1991

An Expert Inference Engine for Generation of Nursing Diagnoses

Tom Edgar
University of North Florida

AN EXPERT INFERENCE ENGINE FOR GENERATION OF NURSING DIAGNOSES

by

Tom Edgar

A thesis submitted to the
College of Computer and Information Sciences
in partial fulfillment of the requirements for the degree of

Master of Science in Computer and Information Sciences

UNIVERSITY OF NORTH FLORIDA
COLLEGE OF COMPUTER AND INFORMATION SCIENCES

May, 1991

The thesis "An Expert Inference Engine for Generation of Nursing Diagnoses", submitted by Tom Edgar in partial fulfillment of the requirements for the degree of Master of Science in Computer and Information Sciences has been

Approved by the thesis committee: Date

Signature Deleted

Thesis Adviser and Committee Chairman 5/3/91

Signature Deleted

5/3/91

Signature Deleted

5/3/91

Accepted for the College of Computer and Information Sciences:

Signature Deleted

Interim Dean 5/3/91

Accepted for the University:

Signature Deleted

Vice-President for Academic Affairs 5-7-91

ACKNOWLEDGEMENT

I wish to thank the thesis committee chairman, Dr. J. E. Leitner and the other members of the committee for their patience and support during the development and validation of this project. In addition, I wish to especially thank Kathaleen C. Bloom, who provided expert guidance in the area of nursing diagnosis and who worked closely with me in validating this project.

CONTENTS

TABLE OF CONTENTS (continued)

FIGURES and TABLES

ABSTRACT

Expert computer systems for use in the nursing profession are emerging as a potentially viable alternative to manual procedures. As nursing science continues to develop, the intellectual requirements of assessment and diagnosis are demanding that the professional nurse draw on an ever-increasing bank of knowledge to interact effectively with clients. An expert system appears a promising tool to assist the nurse in storing and accessing some of the knowledge necessary to perform the assessment and diagnostic functions.

Problems and opportunities in applying artificial intelligence techniques to nursing science are documented and the current state of expert systems for nursing are explored.

A new expert system is developed utilizing artificial intelligence to aid the nurse in performing nursing diagnosis. Employing Prolog on an IBM PC computer, the expert system references client cues found during a nursing assessment and proposes appropriate nursing diagnoses based on those cues. The system is then validated against a human, "expert" nurse to determine its soundness and usefulness.

Chapter 1

INTRODUCTION

Figure 1 is a graphical representation of the relationships between the steps of the nursing process as described by Alfaro [Alfaro86]. When a client enters a health care setting,

