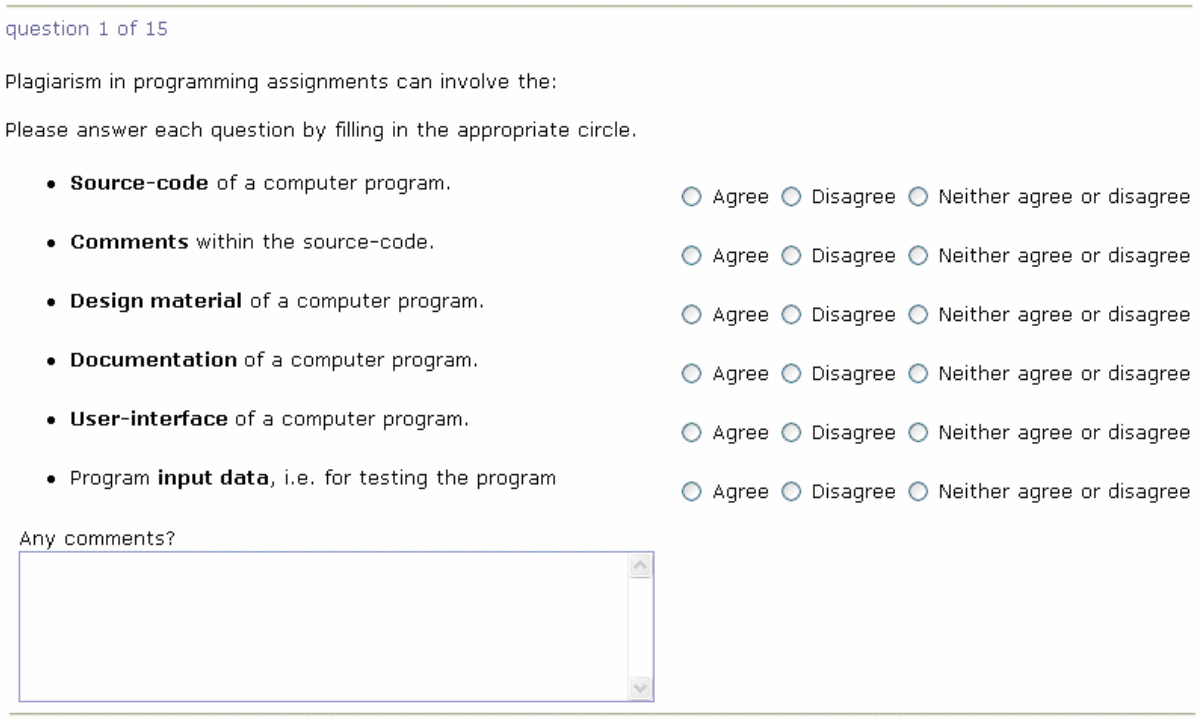


Figure 1: Question 1

The results of question 1 are presented in *Figure 2* below. The figure below corresponds to the following:

A. Source-code of a computer program
B. Comments within the source-code
C. Design material of a computer program
D. Documentation of a computer program
E. User interface of a computer program
F. Program input data, i.e. for testing the program

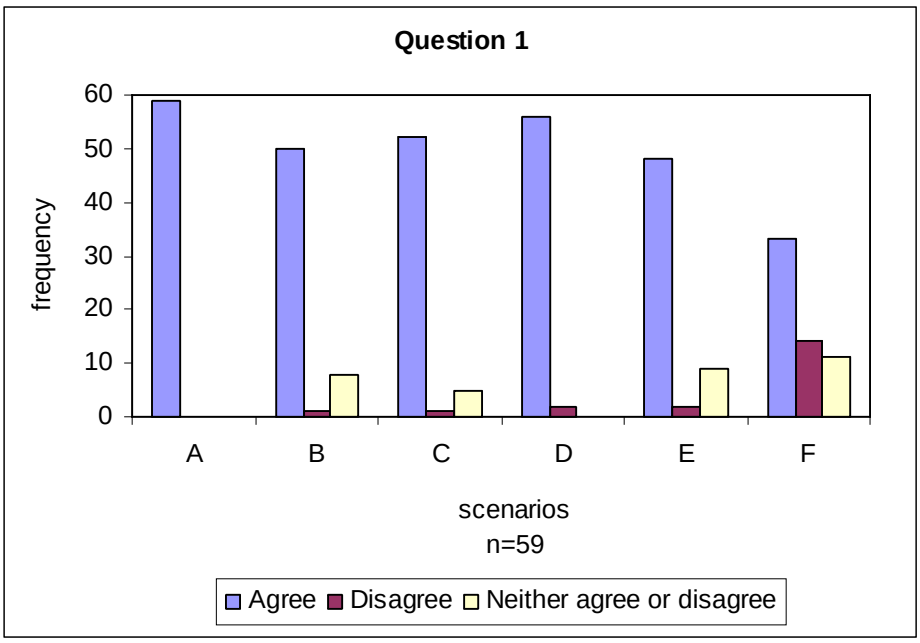


Figure 2: Chart shows the results from the responses to Question 1

All academics agreed that in a programming assignment source-code can be plagiarised. Regarding 'comments within a source-code', the majority of the academics agree that comments can be plagiarised and that comments may also help identify source-code plagiarism cases, "*comments are not in themselves a 'code plagiarism' issue but may help identify a case.*"

Regarding whether 'program input data' can be subject to plagiarism, the academic responses somehow varied. The majority of academics, 40 out of 59, agreed that program input data can be involved in plagiarism. On this issue, academics commented that program input alone cannot contribute to the identification of plagiarism and whether program input can be part of plagiarism depends on the assignment specification.

Furthermore, three academics commented that copying input data is an issue if students are assessed on the testing strategies they have used to test their program. In this case, assessment for plagiarism would be done by observing the testing strategy including the datasets used for testing the program, and the test plan (User Acceptance Testing material, i.e., test plan, system design documentation, technical documentation, user manuals, etc). One academic specified:

> *"The input data are not necessarily plagiarism, but a testing strategy can be (i.e. what and how to perform the testing, datasets used, any UAT forms, etc.) The UI is not necessarily generated by the user, but by the development tool, so it cannot be considered plagiarism in this case."*

Academics commented that whether a user-interface can be subject to plagiarism depends on the assignment requirements, and if a user-interface is required from the students then it can be subject to plagiarism, "*The item on user-interface is really dependent on the task: if the UI was not the essence of the program, then I wouldn't think this is an issue.*"

The majority of academics agreed that any material that is unreferenced can constitute plagiarism. One academic observed:

> *"I require students to write their own code and documentation, except that there are safe-harbour provisions for permitted sources of each. They must cite their use to be*

able to claim the safe-harbour provision. The permitted sources are explicitly identified for the project."

Although the majority of academics agreed that any unreferenced source-code can constitute plagiarism, one academic expressed some uncertainties as to whether code re-use can constitute plagiarism,

> *"I agree that plagiarism could include those areas mentioned above [refers to the choices in question 1], but in o-o environments where re-use is encouraged, obviously elements of re-use are not automatically plagiarism. I think I'd be clear on the boundaries and limits in any given circumstance, and would hope to be able to communicate that clarity to my students, but obviously there will potentially be problems."*

4.2 Question 2

Academics were presented with four small scenarios on copying, adapting, converting source-code from one programming language to another, and using software for automatically generating source-code. The question is shown in *Figure 3*.

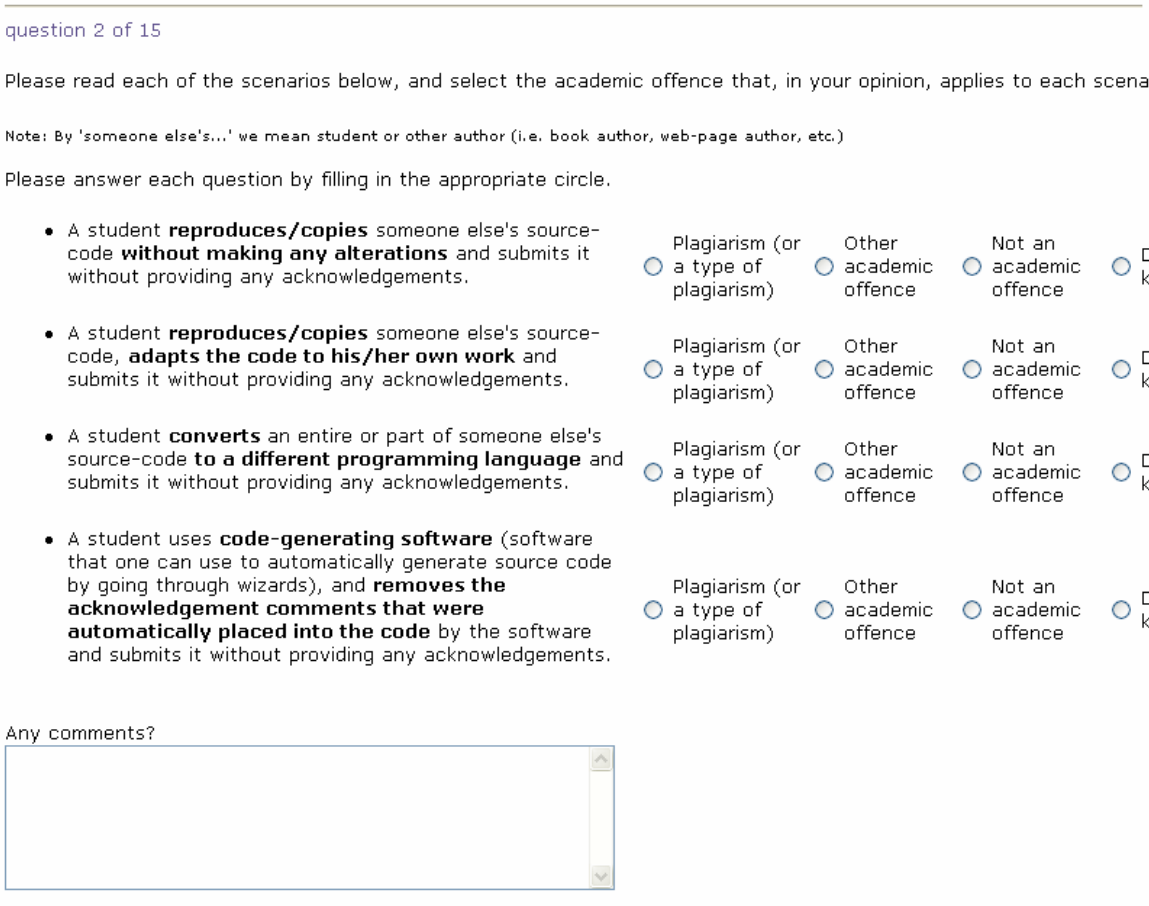


Figure 3: Question 2

The results of question 2 are presented in *Figure 4* below which corresponds to the following:

A. A student reproduces/copies someone else's source-code without making any alterations and submits it without providing any acknowledgements.
B. A student reproduces/copies someone else's source-code, adapts the code to his/her own work and submits it without providing any acknowledgements.
C. A student converts an entire or part of someone else's source-code to a different programming language and submits it without providing any acknowledgements.
D. A student uses code-generating software (software that one can use to automatically generate source-code by going through wizards), and removes the acknowledgement comments that were automatically placed into the code by the software and submits it without providing any acknowledgements.

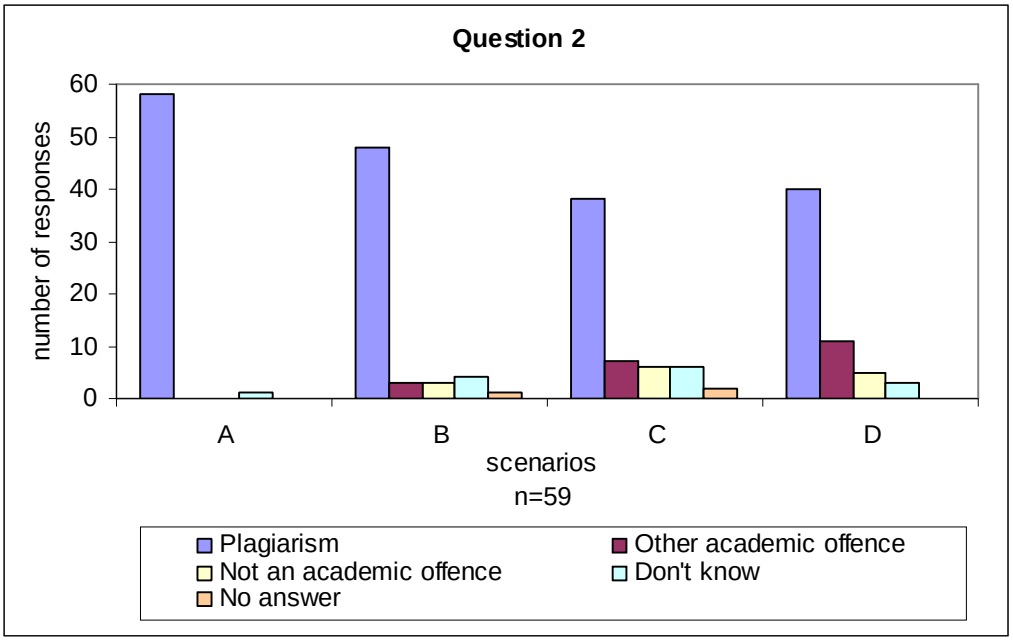


Figure 4: Chart shows the results from the responses for Question 2

For the first scenario, 'a student reproduces/copies someone else's source-code without making any alterations and submits it without providing any acknowledgements', there was a wide agreement between academics (58 out of 59) that this scenario constitutes plagiarism. One academic provided a 'don't know' response justified by the following comment.

> *"Please see comments on code re-use [refers to the comment provided in question 1 - In O-O environments where re-use is encouraged, obviously elements of re-use are not automatically plagiarism. I think I'd be clear on the boundaries and limits in any given circumstance, and would hope to be able to communicate that clarity to my students, but obviously there will potentially be problems]. Use of the API would be legitimate without acknowledgement – or with only the implicit acknowledgement."*

For scenario B, 'A student reproduces/copies someone else's source-code, adapts the code to his/her own work and submits it without providing any acknowledgements' the majority of the academics agreed that this action can constitute plagiarism. Many academics commented that whether this scenario constitutes plagiarism depends on the *degree of*

adaptation of the source-code, i.e., *how much* code is a copy of someone else's work and the extent to which that code has been adapted without acknowledgement. One academic provided a 'don't know' justified by the following comment:

> "*[The second scenario] depends on the adaptation. If the student changes the original beyond all recognition so that there was nothing left of the original to acknowledge, then it wouldn't be plagiarism. Or if the original was part of some material provided in lectures, then the acknowledgement wouldn't be needed, it would almost be taking it as "implicit", particularly if the lecturer encouraged the students to start with the provided code and then adapt it. If the adaptation involves just changing a few variable names (e.g.) then that would be plagiarism."*

However, two respondents were not clear on these points. The first, who did not provide an answer, noted that there were "*some grey areas here as software reuse is often encouraged."* The second regarded the scenario as not being an academic offence, commenting:

> *"This is difficult – as code copied from a website that assists in a specific task is potentially good practice. However, code that is a 100% copy is a different issue. I would also be concerned about the context of this copying. If the only deliverable were to be code and documentation the offence is clear. In this sense I suppose it is an issue of how much of the overall assignment is actually a copy of other work (without acknowledgement)."*

For scenario C, 'A student converts an entire or part of someone else's source-code to a different programming language, and submits it without providing any acknowledgements', several academics remarked that if the code is converted automatically without any or much effort from the student then this can constitute plagiarism. However, if a student takes the ideas or inspirations from code written in another programming language, and creates the source-code entirely *"from scratch"*, then this is not likely to constitute plagiarism.

Furthermore, in their comments academics pointed out that whether the conversion constitutes plagiarism depends on the programming languages, i.e., taking source-code written in one programming language and converting it to a similar programming language can constitute plagiarism, such as from C++ to Java, because the languages are too similar. However, converting from Prolog to C or Java can still constitute plagiarism depending on the amount of work involved in the conversion. In addition, one academic who responded 'don't know', observed:

> *"The key question is whether the student is being misleading about how much work is theirs or not. I can imagine examples where the translation was definitely plagiarism, and I can imagine examples where the student has taken legitimate inspiration from someone else's example code, and has rewritten it in a different language."*

A code-generator is an application that takes as input meta-data (i.e. a database schema) and creates source-code that is compliant with design patterns. An example of shareware code-generator software is JSPMaker (2006), which, given a database, this software quickly and easily creates complete source-code and a full set of JavaServer Pages (2006) that have database connectivity.

We asked whether it constitutes plagiarism if 'a student uses code-generating software, removes the acknowledgement comments that were automatically placed into the

code by the software, and submits it without providing any acknowledgements.' Academics commented that this scenario may constitute plagiarism if the assignment specification instructs students to write the source-code themselves without the use of such software, or it may not constitute plagiarism if permission for use of code-generating software is described in an assignment specification. The majority of the academics considered *unacknowledged* use of such software as plagiarism. One academic who considered this scenario to be 'plagiarism' provided the following comment:

> *"In each case there must be some presumed benefit to the student in doing so (why did they do it otherwise?) and disruption to the assessment system. Even where the advantage might be minimal – e.g. from Prolog to C – a good student would almost certainly acknowledge the issue and use it to discuss the differences."*

The findings suggest that whether or not source-code reuse is allowed in programming assignments, students should always indicate which parts of the source-code were not authored by them, and that using material created by other persons or by software *without providing acknowledgement* can constitute plagiarism.

In conclusion, students should be required to acknowledge any material they use that is not their own original work regardless of the licensing permissions of that material (e.g. open source, free-use, fair-use), and as one academic commented, "*I require the students to acknowledge their dependence on these sources of code even when it is permitted."*

4.3 Question 3

There are several ways students can gain source-code written by another author and present that material as their own work. In this question, the academic is asked to provide his/her opinion as to which academic offence applies to given scenarios. The scenarios describe methods used by students to gain material and present that material as their own work. The purpose of this question is to determine which student actions indicate to source-code plagiarism. Question 3 is shown in *Figure 5*, and the responses of academics for are shown in *figure 6*.

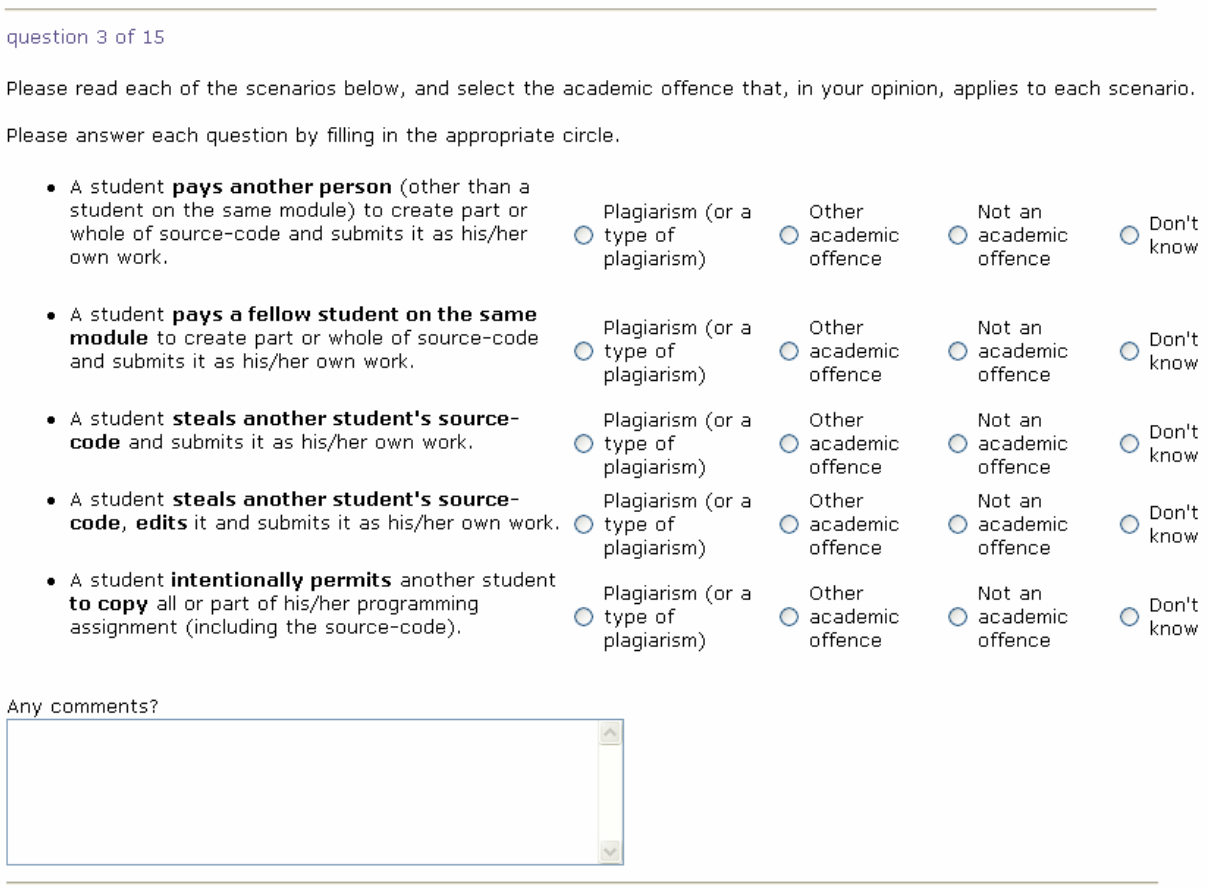


Figure 5: Question 3

Figure 6 corresponds to the following:

A. A student pays another person (other than a student on the same module) to create part or whole of source-code and submits it as his/her own work.
B. A student pays a fellow student on the same module to create part or whole of source-code and submits it as his/her own work.
C. A student steals another student's source-code and submits it as his/her own work.
D. A student steals another student's source-code, edits it and submits it as his/her own work.
E. A student intentionally permits another student to copy all or part of his/her programming assignment (including the source-code).