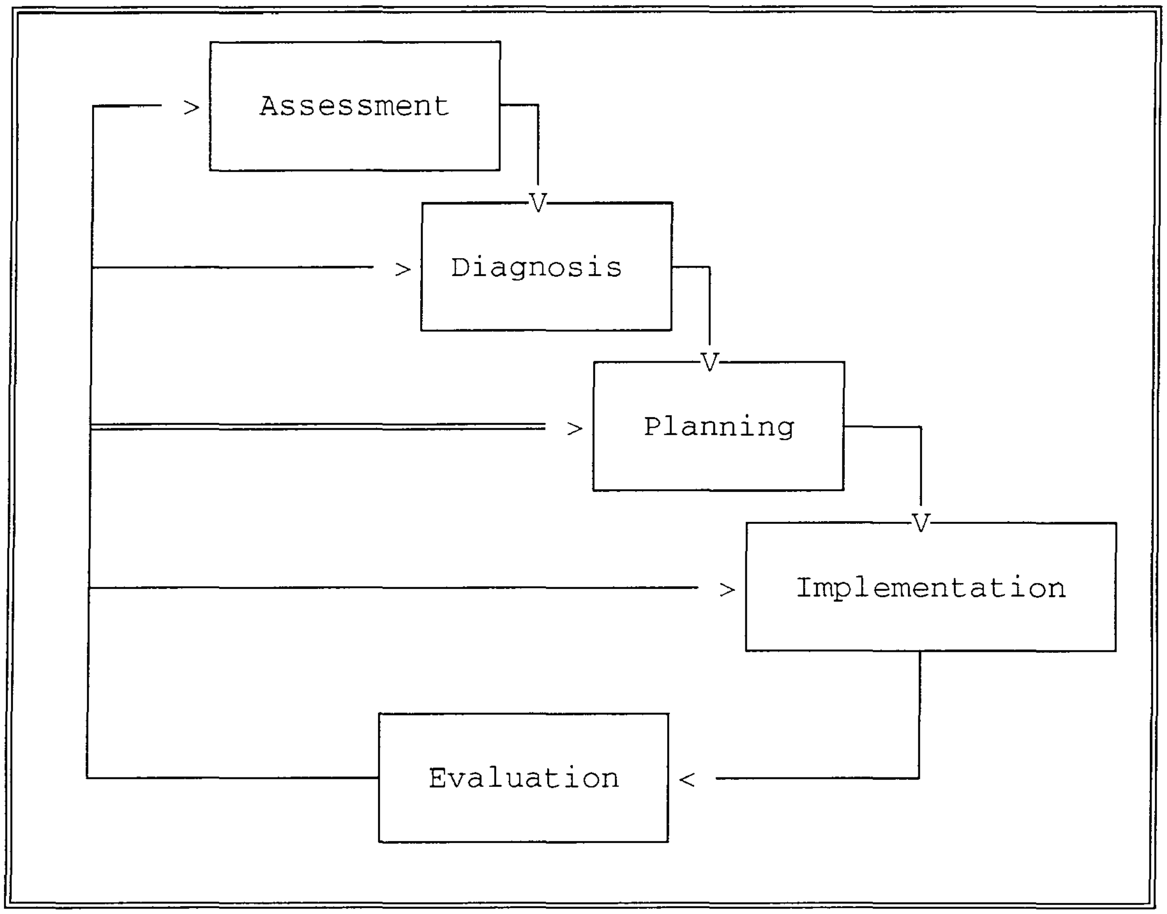


Figure 1: Relationships Between the Steps of the Nursing Process

whether it is a hospital, clinic, doctor's office, or the client's home, the professional nurse employs systematic observational and problem-solving techniques to identify the client's health status. These techniques begin by assessing the current conditions present and developing scenarios of possible or potential problem areas that may be indicated. After a reasonable list of problems have been developed, maintenance of healthful states or intervention to correct less than optimum states is planned, implemented, and the results evaluated. Each step in this process is dependant on the accuracy and completeness of each preceding step. Since clients are continually interacting with their environment, the nurse must apply the process in a cyclical manner. That is, she must evaluate client progress and possibly reenter the process to account for a changing environment or to correct deficiencies.

Nurses must make judgements regarding a variety of assessment data. Diagnosis involves complex thinking about the assessment data gathered from the client, family, records, and other health care providers. This thinking is combined with relevant information stored in the nurse's memory and is used to generate possible explanations for the data. Aspinall reports that various assessment and diagnostic strategies are followed in current nursing practice and warns that potential problems can result when alternative explanations are not explored [Aspinall81]. Carpenito suggests that nurses supplement their own memory-stored information by consulting references or other members of the health-care team [Carpenito89]. This thesis proposes an automated expert assistant as one of the references to help the nurse explore alternatives in her search for explanations.

1.1 Nursing Diagnosis

A nursing diagnosis may be defined in two ways. First, as a problem identification activity performed by the professional nurse and second, as a description of the health states or disrupted interaction patterns with which the nurse can assist a client. The activity described by the first definition is used to produce the description referenced by the second definition. The focus of the project described in this thesis is to produce and evaluate an expert inference engine that will, when combined with the appropriate nursing knowledge, assist in the diagnostic activity performed by the professional nurse and produce the nursing diagnosis statement. Production of the diagnostic statement is the ultimate goal of the system and, for clarity throughout the remainder of this document, the term "nursing diagnosis" will refer to that statement.

Formally, a nursing diagnosis is a statement that describes the human response (health state or actual/potential altered interaction pattern) of an individual or group that the nurse can legally identify and for which the nurse can order the definitive interventions to maintain the health state or to reduce, eliminate, or prevent alterations [Carpenito89]. This diagnosis statement can be further categorized as either an actual diagnosis, a possible diagnosis, or a potential diagnosis. An actual diagnosis is one that the nurse has validated because of the presence of major defining characteristics, or signs and symptoms. A possible diagnosis describes a problem that the nurse suspects may be present but that requires additional data to confirm or rule out. A potential diagnosis describes an altered state that is not currently present, but may occur if certain nursing interventions are not ordered and implemented [Carpenito87].

1.2 Expert Systems

Expert systems are the most common instance of the area of computer science known as artificial intelligence (AI) [Frenzel87]. A computer is said to exhibit artificial intelligence if it is programmed to "think," that is, if it simulates, to some degree, human reasoning under the same conditions [Turing50]. Experts disagree on some of the details of what makes up an expert system but most agree that two major parts are necessary; 1) a knowledge base and 2) an inference engine [Frenzel87, Waterman86]. An additional component found in most functional expert systems is a database of known facts on which the other two components operate. In practice, known facts from the database are matched by the inference engine to theoretical knowledge found in the knowledge base to solve a problem.

1.2.1 Representing Knowledge in the Expert System

The knowledge base provides the domain knowledge necessary to arrive at an intelligent decision. It is of primary importance to the solution of whatever problem the expert system is expected to solve. Completeness of that knowledge is a key ingredient in simulating intelligent behavior but accessibility to the knowledge is also a critical factor. Accessibility can be enhanced or hindered by the way the knowledge is organized in the knowledge base.