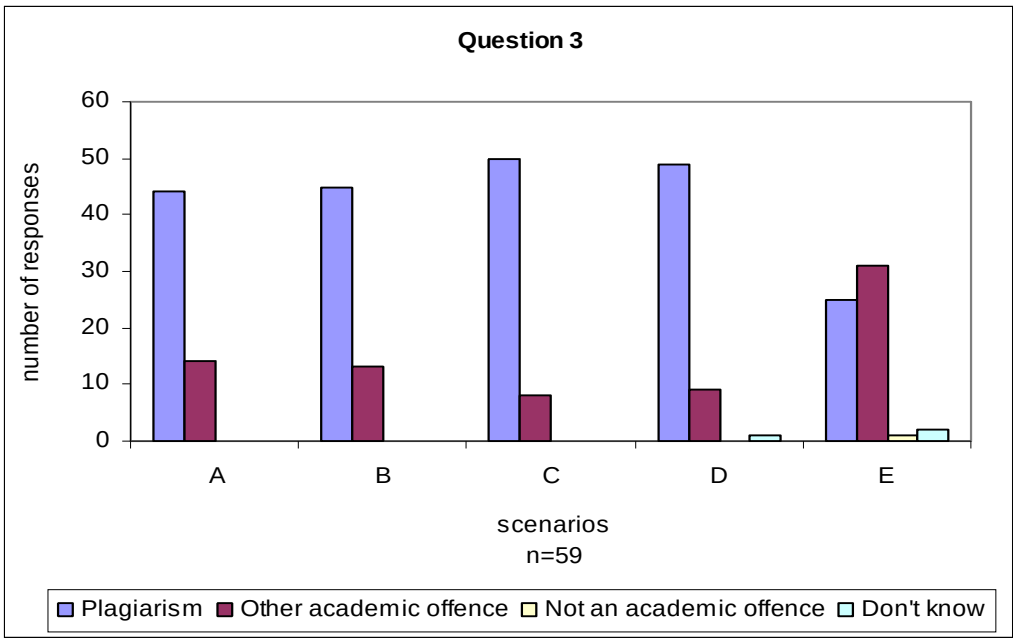


Figure 6: Chart shows the results from the responses for Question 3

In scenarios A to D, plagiarism was committed alongside another academic offence (s) such as cheating and stealing, hence some academics chose 'plagiarism' and others 'other academic offence'. Many academics commented that these scenarios constitute plagiarism as well as other offences. One academic who considered scenarios A and B as 'other academic offence' and scenarios C and D as 'plagiarism' commented that *"... paying someone or deliberately letting someone else use your code would both infringe the no-plagiarism declaration a student signs on submitting coursework - so would be plagiarism"*. Another academic who considered scenarios A and B as 'other academic offence' and scenarios C and D as 'plagiarism' commented that *"Serious cheating or deliberate plagiarism, a 'red-card offence'. I tell the students I am assessing them, not their sources, and this is an attempt to gain a qualification by fraud."* A third academic considered all scenarios to be 'plagiarism' justified by the following comment.

> *"The act of submitting somebody else's work as your own without acknowledgment is generally plagiarism - in some of these cases other academic offences have been committed too. In particular stealing another student's work is an academic offence (and potentially a criminal or civil offence) in its own right."*

Regarding *scenario E*, comments received indicate that some of the academics consider this scenario to be collusion and some consider it plagiarism. Some of the academics that have considered this scenario as collusion commented, *"'intentionally permits...' is collusion."*, *"It's a moot point what the name is, but I think I'd call the last one collusion."*, and some of the academics that considered this scenario as 'plagiarism' provided comments such as, *"I would see the last case as colluding in plagiarism - still an offence."*, *"The last is complicity in plagiarism, which is a form of academic misconduct"*, *"At my University plagiarism and collusion are regarded as, and treated as, the same offence, so I'm not used to making any distinction between the two."*

The findings show that there was a wide agreement between academics that this scenario is an academic offence, and whether it is regarded as collusion, plagiarism or another academic offence depends on university regulations. Some universities consider collusion and plagiarism as separate offences, while other universities do not make any distinctions between the two.

4.4 Question 4

In this question, the academics were asked to provide their opinion as to which academic offence applies to given scenarios regarding group actions, self-plagiarism and collusion. The purpose of this question is to determine which student actions can indicate plagiarism. Question 4 is shown in *Figure 7*.

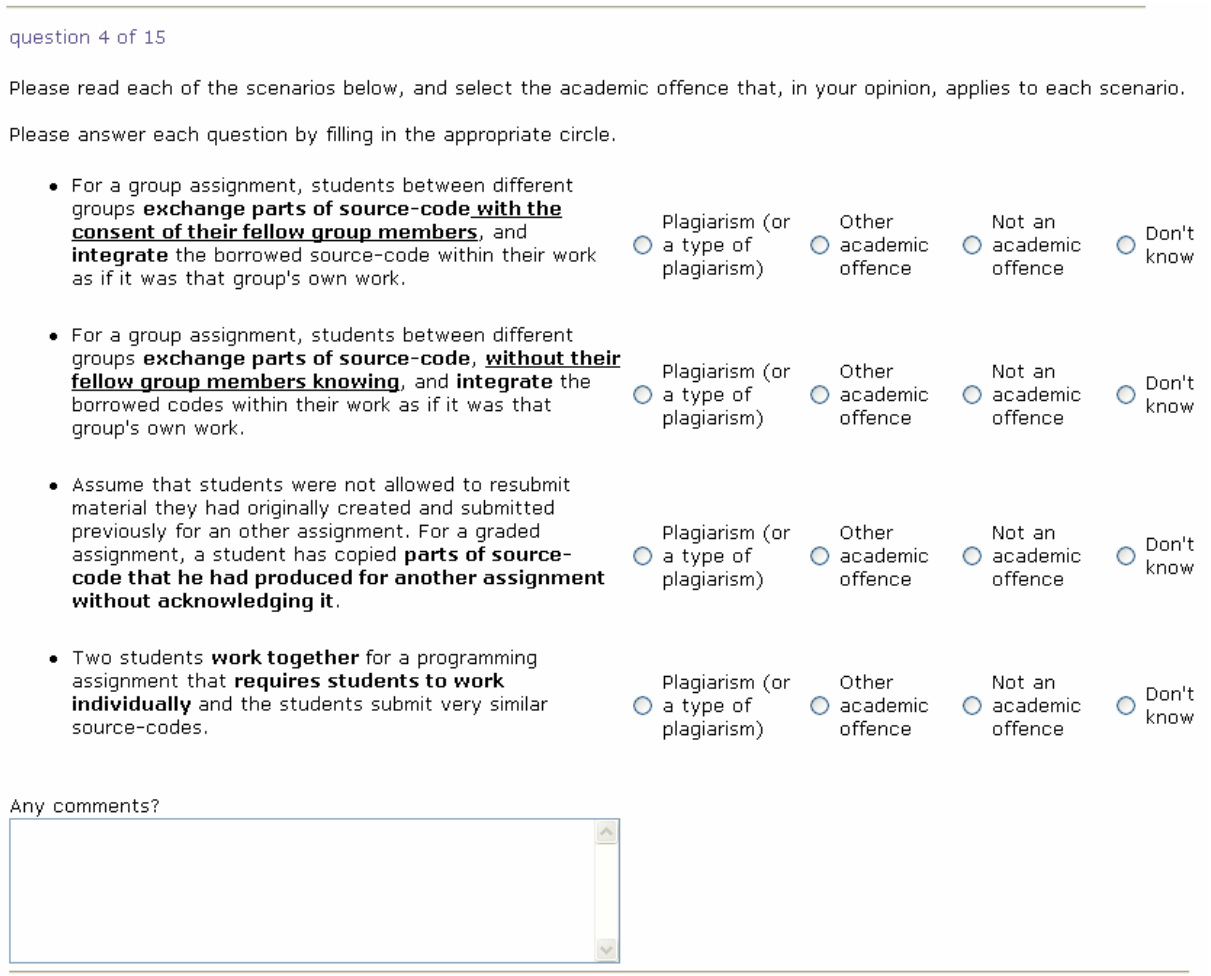


Figure 7: Question 4

The results of question 4 are presented in *Figure 8*, which corresponds to the following:

A. For a group assignment, students between different groups exchange parts of source-code with the consent of their fellow group members, and integrate the borrowed source-code within their work as if it was that group's own work.
B. For a group assignment, students between different groups exchange parts of source-code, without their fellow group members knowing, and integrate the borrowed codes within their work as if it was that group's own work.
C. Assume that students were not allowed to resubmit material they had originally created and submitted previously for another assignment. For a graded assignment, a student has copied parts of source-code that he had produced for another assignment without acknowledging it.
D. Two students work together for a programming assignment that requires students to work individually and the students submit very similar source-codes.

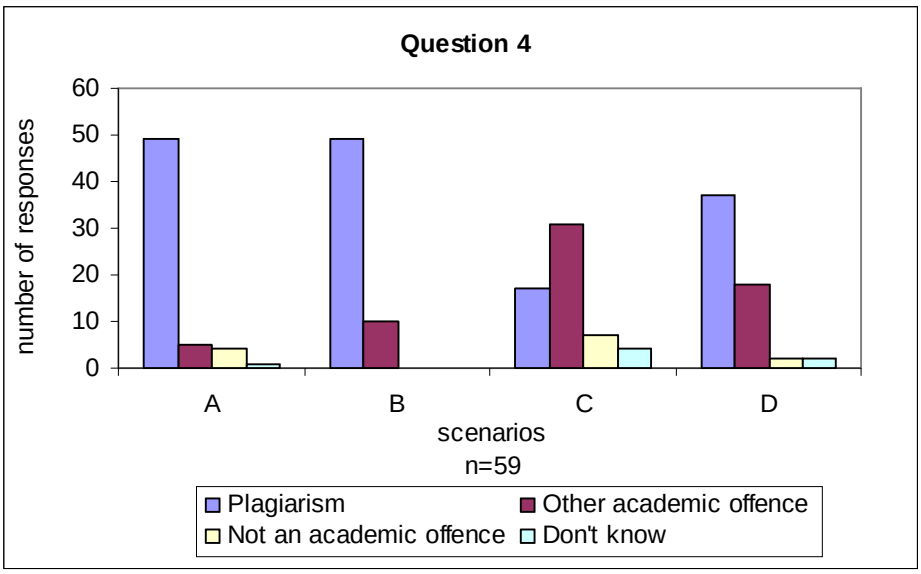


Figure 8: Chart shows the results from the responses for Question 4

The majority of the academics consider scenarios A, and B as plagiarism. Considering the comments given by academics to previous questions, these scenarios can constitute plagiarism, collusion or another academic offence depending on university regulations. Scenario B can constitute plagiarism, collusion as well as another academic offence since material was obtained without permission (the type of academic offence depends on the academic regulations of each university).

Regarding scenarios A and B, some academics expressed some concerns on the issue of plagiarism in group assignments. One academic commented, *"I avoid group assignments for just these reasons. I permit students to work together as long as they individually write their own code and documentation. I regard this as pedagogically valuable"*, while another academic observed *"… Some students learn better while working in groups but they must ensure that the work submitted is original and their own. The tell tale here is the comments (and spelling mistakes) are identical."*

The majority of academics considered scenario D as an academic offence, 'plagiarism' or 'another academic offence'. However, as mentioned above the name depends on university regulations. Academics commented, *"the last one [scenario D] is very tricky to decide; I'm very happy for students to work together, but very unhappy when I have to make this kind of decision. I'd be inclined to say 'plagiarism', but treat a first offence leniently"*, another academic considered this scenario as *"very common and usually accompanied with denials that they had submitted the same code and/or they didn't know they weren't allowed to work together"*.

In student assignments, self-plagiarism occurs when a student copies entire or parts of his/her own assignment and submits it as part of another assignment without providing proper acknowledgement of this fact. However, when we asked academics whether it constitutes plagiarism if a student resubmits source-code they have originally created and submitted previously for another assignment (see scenario C) we received some controversial responses. The majority of the academics (48 out of 59) characterised this scenario as an academic offence (17 as plagiarism and 31 as other academic offence). In their comments, those academics characterised this scenario as *"self-plagiarism"'*, *"breach of assignment regulations if resubmission is not allowed"*, and *"fraud if resubmission is not acknowledged"*.

Some academics consider reusing source-code from other assignments and not providing acknowledgements as 'not an academic offence'. Those academics argue that in object-oriented environments were reuse is encouraged, it seems inappropriate to disallow students from reusing source-code they have produced as part of another programming

assignment. The comments and responses provided by the academics who did not describe this scenario as 'plagiarism' or 'another academic offence' point to the controversial issue on source-code reuse aforementioned. One academic who replied 'don't know' remarked that *'Students should reuse code for assessments where possible!'* and another was clear that it was 'not an academic offence', and emphasised *"I find it hard to assume that students were not allowed to resubmit material."* A third academic, who also stated that it was 'not an academic offence', commented: *"would this ever happen in a programming oriented module when we behove students not to reinvent the wheel?"*

In conclusion, since 48 out of 59 academics characterised the action of resubmitting source-code produced as part of another assessment as a type of academic offence (plagiarism or other) we can conclude that resubmitting source-code without providing appropriate acknowledgements *may* lead to an academic offence if this is not allowed for the particular assignment.

4.5 Question 5

Question 5 consists of brief scenarios about intentional and unintentional plagiarism in graded and non-graded assignments. In this question, academics were asked to provide their opinions as to which academic offence applies to given scenarios. The purpose of this question is to determine whether intentions and assignment importance influence the decision as to whether plagiarism has occurred. Question 5 is shown in the figure below.

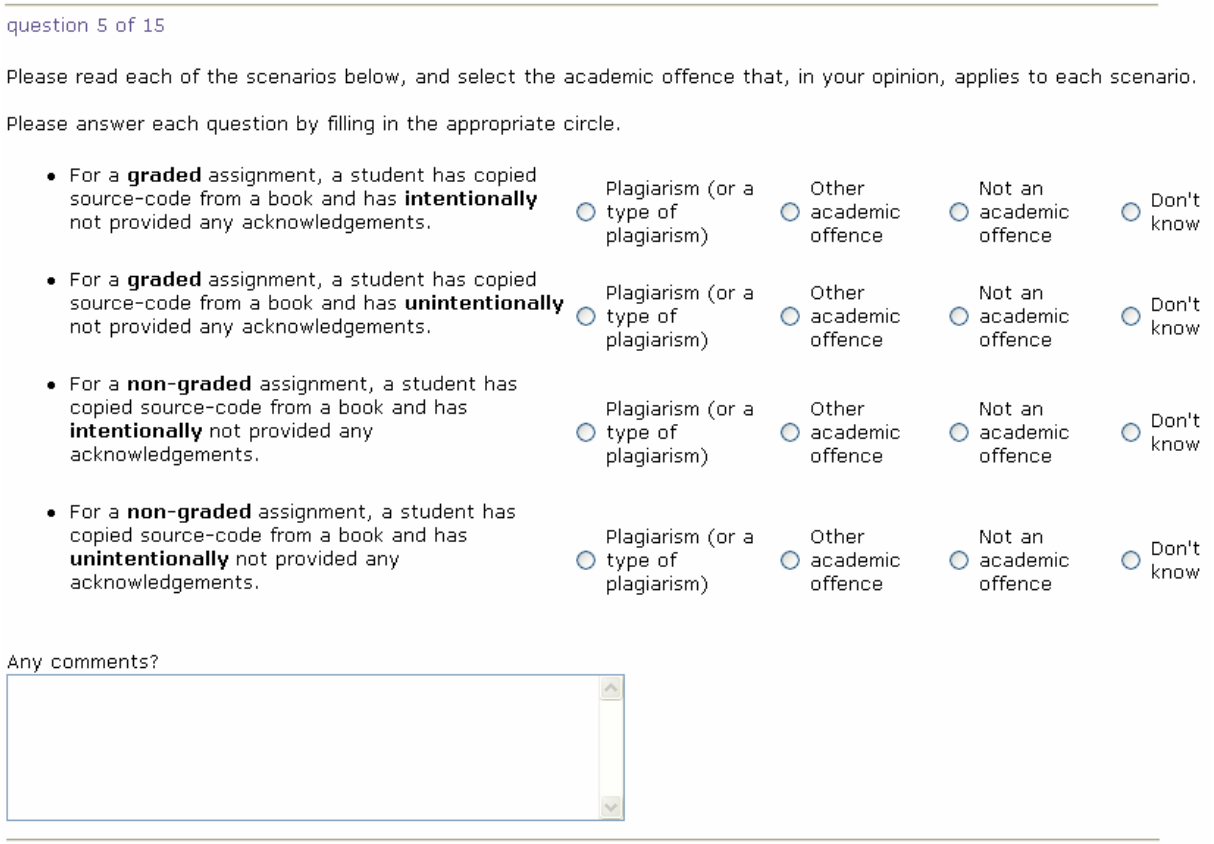


Figure 9: Question 5

The responses from question 5 are shown in *Figure 10*, which corresponds to the following:

A. For a graded assignment, a student has copied source-code from a book and has intentionally not provided any acknowledgements.
B. For a graded assignment, a student has copied source-code from a book and has unintentionally not provided any acknowledgements.
C. For a non-graded assignment, a student has copied source-code from a book and has intentionally not provided any acknowledgements.
D. For a non-graded assignment, a student has copied source-code from a book and has unintentionally not provided any acknowledgements.

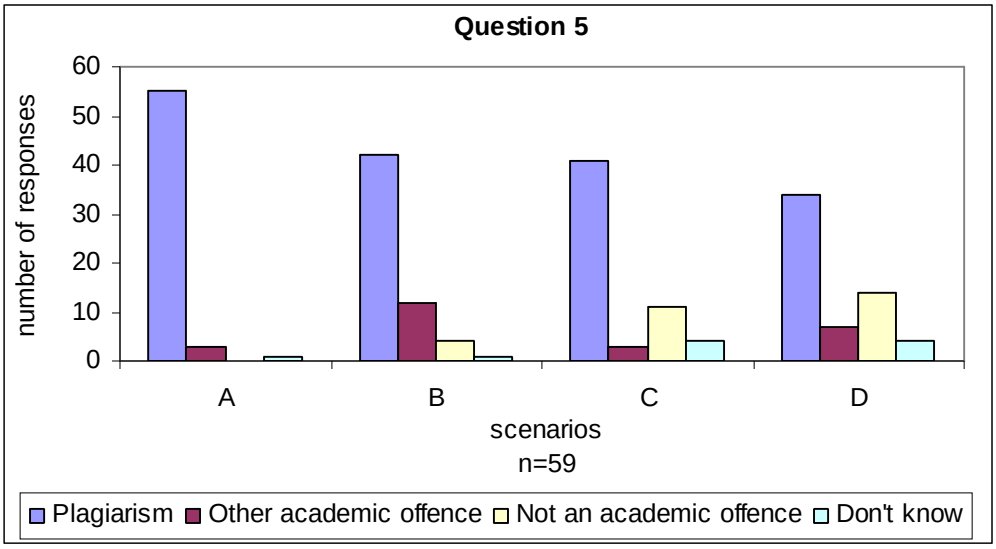


Figure 10: Chart shows the results from the responses for Question 5

The survey included questions regarding plagiarism in graded and non-graded assignments. By non-graded assignments, we refer to assignments that do not contribute to the overall module mark (such as some laboratory and tutorial exercises).

The results show that plagiarism can occur regardless of whether or not the student intended to provide acknowledgements to the sources they used. Academics commented that the degree of intent determines the seriousness of the plagiarism and consequently the penalty applied to the students work. Hence, plagiarism can occur intentionally or unintentionally, and the penalty imposed to the student will depend on the degree of the student's intention to commit plagiarism, *"It is a case of degree here. Typically students do not just fail to reference one source but many. For non graded work (presumably being used formatively) it would be better to highlight the error without formal censure."*

The results also suggest that plagiarism can occur regardless of whether an assignment is graded or non-graded. Two academics commented that plagiarism punishments for graded and non-graded assignments may vary. The first academic who provided a 'don't know' response for scenarios C and D commented that *"[for scenarios C and D] it depends on what the rules were for that non-graded assignment. If they plagiarise then it's plagiarism irrespective of whether the assignment is graded, it just means that the penalty may be different"*, and the second academic who provided a 'plagiarism' response for both scenarios C and D commented that *"the act of plagiarism should be related to whether an assignment is graded or not. The intention also would have to do with the amount of penalty applied to."*

The results show that plagiarism can occur regardless of whether an assignment is graded or non-graded. In addition, many academics commented that action against students should not be taken for non-graded assignments, and in such cases, the students should be

approached and informed or warned about plagiarism implications on graded assignments: *"If the assignment is not contributing towards the mark for the module then the correct protocol should be brought to the attention of the student."* In addition one academic who considered scenarios C and D as 'not and academic offence' commented: *"My logic here is that you can't penalise someone for an assignment they didn't have to do. The last two questions are still plagiarism in my view but you couldn't take any action against the student for it."*

Furthermore, some of the academics commented that their university regulations on plagiarism would only apply to graded work and plagiarism is only an academic offence if it concerns work submitted for credit. One academic stated: "*last two *are* cases of plagiarism but academic regulations might define 'offence' in terms of intention to gain higher grades"*, and another academic observed that the *"last two are not offences but morally incorrect and could lead to the practice being repeated for assessed work when it will be plagiarism"*.

In conclusion, copying without providing any acknowledgements can constitute plagiarism whether this is done intentionally or unintentionally. Plagiarism can occur in both graded and non-graded assignments. The degree of intent determines the seriousness of the plagiarism. The penalty imposed from plagiarism in graded and non-graded assignments may differ and may depend on the degree of intent to commit plagiarism.

4.6 Question 6

Question 6 consists of brief scenarios regarding source-code referencing and plagiarism. In this question, academics were asked to provide their opinions as to which academic offence applies to the given scenarios. The purpose of this question is to determine whether inappropriate referencing can suggest plagiarism. Question 6 is shown in the figure below.

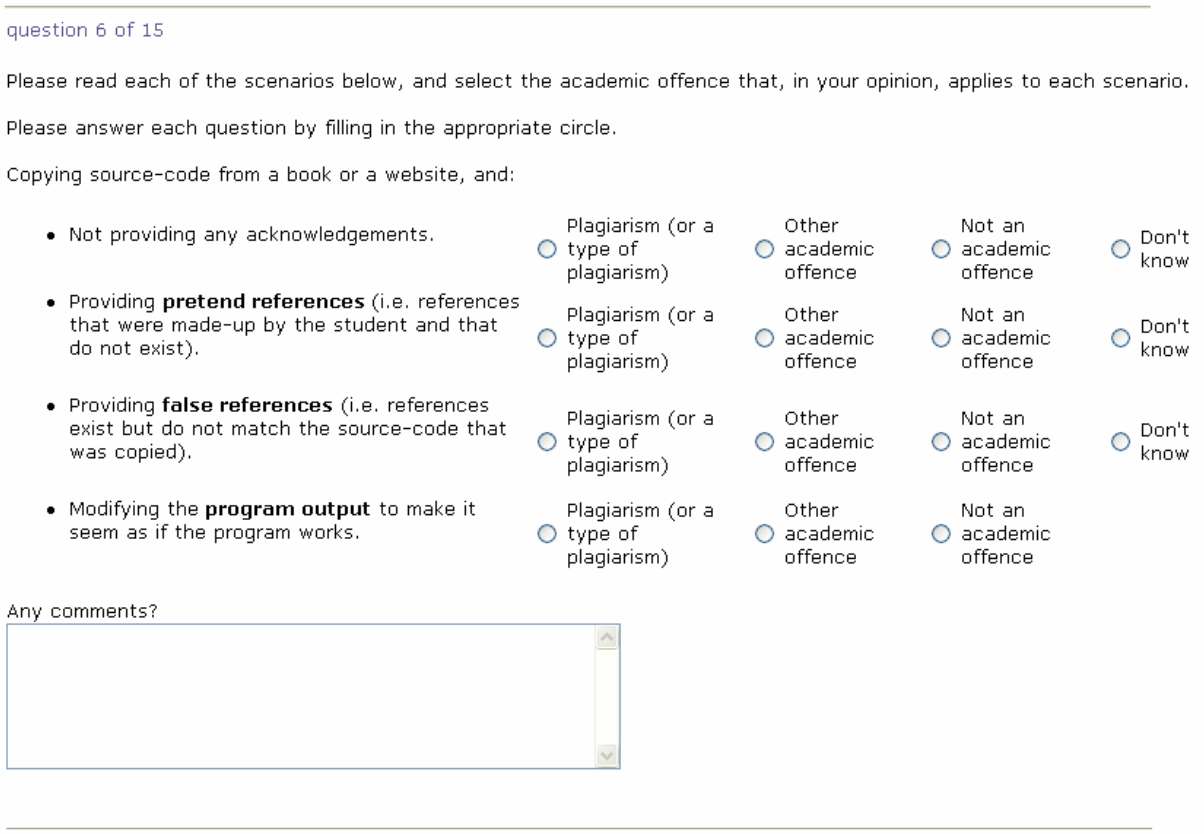


Figure 11: Question 6

The results of question 6 are presented in *Figure 12*, which corresponds to the following:

A. Not providing any acknowledgements
B. Providing pretend references (i.e. references that were made-up by the student and that do not exist).
C. Providing false references (i.e. references exist but do not match the source-code that was copied)
D. Modifying the program output to make it seem as if the program works

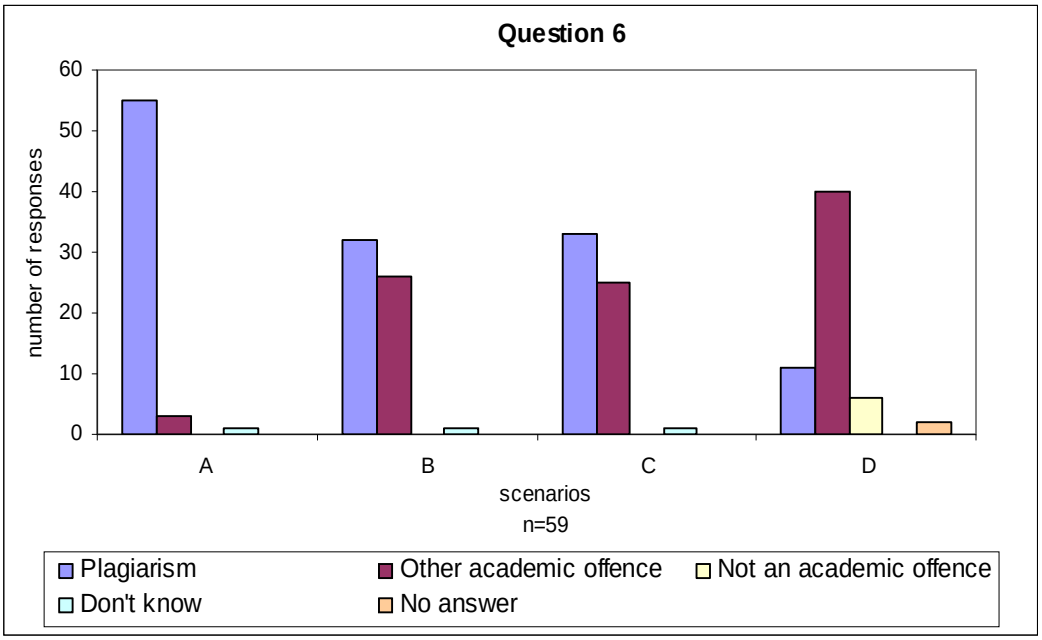


Figure 12: Chart shows the results from the responses for Question 6

The results show that academics consider copying source-code from a book or a website, and not providing any acknowledgements as source-code plagiarism. For scenario A, one academic commented that there is a difference between poor referencing and plagiarism, and that the difference between poor referencing and plagiarism is one that it is not easily determined.

For scenarios B and C, academics commented that these scenarios show intent by the student to plagiarise. Regarding scenario D, if source-code was copied and unacknowledged this can constitute plagiarism as well as other academic offences characterised by academics as *'plagiarism'*, *"falsification"*, *"fraud"*, *"cheating"* and as *"an act that it raises ethical, scientific and academic integrity issues"*.

The findings show that there was a wide agreement between academics that scenarios B, C, and D suggest an academic offence, however whether these are regarded as 'plagiarism' or 'another academic offence' seems to depend on university regulations.

4.7 Question 7

Question 7 consists of brief scenarios to determine the experiences of academics regarding the possibility of plagiarism occurring. In this question, academics were asked to provide their opinion as to which academic offence applies to the given scenarios. Question 7 is shown in *Figure 13*.

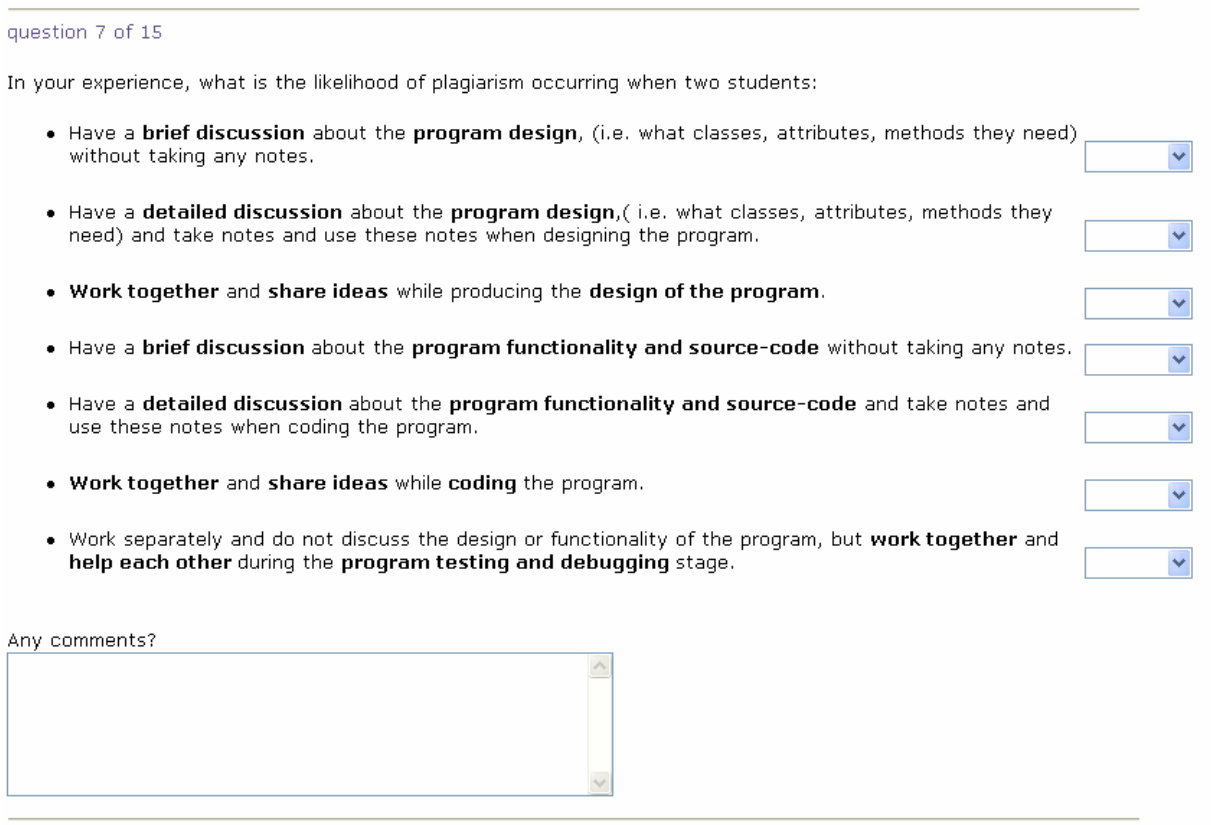


Figure 13: Question 7

The results of question 7 are presented in *Figure 14*, which corresponds to the following:

A. Have a brief discussion about the program design, (i.e. what classes, attributes, methods they need) without taking any notes.
B. Have a detailed discussion about the program design, (i.e. what classes, attributes, methods they need) and take notes and use these notes when designing the program.
C. Work together and share ideas while producing the design of the program.
D. Have a brief discussion about the program functionality and source-code without taking any notes.
E. Have a detailed discussion about the program functionality and source-code and take notes and use these notes when coding the program.
F. Work together and share ideas while coding the program.
G. Work separately and do not discuss the design or functionality of the program, but work together and help each other during the program testing and debugging stage.

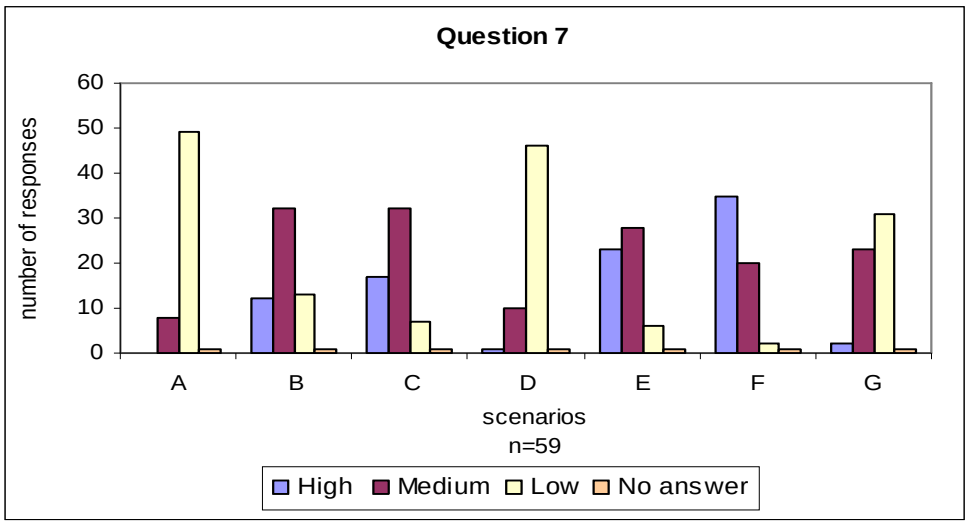


Figure 14: Chart shows the results from the responses for Question 7

The results show that in the experiences of academics there is low likelihood of plagiarism occurring when students have a brief discussion about the program design, (i.e. what classes, attributes, methods they need) or about the program functionality and source-code without taking any notes. Many academics commented that they have *"no objections to students sharing ideas"*, and commented that it is "*pedagogically valuable*" for students to engage actively in sharing ideas while discussing content and assignments as long as they do not copy each others work and hence *"write their own code and documentation"*.

Two academics who encourage students to share ideas commented, *"I have no problem with sharing ideas. Given the restricted types of programs undergrads write, it is inevitable that certain design decisions lead to similar code"*, *"I am personally not too worried about students discussing their work, it is the outright copying that I find most offensive and educationally pointless"*. A third academic observed:

> *"Important to remember that we often expect people to discuss, engage actively in discussing content and assignments - also preparation for future working in teams. Often we need to be much clearer in what we require of students - and what we don't."*

Regarding *scenario G*, academics commented that it would be an academic offence if two students help each other with the coding of the program (i.e., collusion). Furthermore, the likelihood of plagiarism occurring for scenario G depends on the type of testing being carried out. The likelihood of plagiarism occurring during black box testing tends to be low and the likelihood of plagiarism occurring during white box testing tends to be high. Note that with black box testing the tests are conducted at the software interface level, and with white box testing the tests on the code's procedural detail are carried out. Furthermore, the comments for scenario G are that the source-code similarities are likely to be low in such a scenario, but if documentation of testing is required for the assignment submission, and plagiarism occurs in the testing part of the documentation due to the students using the same test cases then the likelihood of plagiarism is high. However, if the students are not asked for their test cases then the probability of plagiarism is low.

In addition, academics commented that their answers on the likelihood of plagiarism occurring would differ depending on the assignment specification. For example, for assignments in which the design of the program is more important than the coding, then the likelihood of plagiarism occurring on the design tasks would be higher, and if the majority of the marks were given for the testing documentation the likelihood of plagiarism occurring on testing documentation would be higher, and so on.

4.8 Summary of Part One: Description of what can constitute source-code plagiarism

The information that was collected from the survey responses was analysed and collated to create a description of what constitutes source-code plagiarism from a wide academic perspective. All of the information presented in this section was gathered from the responses to questions in the survey.

4.8.1 Source-Code Plagiarism

Source-code plagiarism in programming assignments can occur when a student re-uses (4.8.1.1) source-code authored by someone else by obtaining (4.8.1.2) the source-code either with or without the permission of the original author and intentionally or unintentionally not properly acknowledging (4.8.1.3) the borrowed source-code and submits it as his/her own work.

If a student reuses (4.8.1.1) source-code that s/he produced as part of another assessment (in which s/he has gained academic credit) without properly acknowledging (4.8.1.3) this fact, it can constitute self-plagiarism or another academic offence (name of academic offence depends on university regulations).

If a student reuses (4.8.1.1) source-code authored by someone else (or produced by that student as part of another assessment) and *provides acknowledgements* then this can constitute to breach of assignment regulations, and not plagiarism (or self-plagiarism).

4.8.1.1 Re-use

Re-use includes the following:

a. Reproducing/copying without making any alterations.
b. Reproducing/copying and minimally or moderately adapting it. Minimal or moderate adaptation occurs when the work submitted by the student still contains some of the original source-code.
c. Converting the whole or part of someone else's source-code to a different programming language. Whether this constitutes plagiarism depends on the similarity between the languages and the effort required by the student to do the conversion. If the student takes ideas and inspirations from source-code written in another programming language and the source-code is entirely authored by the student it may not constitute plagiarism.
d. Automatically generating source-code using code-generating software can constitute plagiarism if the use of such software is not explicitly allowed in the assignment specification.

4.8.1.2 Obtaining

Obtaining the source-code either with or without the permission of the original author includes the following:

a. A student pays another person (or a student on the same module) to create part or whole of their source-code.

b. A student steals another student's source-code.
c. Two or more students collaborate (work together) on a programming assignment that requires students to work individually and the students submit similar source-codes. This can constitute plagiarism or collusion (name of academic offence depends on the academic regulations).
d. Students between different groups carrying out the same assignment, exchange parts of source-code with or without the consent of their fellow group members.

In the above list, source-code plagiarism can co-occur with other academic offences (such as theft, cheating, and collusion) depending on academic regulations. This is a very limited list since there are numerous ways that students can obtain source-code written by other authors.

4.8.1.3 Not properly acknowledging

Not properly acknowledging includes the following:

a. Not acknowledging the source and authorship of the source-code, within the program source-code (in the format of a comment) and in the appropriate documentation.
b. Providing "pretend" references (i.e. references that were made-up by the student and that do not exist) is a form of academic misconduct, often referred to as fabrication, and it can co-occur with plagiarism.
c. Providing "false" references (i.e. the references exist but do not match the source-code that was copied) is a form of academic misconduct, often referred to as falsification, and it can co-occur with plagiarism.
d. Modifying the program output to make it seem as if the program works when it is not working is a form of academic misconduct (i.e., falsification), and it can co-occur with plagiarism.

4.9 Conclusion

An important issue addressed in the first part of the report is that of source-code reuse. Some academics have expressed uncertainties whether reuse without acknowledgement constitutes plagiarism, however, the majority of academics suggest that when reuse is permitted in assignments students should provide acknowledgement of the parts of the source-code written by other authors.

The issue on whether it constitutes plagiarism when students reuse (without providing acknowledgements) source-code they have produced as part of other assignments has received different opinions. The majority of academics consider resubmission of source-code without acknowledgement as an academic offence, however a small but significant number of academics disagree that this action constitutes an academic offence since source-code reuse is encouraged in object-oriented programming languages.

We have used the information gathered from the survey to create a description of what can constitute source-code plagiarism from the perspective of academics who teach programming on computing courses.

5.0 Survey Part Two

This section is concerned with analysing the information gathered from the responses to questions 8 to 12 in the survey. Questions 8 and 9, are concerned with the importance of an assignment in terms of its contribution to the overall module mark for academics to consider proceeding with investigation into possible plagiarism. Questions 10 and 11, are concerned with the amount of similarity that two pieces of source-code must contain in order for academics to take certain actions on plagiarism. Question 12 consists of two similar pieces of source-code and the academics were asked to provide ratings as to how similar are the pieces of source-code, and the likelihood that plagiarism occurred.

During the design of the questionnaire, it was suspected that some academics might be reluctant to provide answers to questions asking for quantitative information due to the subjectivity issues surrounding the topic of similarity thresholds and source-code plagiarism. However, for research purposes, such questions were considered as important because they could provide an insight into similarity thresholds between source-codes for certain actions to be taken, that would be important during the development of software to detect source-code plagiarism.

5.1 Statistics for quantitative analysis

Due to the subjective nature of providing similarity thresholds it was suspected that the values provided by academics for questions asking for quantitative information would vary. Therefore, it was important that appropriate statistics were selected and used in order to analyse and describe as accurately as possible the quantitative information gathered from the survey.

Statistics such as the mean and standard deviation may only be appropriate for datasets with symmetric distributions because such statistics assume that the datasets have a normal distribution. Statistics such as median and the range are appropriate for datasets that may or may not have a normal distribution. Therefore, as a starting point of analysing the data, it was important to employ some statistics that describe the distribution of the data in order to determine beforehand which statistics to use.

To measure the variability in the values provided by academics, the *skewness* of each dataset was measured. Box-and-whisker plots were used to compare the distribution of data, and histograms were produced for comparing the distribution of frequencies.

5.1.1 Skewness

Skewness characterizes the degree of asymmetry of a distribution around its mean. Skewed distributions do not have a centre or apparent typical value for the distribution. If the distribution of the data is not symmetrical, it is called asymmetrical or skewed. In a symmetric (non-skewed) distribution, the typical value would be the centre of the distribution. The higher the number of skewness the more the asymmetrical the data is, and the nearest the skewness is to 0 the more symmetrical it is. The skewness for a normal distribution is zero, and any symmetric data should have skewness near zero.