Although many methods of organizing knowledge are available for use in an expert system, the rule based method sometimes known as production rules is, by far, the most common [Waterman86]. A rule consists of two parts that embody some bit of knowledge. The first part, the antecedent, expresses a condition or premise and the second part, the consequent, states an action or conclusion that applies when the first part is true. The antecedent is prefaced by "IF" and the consequent is prefaced by "THEN" as in IF (premise) THEN (conclusion). For example, a rule may be stated as follows:

```
IF   (a car has no fuel)
THEN (the car will not run).
```

This type of knowledge representation is one of the most flexible. Rules can express a wide range of knowledge in a form suitable for automation. They are both easily understood and compatible with the way our minds store and apply knowledge. They can, therefore, simplify the job of explaining how an expert system reached a conclusion. Modification of and addition to the knowledge base is accomplished by simply rewriting an old rule or adding a new rule. These changes can take place without affecting the rest of the rules.

A high degree of detail is usually necessary to adequately represent a knowledge domain, regardless of the representation method. Details about objects, their characteristics, and actions to take under certain conditions can become very complex. Because of the necessary complexity, and because of the benefits cited for rule based methods, some experts have concluded that they are the best way to model human domain knowledge in an expert system [Frenzel87].

1.2.2 The Inference Engine

One way to conceptualize an inference engine is by thinking of it as that part of an expert system that contains general knowledge about solving a problem. The inference engine is actually software that implements a search and pattern-matching operation and allows the computer to perform in an intelligent manner. It is sometimes referred to as a rule interpreter because it's operation is somewhat akin to a computer language interpreter. However, instead of interpreting a computer program on a line by line basis, it examines rules and facts in a particular sequence looking for matches among initial and current conditions in the knowledge base. As matches are found, the engine performs various operations germane to the problem it is trying to solve.

Some of the operations may involve adding new facts to the knowledge base that, theoretically at least, increases the computer's knowledge of the problem. These new, inferred facts are referenced to the rule or rules that generated them and a logical linkage is constructed. This linkage is known as an inference chain. Each time a new rule is examined or a new fact inferred, it is checked against the current status of the problem solution stored in the knowledge base. This process is continued until eventually a particular goal is reached or the base of knowledge is exhausted and no new fact can be inferred. In the latter case, no solution is found.

1.2.2.1 Control

An inference engine may follow one of several basic approaches to search for a solution. Forward chaining, backward chaining or a hybrid combination of the two are commonly used techniques in a rule based system.

Forward chaining, otherwise known as modus ponens reasoning [Rowe88], starts with a known fact and proceeds forward in an attempt to match the fact with a rule. Using this technique, the engine attempts to match the fact, or premise, with the left side, or the IF part of the rule. When a match is found, the right side, or THEN part of the rule is executed which may lead to other facts being generated. In a large system with many rules, this procedure can be very time consuming. It is also possible that the search may go off in unproductive directions and generate many valid but unrelated facts. Nevertheless, in a diagnostic system such as the one described in this thesis, forward chaining is a common approach since just a few facts can lead to many possible solutions.

Backward chaining begins with a hypothesis, or solution, and attempts to validate the solution by searching and developing its knowledge base until it has found enough facts to support the hypothesis. It attempts to match the hypothesis with the right, or THEN side of a rule in order to test the conditions, or premises, indicated by the left side. These conditions then become interim hypotheses used in matching other rules until enough logical linkages are generated to support the original hypothesis. Backward chaining is most effective when many facts are available to support a relatively few solutions. However, it is possible that backward chaining can become "fixed" on some particular hypothesis and attempt to explore all avenues of support even when that support does not exist.

Backtracking is used to facilitate the process of backward chaining. It sets "choice points" in the search for solutions where, if one path to a possible solution does not work, another can be chosen. This allows the process to continue to explore many alternate paths until a solution is found, if possible.

A hybrid control method combines elements of both forward and backward chaining. It begins with a known fact, as in forward chaining, and attempts to find a rule that mentions it in the left (IF) side of a rule. When a match is found, backward chaining is performed using the right (THEN) side of the rule as a hypothesis. If enough supporting facts can be found or generated, that hypothesis is validated. The process is repeated until all known facts have been referenced and all possible hypotheses have been generated. Use of this method usually speeds up the process and ensures a solution, if possible. The concurrent use of a forward and backward search can rapidly converge on an answer.

The hybrid control is a good example of the artificial intelligence approach known as "generate and test." It involves a generator that produces possible solutions and an evaluator that tests the validity of those solutions. A hypothesis is generated, through forward chaining of a known fact, and it is tested for support through backward chaining. This hybrid control method will be explored in greater depth in the system design section of this document.

1.2.3 The Database

The database of known facts, otherwise known as a "fact" base contains the current status of the problem to be solved. Initially, this fact base is seeded with known facts, or the initial conditions, when the problem is presented to the expert system. Facts are added or deleted as the result of the inference process. The new state of the fact base is then available for use in other inferences. At any point in time, the fact base contains all that is known about the problem to be solved. It contains, then, valuable information that can be extracted by the user for reporting on the progress or explanation of the problem solution.