5.1.2 Box and whisker plot

A box and whisker plot is a graph that presents information about a dataset in terms of the median value, upper and lower-quartile values, range value, and inter-quartile value of a dataset. The median value (also called 50^{th} percentile) is the middle value (centre value) in a dataset. The lower-quartile (also called 25^{th} percentile) is the median value of the lower part of the dataset set, where 25% of the values are smaller than the median value of the dataset; and the upper-quartile value (also called 75^{th} percentile) is the median value of the second part of the dataset, where 25% are larger than the median.

The range is the difference between the maximum and minimum values of a dataset. The inter-quartile range (IQR) is the difference between the upper-quartile and the lower-quartile. The IQR is the range of the middle 50% of the data set, and eliminates the influence of outliers because the highest and lowest quarters are removed. The IQR is a measure of the spread and also indicates the dispersion of a data set. Dispersion measures how close are values are to the mean value of a dataset.

The box and whisker plot is useful in indicating whether a distribution is skewed and whether outliers (unusually high values) exist in a dataset. Box and whisker plots are useful for comparing more than one dataset, and for comparing the distributions between datasets. The box itself contains the middle 50% of the data. The horizontal line across each box represents the median. The median is the number in the middle of a set of numbers; half the values are smaller than the median and half the values are larger than the median. If the median is not located in the middle of the box, then the distribution is skewed. The whiskers extend from the lower-quartile to the lowest number in the data set and from the upper-quartile to the greatest number in the data set. Hence, the ends of the whiskers indicate the minimum and maximum data values. The whiskers extend to at most 1.5 times the box width (the IQR) from both ends of the box.

Outliers are unusually high values. A mild outlier is a value that is more than 1.5 times the IQR and an extreme outlier is a value that is more than 3 times the IQR. In the box-and-whisker plot the mild outliers are marked with a small circle (o) and the extreme outliers are marked with an asterisk (*).

Box-and-whisker plots seem to be appropriate for analysing the quantitative data from this survey. These plots are particularly good when comparing distributions between datasets, they show variations in the data, unusual values and they also provide statistics about each fourth of the data. Statistics about different parts of a dataset are very important because each value describes a specific part of a dataset. For example, some academics have provided values at the low end of a scale, others values at the high end of a scale, and others values in the middle of the scale. The box-and whisker plot clearly shows where most variation exists in the data and are ideal for comparing more than one dataset.

5.1.3 Histogram

A histogram is a graph that consists of vertical bars showing the distribution of frequencies within certain ranges (or intervals) of values. The histogram can provide information such as the distribution of the data (symmetric or skewed), the most common responses, outliers in the data, and the variation of the data.

5.1.4 Statistics for comparing pairs of values

Several statistical tests exist for testing the differences between paired values. These tests fall into one of two categories, parametric tests and nonparametric tests. Parametric tests assume

that the distribution of the data is normal. A dataset with normal distribution is symmetrical (i.e. not skewed). Examples of parametric tests are the Pearson's correlation coefficient and the paired-samples T-test.

Nonparametric tests do not follow the assumption of normal distribution, and these tests are ideal for datasets with skewed distributions. The nonparametric tests for two related samples test for differences between paired values without making the assumptions required by the paired-samples T-test. Examples of nonparametric tests are the Spearman's rho correlation and the Wilcoxon signed-ranks test.

5.2 Questions 8 and 9

Questions 8 and 9 are concerned with the importance of an assignment, in terms of its weight towards the overall module mark, when investigation into possible plagiarism is to be carried out. The purpose of questions 8 and 9 was to gather some values as to what must be the minimum contribution of an assignment towards the overall module mark for academics to proceed with investigation into possible plagiarism. Questions 8 and 9 are shown in *figures 15 and 16* respectively.

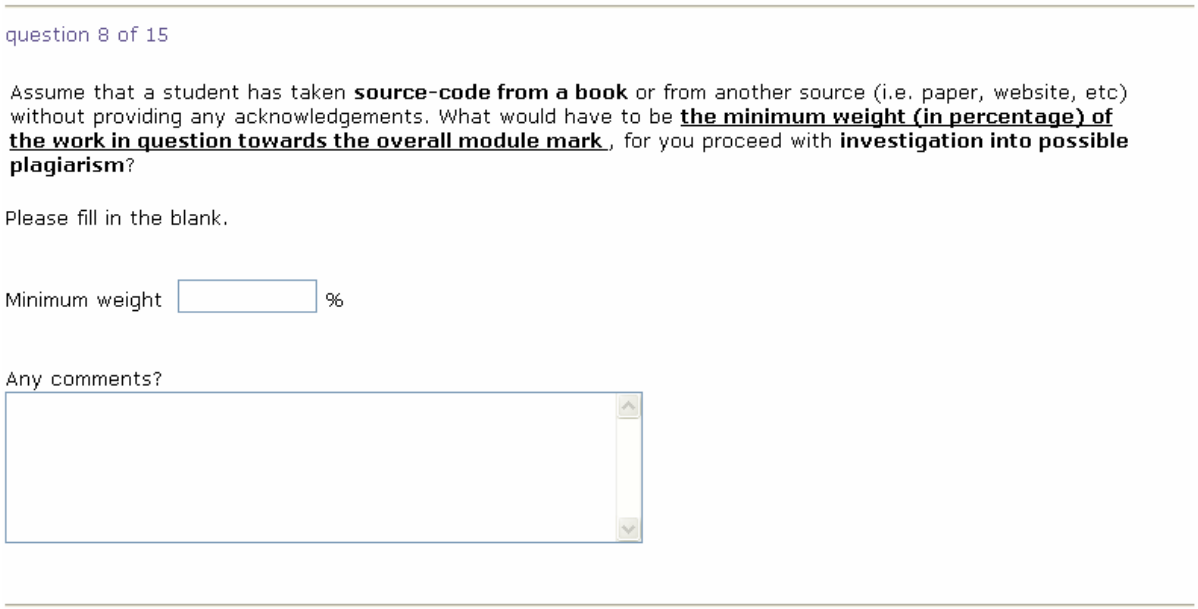


Figure 15: Question 8

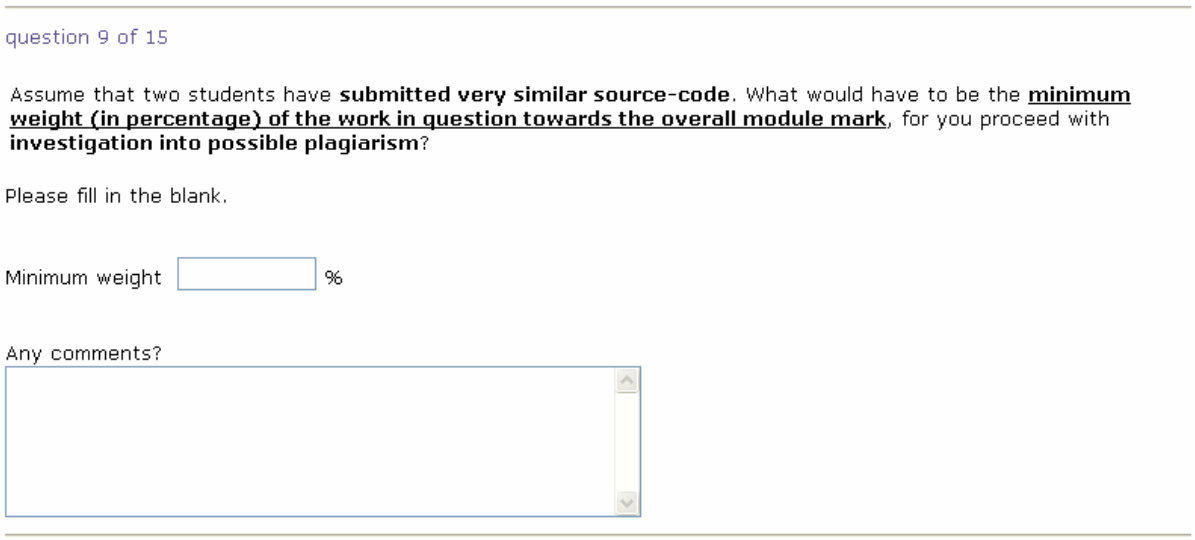


Figure 16: Question 9

Questions 8 and 9 are very similar with their only difference being that in question 8 a student has taken source-code from a book, and in question 9 two students have submitted very similar source-code. The responses for the two questions were analysed and compared. *Table 1* shows the responses for each question.

	Question 8	**Question 9**
Responses	**Frequency**	**Frequency**
0	7	8
1	18	18
2	1	0
5	5	6
10	6	7
12,5	1	1
15	2	2
20	5	5
25	4	3
30	0	1
35	1	0
40	1	0
No answer	8	8
Total	**59**	**59**

Table 1: Frequency table of answers to questions 8 and 9

Response values of zero suggest that even if the weight of an assignment is zero towards the overall module mark, investigation into possible plagiarism would still be pursued. Assignments with zero contribution can include tutorial work, non-graded exercises, and any other work completed by the student as part of a module. *Figure 17* below shows the responses to questions 8 and 9.

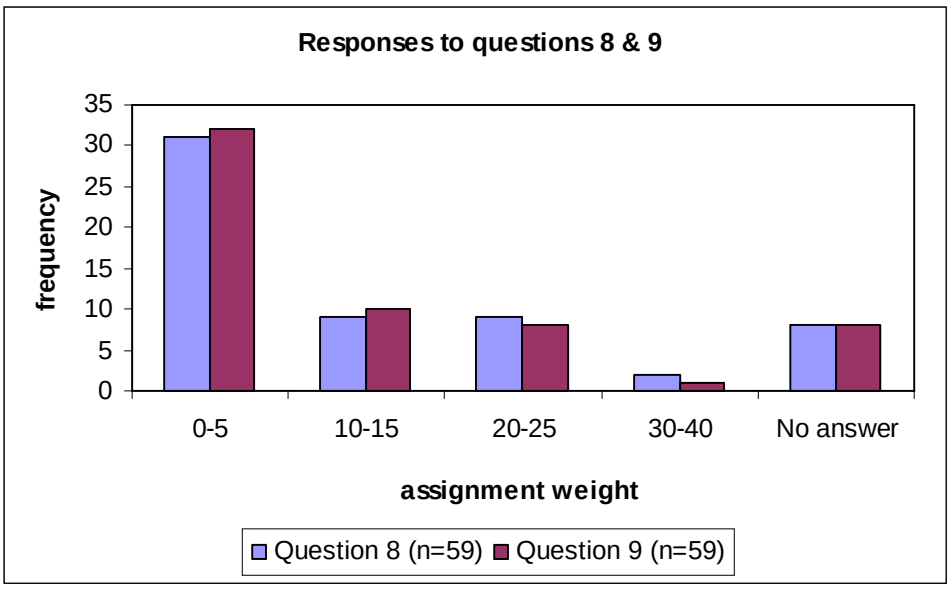


Figure 17: Chart shows the results from the responses for questions 8 and 9

The majority of academics who have provided a response to this question have given a value in the range of 0-5, suggesting a *"zero tolerance"* policy, and many commented that any assignment (even unmarked assignments) should be investigated for possible plagiarism regardless of their contribution to the overall module mark and appropriate action should be taken (whether penalties, or warning). The rationale for this is clear: *"If they cheat on a 5% assignment they'll cheat on a larger one. A short, sharp shock can prevent the full offence."*

In some instances, the severity of the offence is considered by the institution to vary depending on the contribution of the assignment:

> *"Any contribution would bring the same investigation. However, for the full range of penalties to be applied a minimum contribution to the degree award of 7%, or a second or subsequent upheld accusation, is required. This is University policy."*

There was agreement that regardless of the assignment's contribution towards the overall module mark, investigation into possible plagiarism should always be pursued and appropriate action should be taken (whether penalties, or warning). The actions taken by academics when they suspect plagiarism in both graded and non-graded assignments may depend on the university regulations as well as the academics themselves, and some actions (such as warning students) may not appear in university regulations. One academic emphasised, *"All plagiaristic work should be investigated as it has reached such epidemic proportions that the only control is draconian measures for any offence."*

The majority (42 out of 50) of academics have provided the same answer for both question 8 and 9. A pair-wise comparison on the values provided by the eight academics who have provided a different answer for questions 8 and 9 shows that the average for question 9 (average=11.38) is much lower that that of question 8 (average=18.5). This suggests that after considering the assignments contribution towards the overall module mark, some academics would consider proceeding with investigation into possible plagiarism if plagiarism between students was suspected, that they may not otherwise have done if they suspected that a student copied from a book. The statistical significance of this finding was tested using the Wilcoxon signed-ranks Test and it is described below.

The Wilcoxon signed-ranks test is carried out to compare paired medians from the same (or matched) sample. The main reason for carrying out the Wilcoxon signed-ranks test

was to examine in more detail the types of responses provided by academics for questions 8 and 9 rather than just the mean differences. The Wilcoxon signed-ranks method tests the null hypothesis that two related medians are the same.

The hypotheses tested are:

- **H_0:** There is no significant difference between the values provided by academics for question 8 and question 9
- **H_A:** There is a significant difference between the values provided by academics for question 8 and question 9
- **Conclusion:** Reject null hypothesis if test statistic is less than 0.05.

Table 2 shows the number of ranks, mean rank, and the sum of positive and negative ranks. The number of negative ranks is the number of academics who have provided higher values for question 8. The number of positive ranks is the number of academics who have provided higher values for question 9. The number of ties is the number of academics who have provided the same values for questions 8 and 9. The notes below *Table* 2 indicate what the positive and negative ranks relate to.

Ranks				
		N	**Mean Rank**	**Sum of Ranks**
question9 - question8	Negative Ranks	7(a)	4.14	29.00
	Positive Ranks	1(b)	7.00	7.00
	Ties	42(c)		
	Total	50		

a question9 < question8
b question9 > question8
c question9 = question8

Table 2: Wilcoxon Signed-ranks Test

Eight academics have provided different responses for questions 8 and 9. Seven academics have provided a higher value for question 8 and one academic has provided a higher value for question 9. The majority of the academics (42 out of 50) have provided the same values for both questions, and hence the large number of ties.

Table 3 below shows the test statistics of the Wilcoxon signed-ranks test. In *Table 3* if the asymptomatic significance value falls below 0.05 then there is a significant difference in the values provided by academics, and if the value is above 0.05 then there is no significant difference between the values given for questions 8 and 9.

Test Statistics(b)	
	question9 - question8
Z	-1.544(a)
Asymp. Sig. (2-tailed)	0.123

a Based on positive ranks
b Wilcoxon Signed-ranks Test

Table 3: Test statistics for the Wilcoxon signed-ranks test

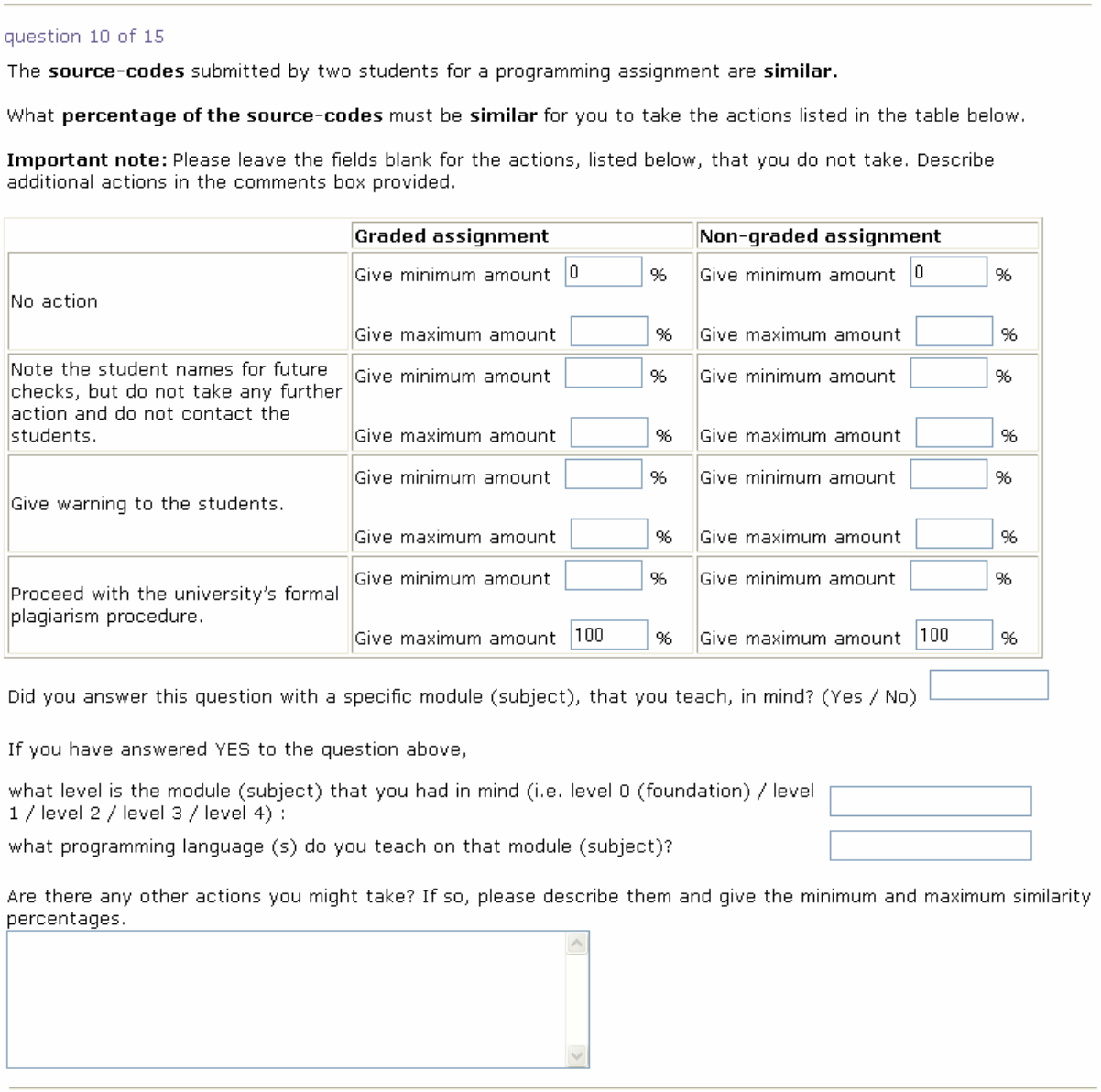
Since the symptomatic significance value is more than 0.05, we can conclude the null hypothesis holds “there is no significant difference between the values provided by academics for questions 8 and 9”.

Therefore, the results suggest that the decision of academics about whether they should proceed with investigation into possible plagiarism is not influenced by whether

plagiarism was suspected due to inappropriate collaboration between students or because students may have used source-code from a book without referencing. In addition, the results show that there is a general agreement between academics that a 'zero tolerance' plagiarism policy is appropriate, investigation into possible plagiarism should always be pursued and any case of plagiarism should be dealt with regardless of the assignment's contribution towards the overall module mark.

5.3 Question 10

The purpose of question 10 is to provide information about the amount of similarity that source-code submitted by two students must have in order for academics to take no action or another form of action. The differences between the values provided by academics for actions corresponding to graded and non-graded assignments were also investigated. The academics were asked to provide threshold values for actions corresponding to graded and for non-graded assignments by completing the relevant blanks in question 10 *(see Figure 18)*.



question 10 of 15

The **source-codes** submitted by two students for a programming assignment are **similar.**

What **percentage of the source-codes** must be **similar** for you to take the actions listed in the table below.

Important note: Please leave the fields blank for the actions, listed below, that you do not take. Describe additional actions in the comments box provided.

	Graded assignment	**Non-graded assignment**
No action	Give minimum amount 0 % Give maximum amount %	Give minimum amount 0 % Give maximum amount %
Note the student names for future checks, but do not take any further action and do not contact the students.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Give warning to the students.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Proceed with the university's formal plagiarism procedure.	Give minimum amount % Give maximum amount 100 %	Give minimum amount % Give maximum amount 100 %

Did you answer this question with a specific module (subject), that you teach, in mind? (Yes / No)

If you have answered YES to the question above,

what level is the module (subject) that you had in mind (i.e. level 0 (foundation) / level 1 / level 2 / level 3 / level 4) :

what programming language (s) do you teach on that module (subject)?

Are there any other actions you might take? If so, please describe them and give the minimum and maximum similarity percentages.

Figure 18: Question 10

This and the next paragraph apply to question 10 as well as to question 11 (see *Section 5.4*). The design of questions 10 and 11 was such that the minimum in one action could serve as the maximum in another action, for example, 'no action' could be 0-60, and 'give warning' could be 60-100. Academics who have answered questions 10 and 11 have provided similarity values that follow on from each other, for the actions they take. Hence, it was only necessary to use the minimum amount provided by academics for each of the actions, in order to avoid extra unnecessary statistical analysis.

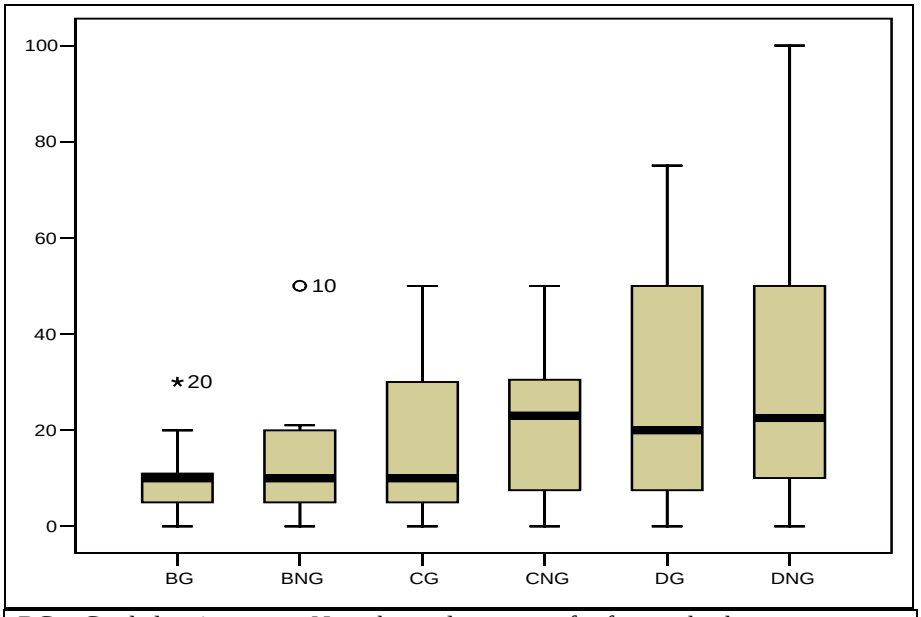
In the statistical analysis, *ms* refers to *minimum similarity* value (i.e. the minimum amount provided by academics). During statistical analysis of questions 10 and 11, the minimum values were used instead of the maximum values because these are considered as more meaningful, since *ms value*s indicate that if a similarity starts at a certain value then the relevant action is taken.

The instructions for questions 10 and 11 asked academics not to provide any values for actions that they would not take and to describe any other actions that they would take in the comments box provided. The number of responses for each action vary, because not all academics take all of the actions listed (i.e. some academics do not give warnings, they proceed with the university procedure when they detect source-code plagiarism; other academics give warnings if similarity is below certain thresholds, and proceed with the university procedure if plagiarism exceeds these thresholds).

A total of 44 out of 59 have responded to question 10 and the majority of those academics who did not respond have provided useful comments explaining their reasons for not completing this question. These are discussed in *"Section 5.5.2: Factors influencing the similarity values provided by academics"*.

For action 'No action for graded assignments' a total of 27 academics provided a *ms value* of zero, and three academics provided *ms values* 2, 5, and 30; and for action 'No action for non-graded assignments', 21 academics provided a *ms* value of zero, and 1 academic provided *ms value* of 2. The responses for action A "Take no action" will not be included in the comparison that follows, because the majority of the academics have provided the value of zero as the *ms value* for taking no action.

The *ms value*s for actions B, C, and D are compared in the box-and-whisker plot in *Figure 19.*



BG – Graded assignments: Note the student names for future checks
BNG – Non-graded assignments: Note the student names for future checks
CG – Graded assignments: Give warning to the students
CNG – Non-graded assignments: Give warning to the students
DG – Graded-assignments: Proceed with the university's formal plagiarism procedure
DNG –Non-graded assignments: Proceed with the university's formal plagiarism procedure

Figure 19: Box plot comparing the responses for each of the actions in question 10

Statistics							
		BG	**BNG**	**CG**	**CNG**	**DG**	**DNG**
N		21	13	33	24	40	20
Range		30	50	50	50	75	100
Minimum		0	0	0	0	0	0
Maximum		30	50	50	50	75	100
Percentiles	25^{th}	5	5	5	8.75	8.75	10
	50^{th}	10	10	10	23	20	22.5
	75^{th}	11	20	30	30.25	50	50
IQR		6	15	25	21.5	41.25	40

Table 4: Statistics for figure 10

Comparing the *ms value* of BG and BNG, the upper-quartile value is higher for the BNG dataset indicating that some academics have provided higher *ms value* in BNG. In addition, a mild outlier exists in the BNG dataset, because one academic provided a relatively high *ms value* of 50, when the rest of the values are between 0 and 20. Also, an extreme outlier exists in BG, because one academic has provided a relatively high *ms value* of 30, when the rest of the values are between 0 and 20. However, outliers were expected because providing similarity values can be a very subjective issue. Outliers are considered as important values in this survey, because they show variability in the opinions of academics.

Comparing the box plots of CG and CNG, the upper-quartile values, the lower-quartile and the median values are higher in CNG and this suggests that some academics have provided higher *ms value* for CNG. This suggests that for some academics to warn students on possible plagiarism there must be a higher similarity between non-graded assignments, than would be if the assignments were graded.

Comparing the values of DG and DNG although the upper-quartiles are the same, the lower-quartile and median values are higher for DNG indicating that academics have provided larger *ms value* for DNG. This suggests that for some academics to proceed with the formal plagiarism procedure there must be a higher degree of minimum similarity between non-graded assignments, than would be if the assignments were graded. In addition, the line between the upper-quartile and the maximum value (top whisker) value for DNG was higher than that of DG, which is another indication that academics have provided higher *ms value* for non-graded assignments.

In all pairs of actions (BG-BNG, CG-CNG, DG-DNG) the minimum value is zero, indicating that any of the actions could be taken if the *ms value* begins with zero. The comments of academics suggest that a zero *ms value* indicates zero tolerance and very low similarity (e.g., 0.001).

In the box-and-whisker plot in *Figure 19* above, the median values of all datasets are not located in the centre of the boxes, and the relative whiskers of each box are asymmetrical (i.e., differ in size), which suggests that the distributions are skewed. The higher the amount of skewness the more the asymmetrical the data is, and the nearest the skewness is to 0 the more symmetrical it is. In addition, the higher the skewness value, the greater is the variability in the responses provided by academics. The figure below shows the skewness values for each dataset.

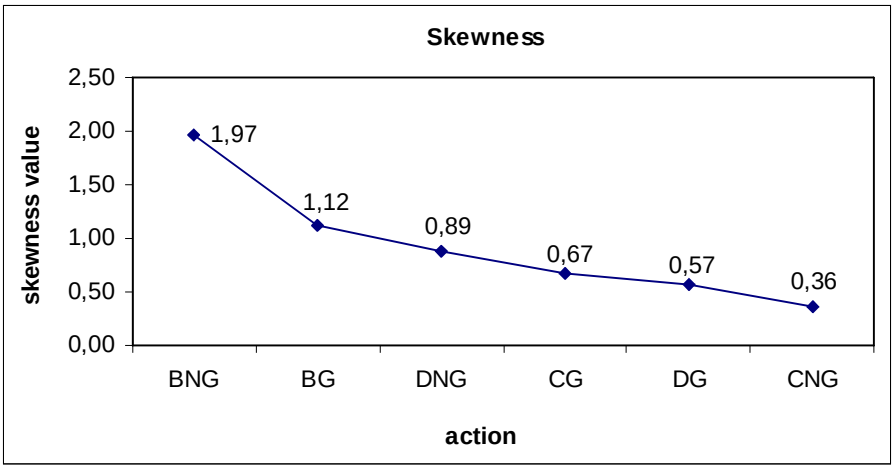


Figure 20: Skewness values of actions in question 10. The actions are sorted in order of skewness. The higher the skewness value, the higher is the degree of asymmetry in the dataset.

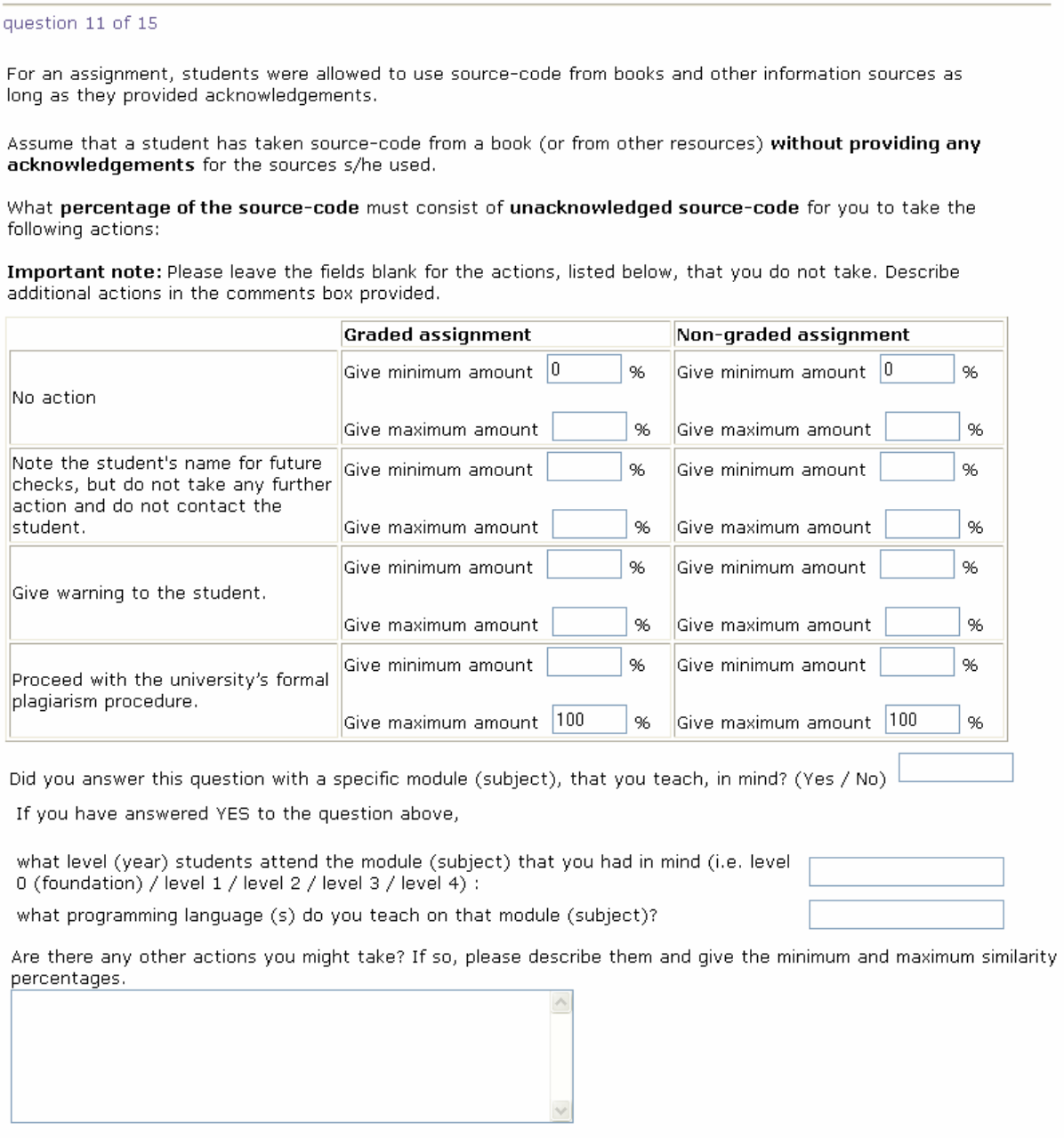
Figure 20 shows that BNG and BG were the most skewed datasets. We suspect that the degree of skewness was higher in the B datasets due to the low number of responses; noting student names is not a very popular action compared to giving warnings, or proceeding with formal university procedure. However, as shown in *Figure 20* a degree of skewness exists in all datasets concerned, some datasets having a higher degree of skewness than others do.

The distribution of the values in all datasets is skewed, which suggests that variability was present in the values provided by academics. The existence of variability in responses suggests that the values provided by academics may be influenced by factors other than just the source-code similarity. During the design of the questionnaire, it was suspected that the responses to question 10 would vary. For this reason, additional questions were provided with

question 10 in order to determine factors that may influence the responses given by academics. The factors that can influence the *ms value* provided by academics are described in *"Section 5.5: An investigation into the variability of similarity values in Questions 10 and 11."*

5.4 Question 11

The purpose of question 11 is to provide information as to how much of the copied source-code submitted by a student must be unacknowledged for the academic to take no action or another form of action. Question 11 is shown in *Figure 21*.


question 11 of 15

For an assignment, students were allowed to use source-code from books and other information sources as long as they provided acknowledgements.

Assume that a student has taken source-code from a book (or from other resources) **without providing any acknowledgements** for the sources s/he used.

What **percentage of the source-code** must consist of **unacknowledged source-code** for you to take the following actions:

Important note: Please leave the fields blank for the actions, listed below, that you do not take. Describe additional actions in the comments box provided.

	Graded assignment	Non-graded assignment
No action	Give minimum amount 0 % Give maximum amount %	Give minimum amount 0 % Give maximum amount %
Note the student's name for future checks, but do not take any further action and do not contact the student.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Give warning to the student.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Proceed with the university's formal plagiarism procedure.	Give minimum amount % Give maximum amount 100 %	Give minimum amount % Give maximum amount 100 %

Did you answer this question with a specific module (subject), that you teach, in mind? (Yes / No)

If you have answered YES to the question above,

what level (year) students attend the module (subject) that you had in mind (i.e. level 0 (foundation) / level 1 / level 2 / level 3 / level 4) :

what programming language (s) do you teach on that module (subject)?

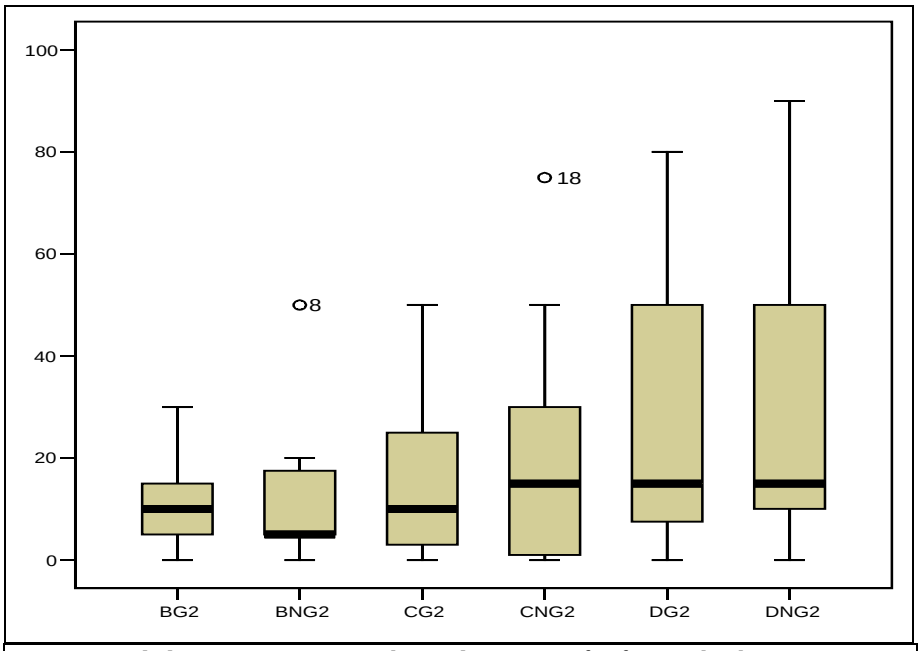
Are there any other actions you might take? If so, please describe them and give the minimum and maximum similarity percentages.

Figure 21: Question 11

The differences between the *ms value* provided by academics for graded and non-graded assignments were investigated. Because questions 10 and 11 both have the same action names, the actions of question 11 were renamed by adding a "2" at the end of each action name, for example, BG in question 10, is BG2 in question 11.

There is a variety in the number of responses for each action because not all academics take all of the suggested actions included in the question. Furthermore, 42 out of 59 academics have responded to this question and the majority of those academics who did not provide an answer provided useful comments explaining their reasons for not responding. These were discussed in "*Section 5.5.2: Factors influencing the similarity values provided by academics"*.

For action "No action for graded assignments", 58 academics provided a *ms value* of 0, and 1 academic provided a value of 5. These responses were the same for action "taking no action for non-graded assignments". These responses were not included in the comparison that follows, because the majority of the academics provided the value of zero as the *ms value* for taking no action. The *ms value*s for actions B, C, and D are compared in the box-and-whisker plot in *Figure 22*.



BG2 – Graded assignments: Note the student names for future checks
BNG2 – Non-graded assignments: Note the student names for future checks
CG2 – Graded assignments: Give warning to the students
CNG2 – Non-graded assignments: Give warning to the students
DG2 – Graded-assignments: Proceed with the university's formal plagiarism procedure
DNG2 –Non-graded assignments: Proceed with the university's formal plagiarism procedure

Figure 22: Box plot comparing the responses for each of the actions in question 11

Statistics							
		BG2	**BNG2**	**CG2**	**CNG2**	**DG2**	**DNG2**
N		17	11	31	21	39	21
Range		30	50	50	75	80	90
Minimum		0	0	0	0	0	0
Maximum		30	50	50	75	80	90
Percentiles	25^{th}	5	5	3	1	7.5	10
	50^{th}	10	5	10	15	15	15
	75^{th}	15	17.5	25	30	50	50
IQR		10	12.5	22	29	42.5	40

Table 5: Statistics for figure 16

Comparing BG2 and BNG2, the upper-quartile value is higher for the BNG2 dataset suggesting that some academics have provided higher *ms value* in BNG2. In addition, a mild outlier exists in BNG2, because one academic provided a relatively high *ms value* of 50, when the rest of the values are between 0 and 20. However, because only 11 academics would take the action of noting student names for future checks, the dataset is small and outliers were expected.

Regarding CG2 and CNG2, the upper-quartile and median values of CNG2 are higher than those values of CG2. This suggests that for some academics to warn students on possible plagiarism there must be a higher degree of similarity between non-graded assignments, than would be if the assignments were graded.

Comparing the values of DG2 and DNG2 although upper-quartiles are the same the lower-quartile values are higher for DNG2 indicating that some academics have provided larger *ms value* for DNG2. This suggests that for some academics to proceed with the formal plagiarism procedure there must be a higher similarity between non-graded assignments, than would be if the assignments were graded.

Furthermore, the range of responses is larger in the non-graded datasets (BNG2, CNG2, DNG2) than the range of their graded pairs (BG2, CG2, DG2) which is another indicator that some academics have provided higher *ms value* for non-graded assignments.

In all pairs of actions (BG2-BNG2, CG2-CNG2, DG2-DNG2) the minimum value is zero, indicating that any of the actions could be taken if the *ms value* begins with zero. The comments of academics suggest that by zero similarity indicates no tolerance and very low similarity (e.g., 0.001).

In *Figure 22* the median in all boxes is not in the centre of the box, and also the relative whiskers of each box are asymmetric, which indicates that the distributions are skewed. The skewness of the distributions of each dataset is shown in *Figure 23*. The lower the degree of skewness is, the less the distribution in the data, and hence the lower the variability in the responses provided by academics.

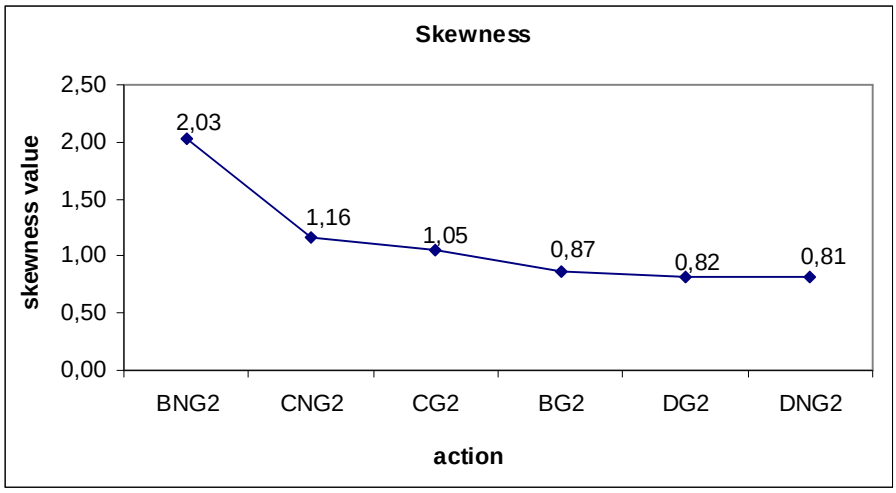


Figure 23: Skewness values of actions in question 11. The actions are sorted in order of skewness. The higher the skewness value, the higher is the degree of asymmetry in the dataset.

Figure 23 shows that BNG2 and CNG2 are the most skewed datasets. The datasets with the lowest degree of skewness were those of proceeding with the formal university procedure in graded and non-graded assignments, which suggests that there was less variability in the values provided by academics in these two datasets.

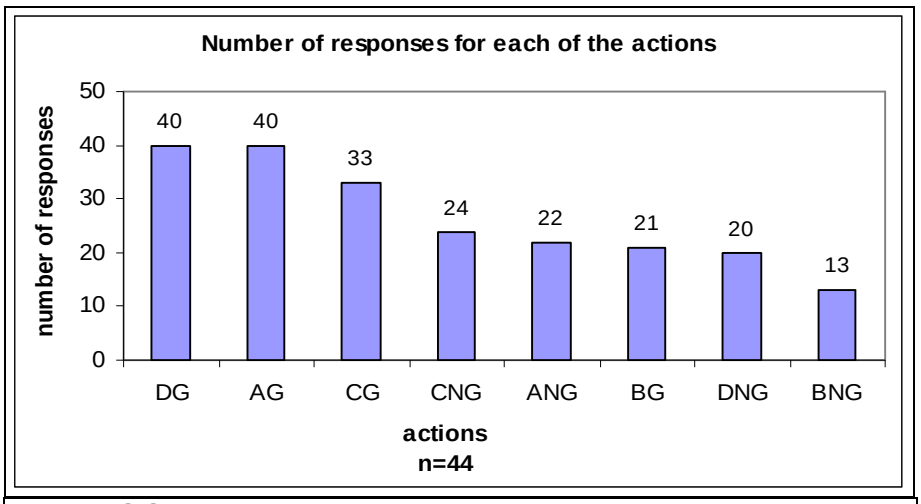
However, there are many factors that can influence the similarity values provided by academics, and there are described in *"Section 5.5: An investigation into the variability of similarity values in Questions 10 and 11"*. The vast majority of academics provided the same comments for questions 11a-11d as they did for questions 10a-10d and hence an analysis of the sub-questions 11a-11d was not statistically appropriate.