In reality, the data (fact) base, knowledge base, and inference engine are all groupings of knowledge on a conceptual level. Although each group may be segregated in physically separate files on a long term storage device such as a computer disk, they are all loaded and merged in the internal memory of the computer. Once the expert system begins operation, all three of these components are simply "knowledge" and become virtually indistinguishable from each other.

CHAPTER 2

AUTOMATED NURSING DIAGNOSIS SYSTEMS

2.1 Need

Use of automated expert systems to generate nursing diagnoses is an area that has begun to receive some recent attention in the literature, yet, to date, has not been widely implemented [Summers89]. The continuous refinement and increasing complexity of nursing science is evidenced by the evolution of conceptual models for nursing care. From the focus on functional abilities [McCain65] through the theory of self-care [Orem71] and Modeling and Role-Modeling [Erickson83] to the concept of the unitary person [Newman84], each step advanced nursing science. At the same time, greater demands were imposed on the cognitive abilities of the nurse. Nursing diagnostic activity, as part of the nursing process, is perhaps one of the most demanding processes on the nurse's cognitive skill.

Nursing service represents the largest and most labor-intensive segment of hospital operations. Cost containment for that service, which includes increased productivity, has become a chief concern [Bailey88].

The potential for beneficial use of expert systems in nursing have been documented by several authors and the implications seem attractive [Schank88, Laborde84]. A properly implemented expert nursing system could provide both financial and cognitive benefits by:

1) increasing the productivity of the nursing staff on hand, assisting in cost containment,
2) providing more consistency of performance through a common, "expert" base of knowledge and application of that knowledge,
3) preserving expert nursing knowledge that could be lost when a knowledgeable nurse retires, changes jobs, or otherwise becomes unavailable,
4) expanding expertise beyond the human expert by making the same knowledge available to remote locations and accessible on a continuous basis,
5) developing the nursing staff, especially the less experienced, through interaction with the "expert," and
6) providing a better understanding of the nursing process by forcing a review of basic problem solving techniques in initially building the expert system [Schank88].

2.2 Barriers

Given the apparent need and potential benefits, and an indication that nurses usually welcome new technologies that broaden their scope of practice [Laborde84], it would seem that development and implementation of expert nursing diagnosis systems should have proliferated. However, numerous roadblocks have been encountered thus far in both expert system development and eventual acceptance by the professional nurse.

Lack of agreement about how nursing knowledge is represented and lack of knowledge about how nurses make decisions has delayed development of expert systems [Ozbolt87]. Knowing how expert nurses make decisions is further identified as a problem area when Woolery describes a "tacit dimension" as that silent or inferred knowledge that the expert knows but cannot tell [Woolery90].

Woolery also cites a general lack of "expert" clinical nursing knowledge and heuristics. Since the term "expert" is hard to define, she contends that some expert system development may be based on knowledge and procedures that are not quantified as being expert.

Compounding these issues is a lack of a formal mechanism for exchange of such information as definitive nursing diagnosis characteristics. This problem may be slowly fading as professional nursing associations become more developed and formalized.

The development of any expert system must be presented with a strong case to overcome the excessive development time and cost. One example of a large expert system for nurses (COMMES, described in the next section) required 80 person-years and $10 million for development [Ryan85]. This system also reportedly incurs a cost of "several hundreds of thousands of dollars" and "dozens of person-years" annually in knowledge base maintenance [Evans88]. Successful development and implementation, then, would appear to require a deep and long term commitment from nursing administration along with a continuity of personnel, leadership, and resources.

2.3 Expert Nursing Systems

Expert systems have been developed and implemented in a variety of areas, including medicine. Expert diagnostic systems are available, for example, for diagnosing and proposing treatments for specific diseases and assisting the physician in determining the proper drug dosage for clients [Schank88, Laborde84]. One of the more well known of the medical diagnostic systems is called MYCIN. It is designed to provide advice in the diagnosis and treatment of infections. Through an interactive interview process, the system learns about the patient and uses a knowledge base to determine the identity of the infecting organism. Once the organism is identified, the system proposes an appropriate treatment regimen. Currently, however, there are only a few expert systems documented in the available literature that are designed for the professional nurse.

2.3.1 COMMES

Ryan describes work done at the Creighton University School of Nursing on a system called COMMES (Creighton On-line Multiple Modular Expert Systems) [Ryan85]. This system contains modules, called nursing consultants, for education consultation, audio/visual aid referrals, and protocol consultation. Also included is a nursing diagnosis consultant that will offer one or more nursing diagnoses and suggest additional potential diagnoses based on available symptoms. This system is implemented centrally at Creighton University and access is obtained through local terminals and a communication network. COMMES appears to be targeted towards the nursing student or professional nurse involved in continuing education.

Cuddigan evaluated the nursing diagnosis consultant (NDC) component to 1) determine if the NDC can reach the same conclusions as a human expert, 2) determine if the NDC is accurate when used by a novice, and 3) provide a formative evaluation of the NDC [Cuddigan89]. The results indicate that the NDC, when used by nursing faculty (representing "expert" nurses), often suspected the correct diagnosis but failed to recommend it with corresponding accuracy. Nursing students (representing novices) scored less accurately when using the NDC, perhaps due to the collection of different assessment data.

The evaluation suggested that poor correlations most often occurred when dealing with human response patterns (valuing, choosing), when the diagnoses had inadequate defining characteristics, when diagnoses were supported by more subjective cues, and when trying to determine relatively new diagnoses.

The results supported the overriding importance of proper selection and validation of the expert knowledge base. Further, although not explicitly stated, the importance of a complete and valid assessment seemed apparent. Differences in accuracy between the faculty and students seemed to point, at least in part, to a variance in the level of assessment ability. The study also highlighted some inherent limitations of a computerized diagnostic system when trying to perform in areas that cannot be explored by cognitive means.

Norris makes a distinction between an artificial intelligence system and an expert system in relation to COMMES [Norris89]. She suggests that COMMES is an expert system, designed to provide general guidelines for the steps of the nursing process, rather than a true artificial intelligence system that could substitute for human judgement. Hence, the nurse is advised to use the information provided by COMMES to augment her own ideas in planning client care, rather than having it substitute for her professional judgement.

2.3.2 CANDI

Another expert system designed for nursing and currently under development is called CANDI (Computer Aided Nursing Diagnosis and Intervention) [Chang88]. CANDI is a knowledge based system for nursing assessment (phase 1) and diagnosis (phase 2) that is designed to run on the IBM-AT class of computer hardware. Originally programmed in Common Lisp, but apparently rewritten in Borland Turbo Prolog [Hirsch89], this system is targeted toward assessment and diagnosis in the diagnostic area of self-care deficit. As is evident for most of the prototype nursing systems researched, the scope of this system is limited to only a small fraction of the possible nursing diagnostic categories. Phase 1 includes intelligent assessment data gathering through a series of approximately 30 screening questions. Abnormal responses to any of the screening questions initiates a more detailed, in-depth set of questions about the abnormal condition. This system is undergoing testing by graduate nursing students at UCLA through use of a portable lap-top computer taken to a client's bedside to conduct a systematic interview. The assessment reportedly takes about the same amount of time as a physical examination and interview done without the use of the computer. Students then submit their assessments and candidate diagnoses to faculty for discussion and further analysis. The data gathered during phase 1 of the project is being organized for implementation of phase 2 - automated display of candidate diagnoses. Currently, however, the linkages between assessment data and candidate diagnoses exist only in the minds and notes of the clinical nursing specialists working on the project [Roth89]. Future enhancements planned include not only the phase 2 diagnostic subsystem, but also an explanation subsystem, a learning subsystem, and an intervention subsystem.

2.3.3 Previous UNF System

Bloom, et al, describe work done at the University of North Florida on a system that will present a problem list of up to fifteen nursing diagnoses in the area of uncomplicated postpartum clients [Bloom87]. This system is designed to be individualized to a specific client based on that client's assessment. Capabilities include the ability to produce care plans associated with the proposed diagnoses and additional care plans defined by the nurse. This system was developed using COBOL on an IBM PC. Testing revealed several opportunities for improvement in the user interface and diagnostic capability. The experiential lessons learned through the development and testing stages demonstrated the depth and detail of knowledge necessary for proper diagnostic activity such as a need for weighting of the client cues based on their importance to the diagnoses. Further, the need for clear dissemination of that knowledge from the nursing users to the system developers was evidenced through the inadvertent omission of key client cues in diagnosis definitions. The lessons learned from this system, in fact, provided the intellectual seed for the system documented in this thesis.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 Goals

The primary goal of this project is to develop an automated expert inference engine that will reference nursing assessment data and a knowledge base of nursing etiologies to produce valid nursing diagnoses. The diagnoses produced should pass the Turing Test of intelligence by being indistinguishable from those produced by a human expert nurse. The Turing Test states that a machine may be regarded as intelligent if it provides the same results as a human would under the same circumstances [Turing50].

Several subgoals necessary in achieving the primary goal and developing a useful system follow.

A) The implementation of the system should allow definition and editing of the knowledge base in a manner that closely resembles the cognitive model of the knowledge held by the expert nurse herself. Part of this implementation should include an indication of the relative importance of the observed signs and symptoms to the diagnosis determination.

B) The inference engine should be able to explain its actions. When it proposes a diagnosis, the engine should be able to state the case for the diagnosis. This statement should include an indication of the strength of the case and the reasons why.

C) The engine should go beyond simple reporting of diagnoses and cues and perform an additional service. It should provide guidance when diagnoses are only partially indicated or when potential diagnoses are identified by alerting the nurse to other indications that may be present or that may develop.