5.5 An investigation into the variability of similarity values in Questions 10 and 11

In this section, the responses academics have provided to the sub-questions of question 10 are analysed, in order to determine the factors that can influence the responses of academics when deciding on source-code similarity thresholds.

5.5.1 Responses to actions

Figure 24 shows that the number of responses varied for each of the actions. The variability in the number of responses occurred because not all academics would carry all of the suggested actions for the given scenario. Recall that the given scenario asked academics to provide similarity values regarding actions they would take if the source-codes submitted by two students for a programming assignment were similar.



AG - Graded assignments: No action
ANG – Non-graded assignments: No action
BG – Graded assignments: Note the student names for future checks
BNG – Non-graded assignments: Note the student names for future checks
CG – Graded assignments: Give warning to the students
CNG – Non-graded assignments: Give warning to the students
DG – Graded-assignments: Proceed with the university's formal plagiarism procedure
DNG –Non-graded assignments: Proceed with the university's formal plagiarism procedure

Figure 24: Number of responses for each action in question 10

The two most common actions that all academics can take in any scenario are to "take no action" and to "proceed with the formal university procedure". Whether the other actions (give warning to students, and note student names for future checks) are taken by academics depends on factors such as the academic regulations and the academics themselves.

There is a variety in the number of responses regarding the actions taken by academics in situations were students submit similar source-code. *Figure 24* above shows the actions in order of popularity. The majority, 40 out of 44, of the academics who have responded to this question would proceed with the formal plagiarism procedure (DG) if the source-codes reached a certain *ms value*. Comparing the number of academics who would proceed with the university formal plagiarism procedure for non-graded assignments (DNG) with the number of academics who would proceed with the university's formal plagiarism procedure for graded assignments the difference in responses is large. Less than half (20 out of 44) of the academics who have answered this question would proceed with the formal procedures if plagiarism was suspected in non-graded assignments, and nearly all (40 out of 44) of the academics who have answered this question would proceed with the formal procedures if plagiarism was suspected in graded assignments.

For graded assignments, actions "proceeding with the formal university procedure" and "Give warning to students" were the most popular, and for non-graded assignments the most popular actions were to "give warning to students" and "taking no action".

There is a variety in the number of responses concerning each of the actions as shown in *Figure 24*. Factors that can influence the similarity values provided by academics are discussed in *Section 5.5.2*.

5.5.2 Factors influencing the similarity values provided by academics

In order to determine some of the reasons that could cause variability in the *ms value* provided by academics, additional questions accompanied question 10. Below is a list of those sub-questions:

a. Did you answer this question with a specific module that you teach, in mind? (yes/no)
b. What level is the module (subject) that you had in mind (i.e. level 0 (foundation) / level 1 / level 2 / level 3 / level 4)?
c. What programming language (s) do you teach on that module (subject)?
d. Are there any other actions you might take? If so, please describe them and give the minimum and maximum similarity percentages.

The responses for questions *a, b, and c* are shown in *figures 25, 26,* and *27* respectively.

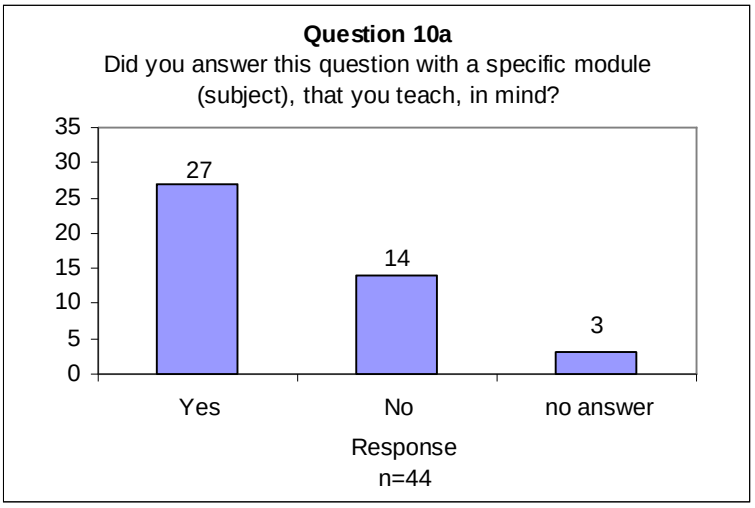


Figure 25: Responses for question 10a

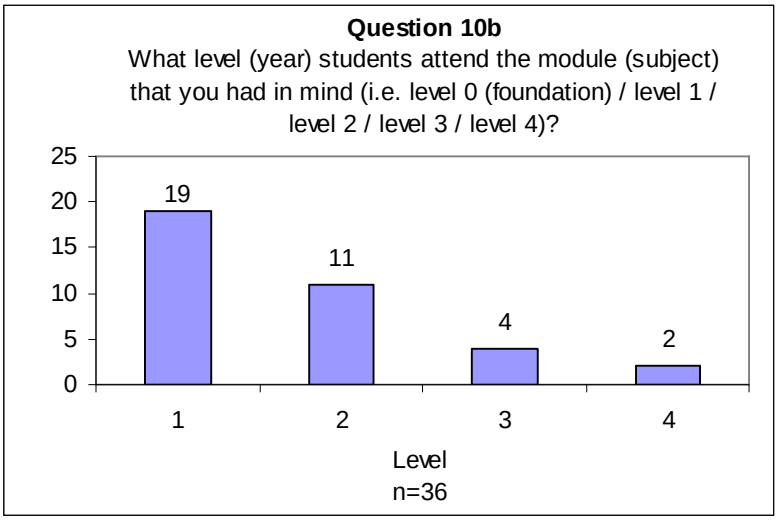


Figure 26: Responses for question 10b

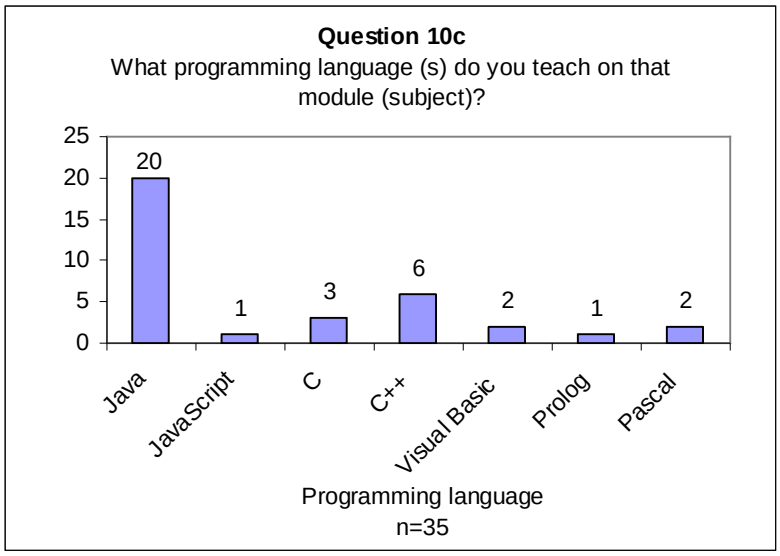


Figure 27: Responses for question 10c

Figure 25 shows that the majority of the academics (27 out of 44) responded to question 10 with a specific module(s) they teach in mind. The total number of responses shown in *Figure 26* is 36 because some of the academics have provided more than one module (and programming language) they had in mind when providing *ms value* for question 10.

Academics who responded to question 10 commented on factors that have affected their responses. Furthermore, many of the academics who did not provide an answer commented that in order to provide an answer they would need to take into consideration other factors. The factors the academics have provided are outlined in the list below:

- University regulations, i.e., some university regulations may not permit lecturers to take actions such as "give warning to students".
- The assignment's contribution towards the overall module mark
- The assignment's contribution towards the degree award
- The similarities between the source-codes in question. Source-code similarities such as those discussed in question 12 are described in *Section 5.7*.
- The student circumstances (i.e. whether they admit to the evidence presented to them, number of offences – 'first offence', 'repeat offence')
- The level of the module
- Proof that plagiarism has occurred in the source-codes in question

For sub-question 10d - "Are there any other actions you might take?" academics described different approaches to dealing with plagiarism. Many academics commented that they do not always take the same form of action. *"One would be more lenient on 'early' modules rather than advanced ones, and cases vary; there is not always a fixed form of action."*

Some academics would give a warning to first time offenders and others would proceed with the formal university procedures. One academic commented, *"I give a warning on the first offence no matter how serious the plagiarism is. Second offence would result in formal action being taken"*, a second academic explained further,

> *"There is no policy on this ... if we suspect plagiarism, no matter how small, it will always be investigated. The minimum outcome is always to warn the student, and any*

plagiarism is actionable within the University's formal procedures. The offence is sufficient in and of itself."

In addition some academics explained that they would only take the formal procedure if the evidence that the student plagiarised is clear, but the students do not admit to have committed plagiarism, *"The action depends on whether the students own up, and whether it's a first or second offence"*. Two other academics provided similar but more elaborated comments:

"The process in this institution is to present the issue to the students - who can then either confirm our suspicions or deny them. The formal disciplinary action would only take place if they deny it and the evidence is clear".

"... if the assignment is over 20% of the module marks, in which case it must go to formal procedures. I usually give the students the chance to come clean, if they do so, they get a warning from me, 0 mark (usually) and nothing further is done".

However, one academic who would proceed with formal procedures in the first instance of plagiarism commented, *"First offence is an interview and a resit assignment capped as a pass as maximum. Second offence is formal university procedure to consider expulsion"*, this was reinforced by a second academic:

"Regardless of severity, the formal procedure always kicks in if a suspicion is raised (i.e. 1 to 100%). The formal procedure has different categories for 'first offence', 'repeat offence', and the way these are dealt with depends on the level of the student (first years are treated within the dept, all others referred to central academic misconduct committee)."

5.6 Comparison of the paired values in questions 10 and 11 provided by academics

In the comparisons in the previous sections, it was apparent that in all pairs of actions some academics provided higher *ms value* for actions corresponding to non-graded assignments. In this section, a pair-wise comparison of the graded and non-graded *ms values* provided by academics is carried out in order to examine the statistical significance of this finding.

The responses of academics who have provided answers to actions for both graded and non-graded assignments were taken and analysed, and the rest of the values were excluded from the datasets. The purpose of performing pair-wise comparisons is to determine the correlations (relationships) between the values provided by academics regarding graded and non-graded assignments, and to determine whether there are differences between them.

In order to determine the correlations between the values in the graded and non-graded datasets the Spearman's and Pearson's correlation coefficient were both candidate tests. For the experiments regarding the comparison between groups of data, the T-test and the Wilcoxon signed-ranks test were both candidates. The Pearson's correlation coefficient measure and the T-test both assume that the distribution of the dataset is normal, whereas the Spearman's correlation coefficient measure and the Wilcoxon signed-ranks test do not place any assumptions on the distribution of the data.

As we have seen from the statistical analysis in the previous section, all the datasets were skewed some at a higher degree than others. The skewness of the distributions of the new datasets was examined in order to select appropriate statistics for carrying out the experiments. The box plot below compares the distributions of the new datasets.

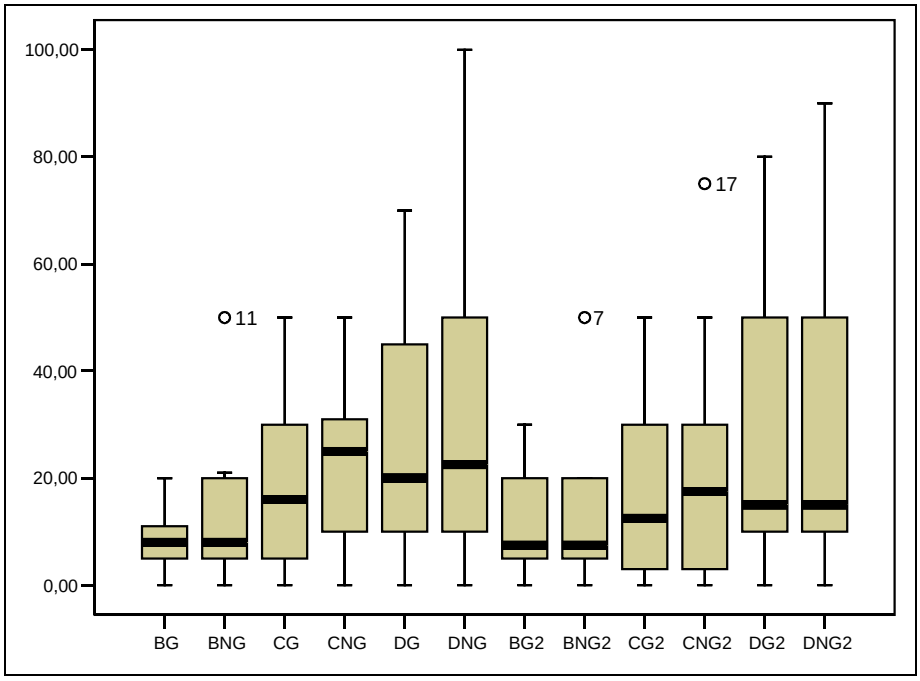


Figure 28: Box plot compares the datasets containing paired-values for questions 10 and 11

Statistics												
	BG	**BNG**	**CG**	**CNG**	**DG**	**DNG**	**BG2**	**BNG2**	**CG2**	**CNG2**	**DG2**	**DNG2**
N	14	14	21	21	20	20	10	10	20	20	21	21
Range	20	50	50	50	70	100	30	50	50	75	80	90
Minimum	0	0	0	0	0	0	0	0	0	0	0	0
Maximum	20	50	50	50	70	100	30	50	50	75	80	90
Percentiles												
25^{th}	5	5	5	10	10	10	5	5	4	4	10	10
50^{th}	8	8	16	25	20	22.5	7.5	7.5	12.5	17.5	15	15
75^{th}	10.75	18.75	30	31	42.5	50	18.75	18.75	30	30	50	50
IQR	5.75	13.75	25	21	32.5	40	13.75	13.75	26	26	40	40

Table 6: Statistics for figure 43

A comparison shows that either the lower or the upper-quartiles of actions corresponding to non-graded assignments are always higher than those quartiles of their paired graded assignments. This suggests that some academics have provided higher *ms values* for actions corresponding to non-graded assignments indicating that more similarity is accepted between non-graded assignments. In the pair-wise comparisons that follow, the paired values provided by academics are examined in order to determine the statistical significance of this finding. The skewness of the new datasets is shown in the figure below.

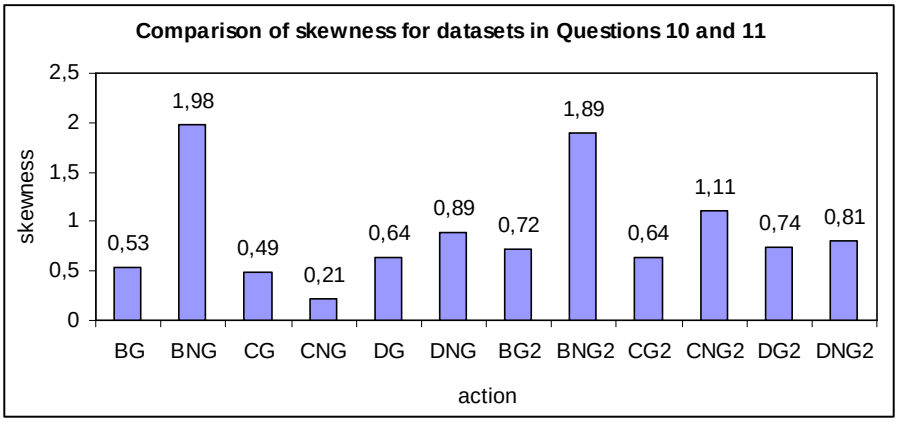


Figure 29: Skewness of the new datasets created for paired-value comparison

Figure 29 shows that the skewness values are high in most datasets suggesting that the distributions are asymmetric and positively skewed. Because the distributions are skewed, nonparametric tests were used to compare the means of the datasets. These tests were the Spearman rank correlation coefficient, and the Wilcoxon signed-ranks test.

Table 7 displays the sample size, mean, median, skewness and standard deviation for all datasets. Standard deviation measures the dispersion (spread) around the mean.

Descriptive statistics						
		N	**Mean**	**Median**	**Skewness**	**Std. Deviation**
Pair 1	BG	14	8.43	8.00	0.53	6.58
	BNG	14	12.00	8.00	1.98	13.19
Pair 2	CG	21	20.38	16.00	0.49	15.98
	CNG	21	23.24	25.00	0.21	16.83
Pair 3	DG	20	26.10	20.00	0.64	21.10
	DNG	20	31.85	22.50	0.89	28.49
Pair 4	BG2	10	11.00	7.50	0.72	9.94
	BNG2	10	13.00	7.50	1.89	14.94
Pair 5	CG2	20	18.15	12.50	0.64	17.11
	CNG2	20	20.65	17.50	1.11	20.03
Pair 6	DG2	21	28.90	15.00	0.74	26.08
	DNG2	21	29.86	15.00	0.81	28.04

Table 7: Descriptive statistics

The mean column of the *Table 7* shows that the mean values for non-graded datasets were in all pairs higher than graded assignments, and this suggests that across all datasets the academics have provided higher *ms values* for actions corresponding to non-graded assignments. These results suggest that some academics would accept a higher degree of similarity before taking any action if the source-codes in question (whether the source-codes of two students, or between a student and a book) were non-graded. For graded assignments, source-code thresholds would be lower.

The section below describes the Spearman Rank correlation test that was carried out in order to check whether there is a correlation between the values submitted by academics. More specifically, for each action we have the *ms value* for graded and non-graded assignments for *n* academics. The correlation measures the direction and strength of the linear relationship between two variables (in our case the variables are the values for actions given to graded and non-graded assignments).

5.6.1 Spearman's Rank Correlation Coefficient test

The Spearman's correlation coefficient test was carried out to determine the strength (or magnitude) and the direction of the correlation. A correlation of 0 or very close to 0 suggests that there is no association between the two variables compared. A correlation of +1 or -1 is a perfect correlation. The closer the correlation is to +1 or -1 the greater the association is between the two variables compared.

The direction of the correlation indicates the relation of the values. A positive correlation suggests a positive relationship between the two variables, and hence when one variable increases the other variable also increases. A negative correlation suggests a negative relationship between the two variables, and hence when one variable increases the other variable decreases.

Table 8 displays the Spearman's rank correlation coefficient values and the statistical significance value. The significance level calculated for each correlation provides information about the reliability of the correlation. The lower the significance value the higher the reliability.

Spearman Rank correlation for Paired Samples				
		N	Correlation	Sig. (2-tailed)
Pair 1	BG & BNG	14	0.91	0.00
Pair 2	CG & CNG	21	0.93	0.00
Pair 3	DG & DNG	20	0.91	0.00
Pair 4	BG2 & BNG2	10	1.00	0.00
Pair 5	CG2 & CNG2	20	0.87	0.00
Pair 6	DG2 & DNG2	21	0.99	0.00

Table 8: Spearman's Rank correlation coefficient for paired samples

The table above shows positive correlations for all pairs concerned. The high correlation values in all pairs suggest that there is a high association between the variables compared. The positive magnitude of the correlations suggests that as the graded value increases the non-graded value increases too.

Comparing the statistical significance of the correlations between each pair of action, the results show that there is a high statistical significance for all pairs concerned. The Spearman rank correlation values for all pairs is high and pair 4 has a perfect positive correlation. The correlation values of all pairs are highly significant ($p<0.01$). This clearly suggests that there was a high degree of consistency across the responses provided by academics regarding the relationship between the *ms values* they provided for actions corresponding to graded and non-graded assignments.

In conclusion, the results from the Spearman's rank correlation coefficient test show that strong positive correlations of high statistical significance exist for all pairs of datasets. To gather more information about the revealed correlations, and hence the relationship between the values provided by academics for graded and non-graded assignments, the Wilcoxon signed-ranks test was performed.

5.6.2 Wilcoxon signed-ranks Test

Nonparametric tests for two related samples are carried out to test for differences between paired scores without making assumptions about the distribution of the dataset. The Wilcoxon signed-ranks test is carried out to compare paired medians from the same (or matched) sample. The main reason for carrying out the Wilcoxon signed-ranks test is to examine in more detail the types of responses provided by academics rather than just the mean differences. The purpose of this test is to compare the responses for graded and non-graded assignments in each action group.

The Wilcoxon signed-ranks method tests the null hypothesis that two related medians are the same. In the Wilcoxon test, ranks are based on the absolute value of the difference between the two test variables. The sign of the difference is used to classify cases into one of three groups: differences below 0 (negative ranks), above 0 (positive rank), or equal to 0 (ties). Tied cases are ignored.

The hypotheses tested are:

- **H_0:** There is no significant difference between the *ms value* provided by academics for actions corresponding to graded and non-graded assignments.
- **H_A:** There is a significant difference in the *ms value* provided by academics for actions corresponding to graded and non-graded assignments**.**
- **Conclusion:** Reject null hypothesis if test statistic is less than 0.05.