3.2 Scope

This project is focusing on a engine that will perform only one part of the nursing process - the diagnosis. To perform a reasonable diagnosis, an assessment of the current state of the client must be performed and that assessment data made available to the diagnostic process. However, since the actual assessment is not addressed in this project, provisions must be made to ensure the availability of that data in a form that the diagnostic process can use. Therefore, the engine must be able to demonstrate some degree of "meta-knowledge." It should "know" something about the data it needs for successful diagnostic activity and be able to "reach out" to the outside environment to get that data, or generate it within itself. It must, therefore, have a sense of the assessment data necessary to confirm a particular diagnosis and be able to supplement the available data, if possible, to match a diagnosis definition. That is, it should attempt to intelligently build a higher level of assessment data when the available data is not sufficient to match the definition.

As an interim step toward validation of the inference engine a knowledge base of nursing etiologies must be defined and implemented. The definition phase involves selection of a set of testable diagnoses and an indication of how these diagnoses may be determined. This can be accomplished by interviewing an expert nurse knowledgeable in the area of nursing diagnostic techniques and subsequently refining the knowledge represented.

A member of the University of North Florida Department of Nursing (acting as an expert nurse) determined fourteen testable nursing diagnoses. This set is a subset of the North American Nursing Diagnosis Association diagnoses and consists of nursing diagnoses related to uncomplicated postpartum client care. Defining characteristics and risk factors (cues) were determined for these diagnoses based on nursing texts, primarily [Carpenito89]. Additional cues for most diagnoses were proposed by the nursing expert and included in the definitions. All cues were assigned relative importance to the individual diagnoses through a weighting scheme that involved assigning a weight to each cue.

Additionally, several hypothetical diagnosis definitions have been developed that included more complex cue relationships. These definitions were designed specifically to exercise the engine more rigorously than would be necessary for the real life nursing diagnosis definitions.

3.3 Design

3.3.1 Knowledge Representation

A basic question that had to be answered early on was the form of knowledge representation for diagnosis data. The rule based methodology appeared to be a reasonable approach for this type of data analysis since the presence of a diagnosis could be confirmed through a series of if-then propositions. It also closely resembled the cognitive model found to be evident through interviews with the nursing expert. Numerous instances of the statement "If this set of cues is present, then that diagnosis is indicated" were encountered during the initial interviews. Further, the rule based approach provided a reasonable and relatively clean method of adding and changing diagnoses without impacting other procedures or diagnosis definitions.

As the general design progressed, the appearance of a diagnosis definition began to resemble a tree-like structure (Figure 2) where the diagnosis can be thought of as the "trunk" of the tree and can be confirmed through the existence of one or more cue "branches." A simple descriptive statement defining the reasoning behind the diagnostic process seemed to be "if enough branches are observed, then there must be a trunk to support them."

An interesting feature found to be necessary was the ability to allow multiple levels of cue definitions. In the definition for the diagnosis "Altered Comfort," for example, one defining characteristic is "Autonomic response in acute

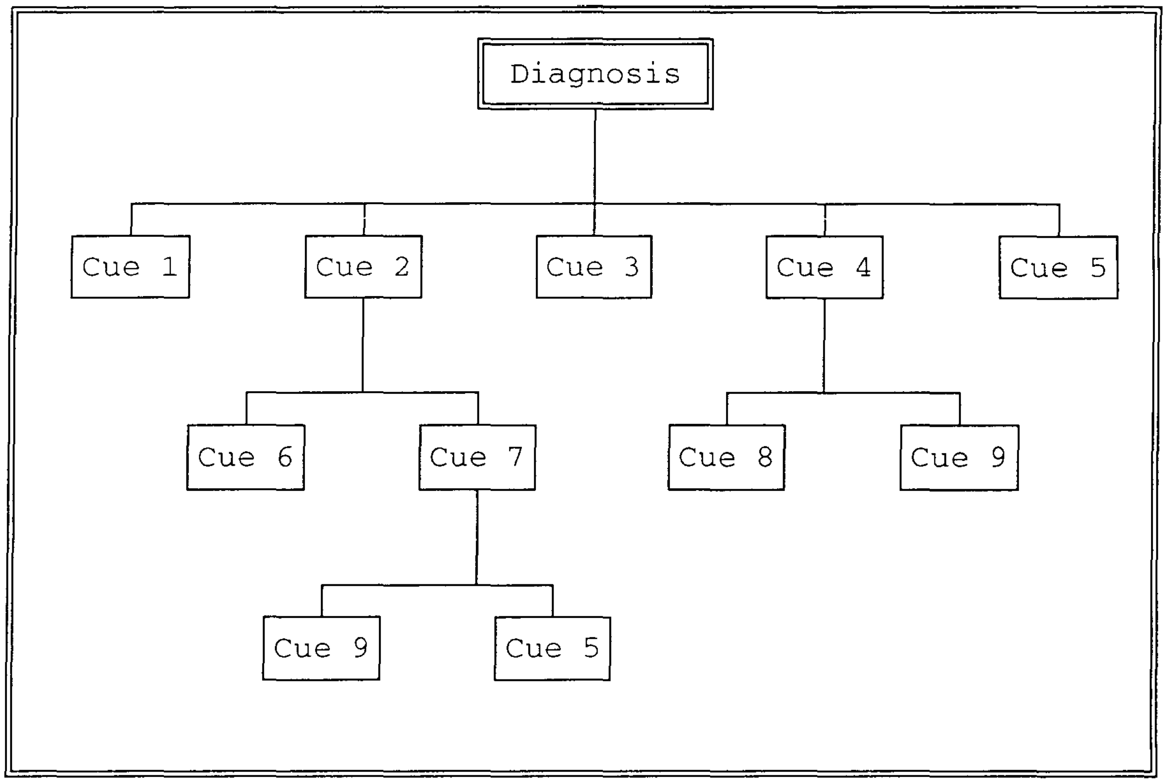


Figure 2. Diagnosis representation

pain." That cue is not available directly from a client assessment but must be determined from a combination of 1) an increase in blood pressure in acute pain, 2) an increase in respiration in acute pain, and 3) an increase in pulse rate in acute pain, all of which should be available directly from the assessment. Since these types of cues are developed within the confines of the diagnosis system rather than coming from the external environment, they are referred to as "generated cues." In Figure 2, cues that have other branches represent those cases where a cue's existence depends on the existence of other cues and must be "generated" during the diagnostic activity.

3.3.2 Cue Weighting

Cue weighting is used to selectively determine the relative importance of an individual cue to a diagnosis. Each cue is assigned a weight between 1 and 100 when used as part of a diagnosis definition. Cues of minor importance to a diagnosis are assigned small weights while more important cues are assigned larger weights. The weights of the cues exhibited by a client are summed and, if the weights reach a predetermined threshold, the diagnosis is indicated. An aggregate weight of 100 was chosen as the clustering threshold to confirm a diagnosis. Critically important cues that, alone, can confirm a diagnosis are assigned the weight of 100. These cues can confirm a diagnosis, then, without the need to reference any other cue.

Weighting of the cues in this manner not only allows the system to provide a yes/no answer to whether a diagnosis should be proposed, but serves as a form of analog scale. This scale can suggest a diagnosis as "absolutely" confirmed with a weight of 450, for instance, or perhaps, "almost" confirmed with a weight of 95.

The weighting threshold of 100 delineates the difference between "actual" diagnoses and "possible" diagnoses. Aggregate cue weights equal to or exceeding 100 indicate that the diagnosis should be confirmed ("actual" diagnosis) while an aggregate between 1 and 100 means that there is some indication for the diagnosis but not enough for confirmation ("possible" diagnosis).

Generated cues are processed in the same manner - that is, a higher level cue is generated only if the lower level cues that define it have an aggregate weight of 100 or more.

3.3.3 Potential Diagnoses

Proposing potential diagnoses presents a significantly different set of circumstances. Although similar design criteria are necessary in proposing a potential diagnosis, the defining cues are risk factors rather than defining characteristics. Cues such as "surgery," for instance, should suggest that there is a potential for "infection" even when no defining characteristics for infection are present. Therefore, a way to separate risk factor cues from defining characteristic cues is necessary. This is accomplished by assigning defining characteristics names that begin with the letter "c" and risk factors names that begin with the letter "r" when used in the diagnosis definitions.

Risk factor cue weighting is handled similarly to that of defining characteristic cue weighting. However, when referencing defining characteristics, a diagnosis may be designated as either "actual" or "possible," with the sum of the cue weights as the determining factor. A potential diagnosis, as defined in this expert system, has no "possible" designation. Therefore, the decision to propose a potential diagnosis is made only when the aggregate weight of the client cues equals or exceeds 100.

3.3.4 Control

The two basic schemes of searching and pattern-matching, forward chaining and backward chaining, were explored. When applied in their pure forms, both were found deficient in one or more areas.

Forward chaining would take a set of client cues and process them, one by one, to develop an inference chain that would result in the proposal of a diagnosis. Since a single client cue is usually not sufficient to confirm a diagnosis, most of these chains would not result in confirmation. The most likely result would be a large set of possible diagnoses, probably duplicated many times. Only a few diagnoses would be confirmed in those cases where the presence of a single cue, by itself, is confirmation of the diagnosis.

Backward chaining would sequentially select each diagnosis and try to confirm it through a search of the client cues and comparison with the diagnosis definition. This means that an attempt would be made to validate every diagnosis defined in the knowledge base whether or not there was any indication for that diagnosis. The proper diagnoses would eventually be found but the processing time could easily get out of hand, especially when many diagnoses are defined.