Table 9, below, shows the number of ranks, mean rank, and the sum of ranks. The number of positive ranks is the number of academics that have provided a higher *ms value* for actions corresponding to non-graded assignments, and the number of negative ranks is the number of academics who have provided a higher *ms value* for actions corresponding to graded assignments. The number of ties is the number of academics that have provided the same *ms value* for both graded and non-graded assignments. The notes below the table indicate what the positive and negative ranks relate to. The chart in *Figure 30* illustrates the ranks shown in *Table 9*.

Wilcoxon Signed-ranks Test					
Ranks					
			N	Mean Rank	Sum of Ranks
Pair 1	BNG - BG	Negative Ranks	0(a)	0.00	0.00
		Positive Ranks	2(b)	1.50	3.00
		Ties	12(c)		
		Total	14		
Pair 2	CNG - CG	Negative Ranks	0(d)	0.00	0.00
		Positive Ranks	4(e)	2.50	10.00
		Ties	17(f)		
		Total	21		
Pair 3	DNG - DG	Negative Ranks	0(g)	0.00	0.00
		Positive Ranks	3(h)	2.00	6.00
		Ties	17(i)		
		Total	20		
Pair 4	BNG2 - BG2	Negative Ranks	0(j)	0.00	0.00
		Positive Ranks	1(k)	1.00	1.00
		Ties	9(l)		
		Total	10		
Pair 5	CNG2 - CG2	Negative Ranks	1(m)	1.00	1.00
		Positive Ranks	2(n)	2.50	5.00
		Ties	17(o)		
		Total	20		
Pair 6	DNG2 - DG2	Negative Ranks	1(p)	1.00	1.00
		Positive Ranks	2(q)	2.50	5.00
		Ties	18(r)		
		Total	21		

a. BNG < BG	j. BNG2 < BG2
b. BNG > BG	k. BNG2 > BG2
c. BNG = BG	l. BNG2 = BG2
d. CNG < CG	m. CNG2 < CG2
e. CNG > CG	n. CNG2 > CG2
f. CNG = CG	o. CNG2 = CG2
g. DNG < DG	p. DNG2 < DG2
h. DNG > DG	q. DNG2 > DG2
i. DNG = DG	r. DNG2 = DG2

Table 9: Wilcoxon Signed-ranks Test

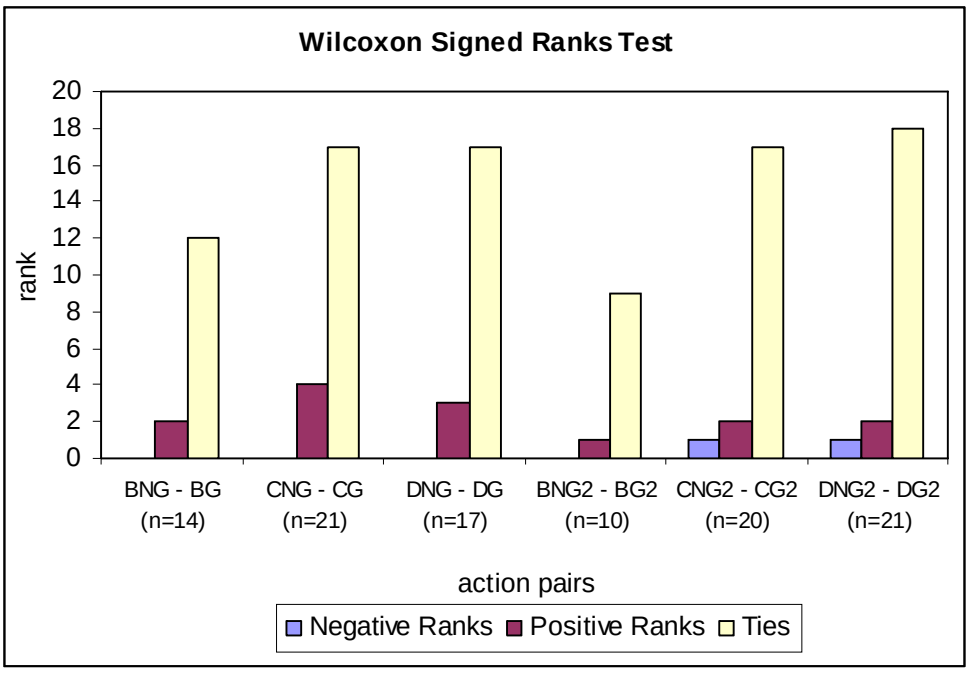


Figure 30: Bar chart shows the number of negative ranks, positive ranks and ties for each pair of actions

Table 9 shows that pairs 5 and 6 had one negative rank, and where the *ms values* given for the graded assignments were lower than those of graded assignment. This was the response of only one academic. In all pairs, the number of ties exceeds by a large proportion the number of academics who have provided other responses.

Table 10 below shows the z scores and the asymptomatic significance values. In the table below, if the significance value is below 0.05 then there is a significant difference in the *ms value* provided by academics, and if the value is above 0.05 then there is no significant difference between the values given for graded and non-graded assignments.

Wilcoxon Signed-ranks Test Statistics						
	BNG - BG	CNG - CG	DNG - DG	BNG2 - BG2	CNG2 - CG2	DNG2 - DG2
Z	-1.342(a)	-1.841(a)	-1.604(a)	-1.000(a)	-1.069(a)	-1.069(a)
Asymp. Sig. (2-tailed)	0.180	0.066	0.109	0.317	0.285	0.285

a Based on negative ranks.

Table 10: Test Statistics for the Wilcoxon signed-ranks Test

Since the significance value is greater than 0.05, we can assume that the null hypothesis holds. No significant difference was found between the *ms values* for actions corresponding to graded and non-graded assignments.

However, there is a possibility that no significant difference above 0.05 was found between the values for graded and non-graded assignments due to the small sample size for each of the actions. For this reason, the six pairs of datasets were merged into two datasets, one dataset contained all the values provided by academics for graded assignments and the other dataset consisted of the non-graded matched values. The new datasets consist of 106 pairs of values.

Spearman's rank correlation coefficient was performed in order to determine the magnitude and direction of the new datasets. The results are shown in *Table 11* below.

Spearman Rank correlation for Paired Samples	
	Graded - NonGraded
Correlation Coefficient	0.937(**)
Sig. (2-tailed)	0.000
N	106

** Correlation is significant at the 0.01 level (2-tailed).

Table 11: Spearman's Rank correlation coefficient for graded and non-graded pairs of values

The table shows positive correlations for the graded and non-graded datasets. The high correlation value suggests that there is a high association between the variables compared. The positive magnitude of the correlations suggests that as the graded value increases the non-graded value increases too.

The correlation value is highly significant ($p<0.01$). This clearly suggests that there was a high degree of consistency across the responses provided by academics regarding the relationship between the values they provided for actions corresponding to graded and non-graded assignments. In conclusion, the results from the Spearman's rank correlation coefficient test show that strong positive correlations of high statistical significance exist for actions corresponding to graded and non-graded assignments.

To gather more information about the relationship between the values provided by academics for graded and non-graded assignments, the Wilcoxon signed-ranks test was performed. The results from the Wicoxon paired-samples signed-rank test are shown below.

Ranks				
		N	**Mean Rank**	**Sum of Ranks**
NonGraded - Graded	Negative Ranks	2(a)	5.50	11.00
	Positive Ranks	14(b)	8.93	125.00
	Ties	90(c)		
	Total	106		

a NonGraded < Graded

b NonGraded > Graded

c NonGraded = Graded

Table 12: Wilcoxon signed-ranks Test

In the 106 pairs of *ms values* provided by academics, ninety academics provided the same *ms value* for actions corresponding to graded and non-graded assignments, fourteen academics provided a higher *ms value* for non-graded assignments, and only two academics provided a higher *ms value* for graded assignments.

In *Table 13* below, the significance value is less than 0.05 and hence we can conclude the null hypothesis does not hold. Therefore, there is a significant difference between the values provided by academics for actions corresponding to graded and non-graded assignments.

Wilcoxon Signed Ranks Test Statistics	
	NonGraded - Graded
Z	-2.956(a)
Asymp. Sig. (2-tailed)	0.003

a Based on negative ranks.

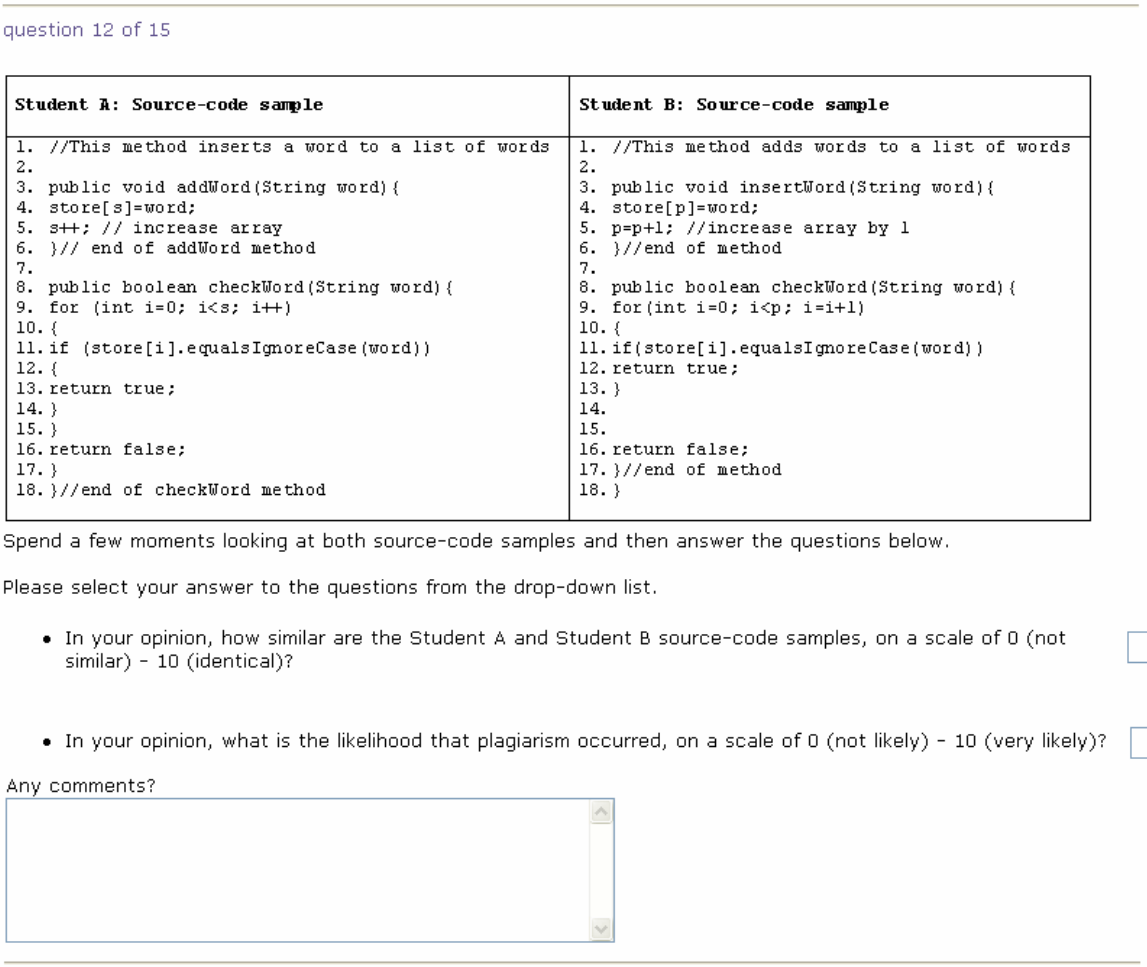
Table 13: Test Statistics for the Wilcoxon signed-ranks Test

In conclusion, although a large number of academics provided the same *ms value* for actions corresponding to graded and non-graded assignments, there are a significant number

of academics who provided a higher *ms value* for actions corresponding to non-graded assignments. This suggests that many academics would accept more similarity between non-graded assignments than they would if the assignments were graded before taking any form of action. Using the Wilcoxon pair-samples signed ranks-test, a pair-wise comparison of the graded and non-graded pairs of values was performed, and the results of the test show a statistical significance of this finding.

5.7 Question 12

In this question the scenario consists of two small pieces of source-code sample, and the academics were asked to rate the samples in term of similarity and the likelihood that plagiarism occurred. Question 12 is shown in *Figure 31*.


question 12 of 15

Student A: Source-code sample	Student B: Source-code sample

1. //This method inserts a word to a list of words
2.
3. public void addWord(String word){
4. store[s]=word;
5. s++; // increase array
6. }// end of addWord method
7.
8. public boolean checkWord(String word){
9. for (int i=0; i<s; i++)
10. {
11. if (store[i].equalsIgnoreCase(word))
12. {
13. return true;
14. }
15. }
16. return false;
17. }
18. }//end of checkWord method
	1. //This method adds words to a list of words
	2.
	3. public void insertWord(String word){
	4. store[p]=word;
	5. p=p+1; //increase array by 1
	6. }//end of method
	7.
	8. public boolean checkWord(String word){
	9. for(int i=0; i<p; i=i+1)
	10. {
	11. if(store[i].equalsIgnoreCase(word))
	12. return true;
	13. }
	14.
	15.
	16. return false;
	17. }//end of method
	18. }

Spend a few moments looking at both source-code samples and then answer the questions below.

Please select your answer to the questions from the drop-down list.

- In your opinion, how similar are the Student A and Student B source-code samples, on a scale of 0 (not similar) - 10 (identical)?
- In your opinion, what is the likelihood that plagiarism occurred, on a scale of 0 (not likely) - 10 (very likely)?

Any comments?

Figure 31: Question 12

The results are shown in *Table 14* below.

Response scale	Source-code similarity	Likelihood that plagiarism occurred
0	0	1
1	0	0
2	1	5
3	2	5
4	1	0
5	4	6
6	3	5
7	13	5
8	18	6
9	7	10
10	3	9

Table 14: Table shows the results from question 12

Fifty-two academics have answered this question. Of the seven academics who did not provide an answer, two provided comments. One academic commented that his answer to this question would depend on the assignment's specification and standards, and the source-code examples given to students during classes, and for this reason the academic felt that is was not possible to answer question with *"accuracy in isolation"*. A second academic who did not provide a response commented that the source-code fragments seem very suspicious, and the logic of the two programs is the same.

The results in *Table 14* show that the majority of the academics felt that the two source-code samples were very similar giving them a similarity value of 7 – 8. However, the responses regarding the likelihood that plagiarism has occurred were somehow fuzzy and the responses more distributed. To get a better view of the responses, the data in *Table 14* was placed into three smaller groups, such that 0-3 is low similarity, 4-6 is moderate and 7-10 is high similarity (*see Table 15*).

The chart shows that the majority of academics have found the two source-code samples to be very similar giving them a similarity rating of 7-10. Regarding the likelihood of whether plagiarism occurred, there was a variety in the answers given by academics. Twenty-five out of 44 academics selected a 'likelihood of plagiarism occurred' value between 8 and 10, and 35 out of the 52 academics selected a value between 6 and 10.

Grouped response scale	Source-code similarity	Likelihood that plagiarism occurred
0-3	3	11
4-6	8	11
7-10	41	30

Table 15: Table shows the grouped results from question 12

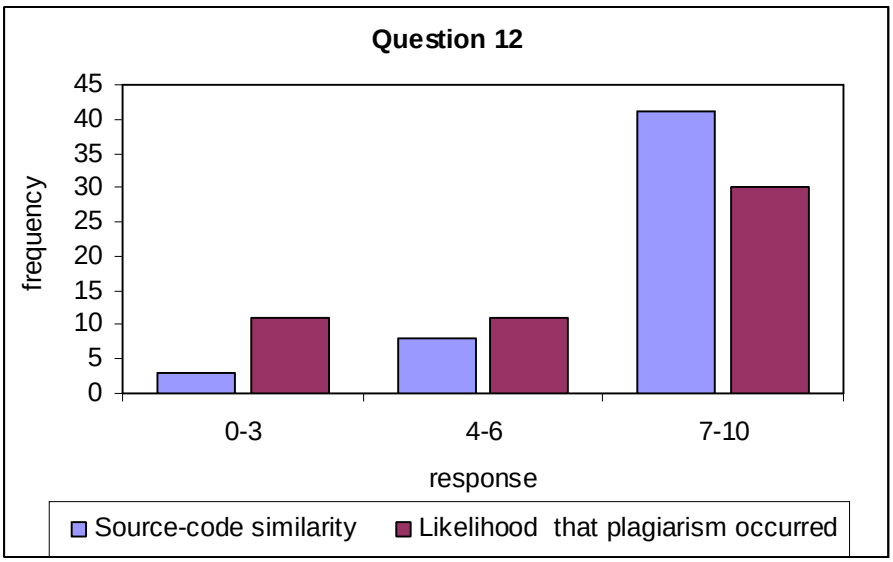


Figure 32: Chart shows the grouped results from the responses for Question 12

The results show that the academics have given a similarity value and a 'likelihood that plagiarism occurred' value between 7 and 10.

Academics commented that in order to make a decision about whether plagiarism occurred they would need to take into consideration the assignment requirements and standards, source-code examples given to the students and whether the problem has many solutions. One academic noted that *"Context matters. For example, was there a similar example in the lecture notes?"* this was reinforced by a second academic,

> *"The likelihood is very dependent on the context - the exact nature of the task they were set, any skeleton code given, and the nature of supporting examples the students were exposed to."*

Academics have characterised the source-code samples provided in question 12 are *"short", "simple", "trivial (unimportant)", "standard", "frequently published", "of limited functionality and solutions"* and hence they could not make an exact decision as to whether plagiarism has definitely occurred. One academic stressed,

> *"This is too small a sample to conclude, and the problem does not have sufficient variance in solution. If simple problems are asked, plagiarism will probably occur and is unlikely to be reliably detected."*

A second academic also emphasised,

> *"For small samples, it's very hard to tell. If the samples both matched the kind of code I'd used in lectures/labs, I'd be less inclined to think that the students had colluded."*

Some academics have commented that are that although the source-code fragments are small they are highly suspicious. Those academics observed,

> *"The formatting is dreadful and identical. A sure sign to me of plagiarism. It is rare for students to remember to put in comments, let alone such similar ones", "I regard deliberately attempting to conceal plagiarism as is done here to be much more serious than 'innocent plagiarism'."*

Furthermore, academics expressed common opinions on the issue of students producing similar methods that are limited in solutions, *"The programs have very limited functionalities and the solutions are bound to be similar", "The example is a bit too simple and frequently published to be sure that plagiarism occurred"*.

Respondents indicated which factors they considered to have provided evidence of similarity for the two code fragments.

- Lack of indentation, bad or identical indentation
- Both source-codes have the same location of white spaces between words i.e., the white spacing in store[s]=word are the same. There are many ways of writing this, for example, store [s] = word; store[s] = word, etc.
- Changed identifiers but same structure and logic the same
- Program logic is the same
- Source-code structure is the same
- Bad source-code formatting
- Similar source-code comments
- The same number of lines and each line has the same functionality
- Shared class variable (store)
- The fact that both students used an array (rather than a linked-list etc...)
- Similar mistakes in the two source-codes can indicate collusion and plagiarism
- Similarity between unusual lines of code
- Lexical, syntactical, grammatical and structural similarities

It was observed that source-code plagiarism can only be *proven* by observing the pieces of source-code that serve *important program functionality* and that are the student's own unique approach to solving the specific problem. Such programs will contain methods which to some extend are unique in program logic, approach and functionality.

Small pieces of code that are likely to be similar in many solutions can be used to examine further the likelihood that plagiarism has occurred, but alone they may not be sufficient for proving plagiarism. In a simple and tightly specified problem, the program (solution) submitted by students may be very similar due to the nature of the problem and not due to plagiarism. Furthermore, common opinions were expressed on the issue of students producing similar methods that are limited in solutions. For example:

> *"Small O-O methods are almost bound to be very similar, especially if you have discussed a previous analysis (e.g. Abstract Data Type definitions) and have used a standard development method. In my experience students can submit virtually identical work without improper (or any) collusion."*

This was reinforced by another academic who also noted the effect on the threshold at which an academic offence is considered to have occurred:

> *"... a simple, tightly specified problem (which the words program could well be) is expected to result in many similar submissions, so the similarity threshold for invoking the plagiarism procedures would be higher."*

5.8 Summary of Part Two

The majority of academics believe that investigation into possible plagiarism should always be pursued and appropriate action should be taken (whether penalties or warning) regardless of the assignment's contribution towards the overall module mark. However, factors such as university regulations may restrict academics from proceeding with formal plagiarism procedures as these may only apply to assignments that contribute towards the overall module mark (or course grade). The assignment's contribution towards the overall module mark does not seem to influence the decision of academics whether they should proceed with investigation into possible plagiarism.

A large number of academics have provided the same *ms value* for actions corresponding to graded and non-graded assignments, however a significant part of the sample has provided a higher *ms value* for actions corresponding to non-graded assignments. This suggests that many academics would accept more similarity between non-graded assignments than they would if the assignments were graded before taking any form of action. A pair-wise comparison using the Wilcoxon pair-samples signed ranks-test has proven the statistical significance of this finding.

There was variability in the *ms values* provided by academics. The results show that the majority of the academics have provided an *ms value* with one or more specific modules that they teach in mind. This suggests that the *ms value* provided by academics are very subjective and depend on various factors such as university regulations, assignment specification, student circumstances, and the actual similarity between the source-code submitted by the students. In addition, the actions taken by academics depend on the university regulations as well as the academics themselves. For example, actions such as noting student names for future checks and warning students may not appear in university regulations but academics may undertake those actions when they consider it appropriate to do so.

When comparing two source-code fragments for similarity, academics may look for lexical and structural similarities that suggest evidence of plagiarism. Once similarities are detected between two pieces of source-code, there may be circumstances in which whether source-code plagiarism occurred depends on factors such as the nature of the problem, the possible solutions to the problem and material students are given in class. For example, in a simple and tightly specified problem, the programs (solutions) submitted by students may be very similar due to the nature of the problem and not due to plagiarism.

In conclusion, source-code plagiarism can only be proven by observing the pieces of source-code that serve important program functionality and that are a student's own unique approach to solve the specific problem. Such programs will contain methods with to some extend are unique in program logic, approach and functionality.

6.0 Conclusion

There exists survey-based research regarding the prevalence of source-code plagiarism in academia. However, we are not aware of surveys on the issue of what constitutes source-code plagiarism in UK universities.

In this report we have presented a detailed analysis of the responses gathered from a survey conducted with UK academics who teach programming on computing courses. The aim of the survey was to establish what is understood to constitute source-code plagiarism in an undergraduate context.

There appears to be no commonly agreed description on what constitutes source-code plagiarism from the perspective of academics who teach programming on computer courses. The responses of academics to the questions provided in the survey, along with their generous comments, which provided an insight into their perspectives regarding source-code plagiarism, have enabled the creation of a detailed description on the issue of what constitutes source-code plagiarism from the academics point of view.

There is a general agreement that a ‘zero tolerance’ policy is appropriate, but there are other issues for which there is a divergence of opinions, including source-code use and acknowledgement, self-plagiarism, and differences between the approach to plagiarism in graded and non-graded work.

The differences in opinion on the issues of source-code use and acknowledgement were concerned with whether source-code reuse without acknowledgment, and resubmission of source-code produced for another assignment without acknowledgment, can constitute self-plagiarism.

There was a general agreement between academics that students should acknowledge parts of the source-code that were not originally authored by them, although some academics have expressed uncertainties whether reuse without acknowledgement constitutes plagiarism.

Regarding the issue of whether it constitutes plagiarism when students reuse (without providing acknowledgement) source-code they have produced as part of another assignment, the majority of academics consider resubmission of source-code without acknowledgement as an academic offence, however a small but significant number of academics disagree due to source-code reuse being encouraged in object-oriented programming.

Although two pieces of source-code code may contain structural and lexical similarities, there are other factors that could influence their judgement as to whether plagiarism occurred. Such factors include source-code templates given to students and the possible solutions to the fragments of source-code that are being compared.

REFERENCES

Bull, J., et al. (2001*). Technical review of plagiarism detection software report. CAA centre*. University of Luton. Available at: http://www.jisc.ac.uk/uploaded_documents/luton.pdf [accessed 21 February 2006].

Culwin, F. MacLeod, A. & Lancaster, T. (2001). *Source-code Plagiarism in UK HE Computing Schools, Issues, Attitudes and Tools*. Technical Report No. SBU-CISM-01-02. South Bank University.

Decoo W. (2002). *Crisis on campus: Confronting academic misconduct*. MIT Press Cambridge, England. ISBN:0-262-04201-0

Donaldson, John L., Lancaster, Ann-Marie & Sposato, Paula H. (1981), A Plagiarism Detection System, *Twelfth SIGSCE Technical Symposium*, St. Louis, Missouri, pp. 21-25.

Faidhi and Robinson. (1987), An empirical approach for detecting program similarity within a university programming environment. *Computers and Educatio*n, vol. 11, no. 1, pp. 11-19.

Gomes L. (2006). Some Students Use Net To Hire Experts to Do Their School Work. *The Wall Street Journal [Online]*. Available at: http://online.wsj.com/public/us [accessed 21 February 2006].

Halstead M. H. (1972), Natural laws controlling algorithm structure?, A*CM SIGPLAN Notices*, vol.7 no.2, pp.19-26.

Halstead M. H. (1977), *Elements of Software Science*, Elseveir.

JavaServer Pages Technology (2006). Sun Microsystems. Available at: http://java.sun.com/products/jsp/ [accessed 21 February 2006].

Joy, M. & Luck, M. (1999). Plagiarism in Programming Assignments, *IEEE Transactions on Education*, vol. 42, no. 2, pp. 129-133.

JSPMaker v1.0.1. (2006). [Computer Software]. e.World Technology Ltd. Available at: http://www.hkvstore.com/jspmaker/ [accessed 21 February 2006].

Maxwell Nicholas (2004), D*ata matters: Conceptual Statistics for a Random World*, Key College Publishing.

Ottenstein K. J. *A Program to Count Operators and Operands for ANSI-Fortran Modules*, IBM Tech. Report CSD-TR-196, June 1976.

Ottenstein J. K (1976b), An algorithmic approach to the detection and prevention of plagiarism, *ACM SIGCSE Bulletin*, vol. 8 no. 4, pp.30-41.

Prechelt, L. Malpohl, G. & Philippsen, M. (2000). *JPlag: Finding Plagiarisms Among a Set of Programs.* Technical Report 2000-1. Facultät für Informatik, Universität Karlsruhe, Germany.

Prechelt, L. Malpohl, G. & Philippsen, M. (2002). Finding Plagiarisms among a Set of Programs with JPlag. *J. UCS*, vol.8, no.11, pp. 1016-1038.

Robinson, Sally S. & Soffa, M. L. (1980). An Instructional Aid for Student Programs. *SIGCSE Bulletin,* vol. 12, no. 1, pp. 118-129.

Sutherland-Smith, W. (2005). Pandora's box: academic perceptions of student plagiarism in writing. *Journal of English for Academic Purposes*, vol.4, pp. 83-95.

Verco, Kristina L. & Wise, Michael J. (1996a), Software for Detecting Suspected Plagiarism: Comparing Structure and Attribute-Counting Systems, *Proceedings of First Australian Conference on Computer Science Educatio*n, July 3-5 1996, Sydney, Australia.

Verco, Kristina L. & Wise, Michael J. (1996b), Plagiarism á la Mode: A Comparison of Automated Systems for Detecting Suspected Plagiarism. *The Computer Journal* vol. 39, no.9. pp. 741-750.

Whale, G. (1990a). Identification of Program Similarity in Large Populations, *Computer Journal*, vol.33, no.2. pp 140-146.

Whale G (1990b), Software Metrics and Plagiarism Detection, J. *Systems and Software*, vol. 13. no.2. pp 131-138.

Wise M (1992), Detection of similarities in student programs: YAP'ing may be preferable to plague'ing, *SIGCSE*, Proceedings of the twenty-third SIGCSE technical symposium on Computer science education.

Wise, M. J. (1996). YAP3: Improved Detection of Similarities in Computer Program and Other Texts, *SIGCSE Technical Symposium*, Philadelphia, USA, pp. 130-134.

APPENDICES

Appendix A: Source-Code Plagiarism Survey

Source-code plagiarism survey

Thank you for taking the time to participate in this online survey.

We are conducting this survey to gather the perspectives of academics on what constitutes source-code plagiarism in programming assignments.

All the information that you provide in this questionnaire will be kept confidential. Further, neither you nor your institution will be identified in the thesis or in any report or publication based on this research. There are no known or anticipated risks to participation in this study.

Thank you very much for your co-operation in our research and we look forward to your response.

Georgina Cosma
PhD student
Department of Computer Science
University of Warwick, Coventry, CV4 7AL, UK

If you have any queries about this study, or would like additional information about completing this questionnaire please feel free to contact us at g.cosma@warwick.ac.uk.

INSTRUCTIONS

Please read the instructions below before proceeding with the questionnaire.

1. You must currently be an academic and teaching (or have previously taught) on at least one programming subject to participate in this survey.
2. You will not need to consult with anyone when completing the survey. We are interested in your individual responses to these questions.
3. You will notice that there are some similar questions with similar wordings. However, these questions are different in important ways and please consider each question separately as you answer them.
4. Note that there are no right or wrong answers to any of the questions in this survey.

IMPORTANT NOTES

Please read the notes below before proceeding with the questionnaire.

1. Students are required to acknowledge (reference) any material they use that is not their own original work.

2. Students are required to provide acknowledgements (references) for any material they use regardless of the licensing permissions of that material (e.g. open source, free-use, fair-use).

3. We are interested in similar source-code based on textual information irrespective of any background information about the student, for example, whether or not they have plagiarised before.

Section 1: Plagiarism Methods

In this section we will describe small scenarios about ways students have used material from sources, such as books, and we will ask you to select the type of academic offence that applies to each scenario.

Note: Throughout the survey, you will come across the term "academic offence". An "academic offence" is an action by a student that breaks the university rules. Examples of academic offences include plagiarism, cheating, falsifying data, handing in the same work more than once, etc. Committing an academic offence can lead to penalties.

question 1 of 15

Plagiarism in programming assignments can involve the:

Please answer each question by filling in the appropriate circle.

• Source-code of a computer program.	Agree	Disagree	Neither agree or disagree
• Comments within the source-code.	Agree	Disagree	Neither agree or disagree
• Design material of a computer program.	Agree	Disagree	Neither agree or disagree
• Documentation of a computer program.	Agree	Disagree	Neither agree or disagree
• User-interface of a computer program.	Agree	Disagree	Neither agree or disagree
• Program input data, i.e. for testing the program	Agree	Disagree	Neither agree or disagree

Any comments?

question 2 of 15

Please read each of the scenarios below, and select the academic offence that, in your opinion, applies to each scenario.

Note: By 'someone else's...' we mean student or other author (i.e. book author, web-page author, etc.)

Please answer each question by filling in the appropriate circle.

• A student reproduces/copies someone else's source-code without making any alterations and submits it without providing any acknowledgements.	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know
• A student reproduces/copies someone else's source-code, adapts the code to his/her own work and submits it without providing any acknowledgements.	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know
• A student converts an entire or part of someone else's source-code to a different programming language and submits it without providing any acknowledgements.	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know
• A student uses code-generating software (software that one can use to automatically generate source code by going through wizards), and removes the acknowledgement comments that were automatically placed into the code by the software and submits it without providing any acknowledgements.	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know

Any comments?

---

question 3 of 15

Please read each of the scenarios below, and select the academic offence that, in your opinion, applies to each scenario.

Please answer each question by filling in the appropriate circle.

• A student pays another person (other than a student on the same module) to create part or whole of source-code and submits it as his/her own work.	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know
• A student pays a fellow student on the same module to create part or whole of source-code and	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know

submits it as his/her own work.

- A student steals another student's source-code and submits it as his/her own work. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

- A student steals another student's source-code, edits it and submits it as his/her own work. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

- A student intentionally permits another student to copy all or part of his/her programming assignment (including the source-code). ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

Any comments?

question 4 of 15

Please read each of the scenarios below, and select the academic offence that, in your opinion, applies to each scenario.

Please answer each question by filling in the appropriate circle.

- For a group assignment, students between different groups exchange parts of source-code with the consent of their fellow group members, and integrate the borrowed source-code within their work as if it was that group's own work. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

- For a group assignment, students between different groups exchange parts of source-code, without their fellow group members knowing, and integrate the borrowed codes within their work as if it was that group's own work. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

- Assume that students were not allowed to resubmit material they had originally created and submitted previously for another assignment. For a graded assignment, a student has copied ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

parts of source-code that he had produced for another assignment without acknowledging it.

- Two students work together for a programming assignment that requires students to work individually and the students submit very similar source-codes. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

Any comments?

Section 2: Acknowledging Sources

In this section we will describe small scenarios about how students have acknowledged (referenced) material from sources, such as books, and we will ask you to select the type of academic offence that applies to each scenario.

question 5 of 15

Please read each of the scenarios below, and select the academic offence that, in your opinion, applies to each scenario.

Please answer each question by filling in the appropriate circle.

- For a graded assignment, a student has copied source-code from a book and has intentionally not provided any acknowledgements. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know
- For a graded assignment, a student has copied source-code from a book and has unintentionally not provided any acknowledgements. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know
- For a non-graded assignment, a student has copied source-code from a book and has intentionally not provided any acknowledgements. ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know
- For a non-graded assignment, a student has copied source-code from a book and has unintentionally not provided any ○ Plagiarism (or a type of plagiarism) ○ Other academic offence ○ Not an academic offence ○ Don't know

acknowledgements.

Any comments?

question 6 of 15

Please read each of the scenarios below, and select the academic offence that, in your opinion, applies to each scenario.

Please answer each question by filling in the appropriate circle.

Copying source-code from a book or a website, and:

	Plagiarism (or a type of plagiarism)	Other academic offence	Not an academic offence	Don't know
• Not providing any acknowledgements.	○	○	○	○
• Providing pretend references (i.e. references that were made-up by the student and that do not exist).	○	○	○	○
• Providing false references (i.e. references exist but do not match the source-code that was copied).	○	○	○	○
• Modifying the program output to make it seem as if the program works.	○	○	○	○

Any comments?

question 7 of 15

In your experience, what is the likelihood of plagiarism occurring when two students:

- Have a brief discussion about the program design, (i.e. what classes, attributes, methods they need) without taking any notes.
- Have a detailed discussion about the program design,(i.e. what classes, attributes, methods they need) and take notes and use these notes when designing the program.
- Work together and share ideas while producing the design of the program.
- Have a brief discussion about the program functionality and source-code without taking any notes.
- Have a detailed discussion about the program functionality and source-code and take notes and use these notes when coding the program.
- Work together and share ideas while coding the program.
- Work separately and do not discuss the design or functionality of the program, but work together and help each other during the program testing and debugging stage.

Any comments?

Section 3: Document Similarity

In this section we are interested in your views on quantitative issues regarding similar source-code documents.

question 8 of 15

Assume that a student has taken source-code from a book or from another source (i.e. paper, website, etc) without providing any acknowledgements. What would have to be the minimum weight (in percentage) of the work in question towards the overall module mark , for you proceed with investigation into possible plagiarism?

Please fill in the blank.

Minimum weight %

Any comments?

question 9 of 15

Assume that two students have submitted very similar source-code. What would have to be the minimum weight (in percentage) of the work in question towards the overall module mark, for you proceed with investigation into possible plagiarism?

Please fill in the blank.

Minimum weight [ ] %

Any comments?

question 10 of 15

The source-codes submitted by two students for a programming assignment are similar.

What percentage of the source-codes must be similar for you to take the actions listed in the table below.

Important note: Please leave the fields blank for the actions, listed below, that you do not take. Describe additional actions in the comments box provided.

	Graded assignment	Non-graded assignment
No action	Give minimum amount [0] % Give maximum amount [ ] %	Give minimum amount [0] % Give maximum amount [ ] %
Note the student names for future checks, but do not take any further action and do not contact the students.	Give minimum amount [ ] %	Give minimum amount [ ] %

	Give maximum amount %	Give maximum amount %
Give warning to the students.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Proceed with the university's formal plagiarism procedure.	Give minimum amount % Give maximum amount 100 %	Give minimum amount % Give maximum amount 100 %

Did you answer this question with a specific module (subject), that you teach, in mind? (Yes / No)

If you have answered YES to the question above,

what level is the module (subject) that you had in mind (i.e. level 0 (foundation) / level 1 / level 2 / level 3 / level 4) :

what programming language (s) do you teach on that module (subject)?

Are there any other actions you might take? If so, please describe them and give the minimum and maximum similarity percentages.

---

question 11 of 15

For an assignment, students were allowed to use source-code from books and other information sources as long as they provided acknowledgements.

Assume that a student has taken source-code from a book (or from other resources) without providing any acknowledgements for the sources s/he used.

What percentage of the source-code must consist of unacknowledged source-code for you to take the following actions:

Important note: Please leave the fields blank for the actions, listed below, that you do not take. Describe additional actions in the comments box provided.

	Graded assignment	Non-graded assignment
No action	Give minimum amount	Give minimum amount

	0 % Give maximum amount %	0 % Give maximum amount %
Note the student's name for future checks, but do not take any further action and do not contact the student.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Give warning to the student.	Give minimum amount % Give maximum amount %	Give minimum amount % Give maximum amount %
Proceed with the university's formal plagiarism procedure.	Give minimum amount % Give maximum amount 100 %	Give minimum amount % Give maximum amount 100 %

Did you answer this question with a specific module (subject), that you teach, in mind? (Yes / No)

If you have answered YES to the question above,

what level (year) students attend the module (subject) that you had in mind (i.e. level 0 (foundation) / level 1 / level 2 / level 3 / level 4) :

what programming language (s) do you teach on that module (subject)?

Are there any other actions you might take? If so, please describe them and give the minimum and maximum similarity percentages.

question 12 of 15

Student A: Source-code sample	Student B: Source-code sample

1. //This method inserts a word to a list of words
2.
3. public void addWord(String word){
4. store[s]=word;
5. s++; // increase array
6. }// end of addWord method
7.
8. public boolean checkWord(String word){
9. for (int i=0; i<s; i++)
10.{
11.if (store[i].equalsIgnoreCase(word))
12.{
13.return true;
14.}
15.}
16.return false;
17.}
18.}//end of checkWord method
	1. //This method adds words to a list of words
	2.
	3. public void insertWord(String word){
	4. store[p]=word;
	5. p=p+1; //increase array by 1
	6. }//end of method
	7.
	8. public boolean checkWord(String word){
	9. for(int i=0; i<p; i=i+1)
	10.{
	11.if(store[i].equalsIgnoreCase(word))
	12.return true;
	13.}
	14.
	15.
	16.return false;
	17.}//end of method
	18.}

Spend a few moments looking at both source-code samples and then answer the questions below. Please select your answer to the questions from the drop-down list.

- In your opinion, how similar are the Student A and Student B source-code samples, on a scale of 0 (not similar) - 10 (identical)?

- In your opinion, what is the likelihood that plagiarism occurred, on a scale of 0 (not likely) - 10 (very likely)

Any comments?

Section 4: Feedback

In this section we are interested in your comments about this survey.

question 13 of 15

If there are any issues about 'plagiarism in programming assignments' that are not covered by the questionnaire, or if you would you like to provide any comments about the questionnaire, please do so in the space below, or email us at g.cosma@warwick.ac.uk .

question 14 of 15

Please answer each question by filling in the appropriate circle.

Would you like to be informed about the outcome of this survey? Yes No

Would you be prepared to participate in further surveys regarding plagiarism? Yes No

question 15 of 15

If you have answered ' Yes ' to question 14, please type in your details below so that we can contact you, or alternatively email us at g.cosma@warwick.ac.uk with your contact details.

OPTIONAL: Please type in your details.

Your name	
Your job title	
Your e-mail address	
Name of university	
Name of department (School)	

Thank you for taking the time to complete this survey.

Appendix B: Plagiarism techniques in programming assignments

Whale (1990a), Joy and Luck (1999), Prechelt (2000), Wise (1996) describe techniques that students use to hide plagiarism. Faidhi and Robinson (1987) have categorised the possible techniques that students use when plagiarising. Below, is an exhaustive list of possible techniques that students may use when plagiarising programming assignment. The list was created using the sources mentioned above, and from personal experiences created from observing plagiarised sets of source-code submitted by students.

Source-code modifications

Method modifications

1. Modifying the method names
2. Modifying the method body
3. Modifying return values of functions
4. Modifying method arguments
5. Modifying, deleting or modifying modifiers (i.e. private, public, protected, final) of methods
6. Reordering functions
7. Reordering code within functions
8. Replacing method calls with the method body

Identifier (and variable) modifications

9. Modifying the names of identifiers
10. Modifying the types of identifiers
11. Modifying the position of identifier declarations
12. Modifying the attribute declarations from global to local and vice-versa
13. Modifying or deleting modifiers (i.e. private, public, protected, final) in variable declarations
14. Modifying the values assigned to variables (identifiers that can have a value)
15. Splitting/merging identifier declarations

Iteration and Selection statement modifications

16. Modifying conditions to equivalent conditions
17. Modifying statements to equivalent statements
18. Modifying the order of the selection statement
19. Converting a selection statement to an equivalent (i.e. an if to a switch statement)
20. Adding more execution paths

Mathematical expression modifications

21. Modifying the order of the operands (e.g. $x < y$ can become $y>x$)
22. Modifying mathematical expressions but still get the same output
23. Combining multiple mathematical expressions

Data structure modifications

24. Modifying data structures (i.e. using a character array instead of a string)
25. Converting code written using vectors to code using arrays
26. Modifying searching and sorting algorithms

Class modifications

27. Swapping methods between classes
28. Modifying the name of a class

Functionality modifications

29. Introducing bugs on purpose or not on purpose
30. Fixing bugs
31. Adding functions and code that are never called

Documentation modifications

Source-code comment modifications

32. Modifying (rewording) the comments in the code
33. Modifying the position of the comments
34. Modifying surrounding comments to line comments and vice-versa
35. Splitting/merging comments

Source-code documentation modifications

36. Program specifications
37. User manuals
38. Technical manuals
39. Source-code accompanied documentation
40. Testing documentation

User Interface modifications

Interface appearance modifications

41. Modifying colours/fonts etc
42. Modifying the user interface text (see free text plagiarism)
43. Modifying the layout of the objects on the interface

User input modifications

44. Requesting data to be inputted by the user in a different format e.g. 01/July/2005 instead of 01/07/2005
45. Requesting data to be inputted by the user in a different order
46. Requesting extra or redundant data to be inputted

Program output modifications

47. Modifying the appearance of the outputted data

Program implementation modifications

48. Converting a program to a different programming language
49. Converting code into an application
50. Embedding code into a webpage

Program design modifications

51. Modification to design elements of a program e.g., UML diagrams, program architecture and framework, etc.