A direction control, or heuristic, was needed that would direct the search of the knowledge base to only those diagnoses that showed promise. Generate and test seemed a reasonable approach, involving a combination, or hybrid, of the two previous methods. The system would select a client cue and, through forward chaining, find a diagnosis that mentions that cue in its definition (the hypothesis is generated). It then would shift to a backward chaining process and attempt to validate that diagnosis with the other client cues present (the hypothesis is tested). Under this arrangement, the system would direct its search only to those diagnoses that have a chance of being validated.

Further, if a diagnosis could be confirmed with the available client cues, use of the backward chaining technique would ensure that it would be. In this way, the system would be able to rapidly converge on the correct diagnoses, confirm them or indicate them as possible, and disregard all others. This hybrid forward-backward chaining control is the one implemented in this system.

3.3.5 Computer Language

Two computer languages are used for most artificial intelligence (AI) applications - Lisp and Prolog.

Lisp (LISt Processing) was developed at MIT in the late 1950s and early 1960s and is a flexible symbol processing language. The best feature of Lisp is its ability to process lists of diverse symbols, as its name implies. Due primarily to the length of time it has been in existence, Lisp is the most widely used programming language for AI applications. Almost any data may be represented as a list or set of lists and therein lies the flexibility of the language. Data structures other than lists, such as rules, may be handled in Lisp by converting them to a list representation.

Prolog (PROgramming in LOGic), on the other hand, is designed specifically for handling if-then rules. Developed in Europe in the 1970s, it was originally designed to aid in natural language processing. It has been well received, however, by the AI community throughout the world and is the choice of the Japanese for programming their highly publicized Fifth Generation Computer project. Prolog has three key features that give it an advantage over Lisp [Rowe88]. First, Prolog can easily, and naturally, represent formal logic, the most common method of representing if-then rules. Second, it provides automatic backtracking, which facilitates the mechanism for "search," a standard procedure for AI applications. Third, Prolog supports multidirectional reasoning, which means that, depending on the circumstances, arguments to a procedure may be used as both input or output for that procedure. This makes Prolog a very flexible language, indeed. Because of these reasons, Prolog was selected as the programming language for the expert system described in this thesis.

Due to their wide proliferation and the continuing technological advances evidenced by their increased speed and storage capacity, small personal computers seemed a good target for this artificial intelligence application. Interviews with the faculty of the College of Computer and Information Science at the University of North Florida suggested that the best Prolog development package for personal computers was available through the Arity Corporation. It was decided, then, that programming for this expert system was to be accomplished using the latest version of Arity Prolog on an IBM PC/XT.

CHAPTER 4

THE SYSTEM

The final product is a generalized, menu-driven expert inference engine that will compare individual client data with a knowledge base of nursing etiologies to produce a set of actual, possible, and potential nursing diagnoses. The system is built on the premise that there is a finite number of cues that can be present in any one client and that those cues may trigger many diagnoses, based on the diagnosis definitions and client cues. The number of diagnoses presented is limited only by the capabilities of the computer hardware (memory, etc.) and the number of definitions found in the knowledge base.

Diagnoses are defined as an unordered set of weighted cues. A thresholding technique is employed such that actual, possible, and potential diagnoses are presented or not depending on the sum of the weights of the cues exhibited by the client. An arbitrary weight of 100 is used to delineate the difference between actual and possible diagnoses. Correspondingly, the value of 100 is also necessary to propose a potential diagnosis.

4.1 Operation

4.1.1 Diagnostic Process

As explained in the system design section, the inference engine uses a hybrid forward-backward chaining control. Client cues found in the client data record are used to identify candidate diagnoses and the system attempts to validate the candidates by attempting to match other client cues with the diagnosis definitions. Cues found in the client data record are translated internally into a set of Prolog facts in the form:

```
cue(cue_name).
```

where cue_name is the name of a specific client cue. Diagnoses are defined by a set of Prolog facts in the form:

```
diagnose(diagnosis_name, cue_name, cue_weight).
```

where diagnosis_name is the name of the diagnosis and cue_weight is the weight (or importance) of the cue to the specific diagnosis. Each cue used in a diagnosis definition will have a corresponding "diagnose" fact associated with it.

Diagnosis determination is accomplished by a rule that matches cue_name from the client data record facts to the cue_name found in the diagnosis definition facts. The cue_weights are summed for all matches and the appropriate diagnosis is proposed depending on the combined weights and types of cues referenced.

4.2 User Interface

Careful consideration has been given to the method of operation and the interaction with the user. An attempt has been made to make the system as intuitive as possible; that is, for it to operate as the user expects it to operate.

4.2.1 Menus

All functions available from the system are requested through menu selection. The menuing system consists of a top-level menu that is displayed across the top of the screen from which major functional categories may be selected. Once a major category is chosen, a pull-down menu associated with the category, which includes specific selections for sub-functions, is displayed. Choosing a major function or any of the sub-functions is accomplished through "pointing" or by choosing a highlighted letter found in the function description. When "pointing," a highlighted bar appears over a specific choice and may be moved with the arrow keys found on the keyboard. To choose a specific function, the "Enter" key is pressed when the bar is over the selected function. The functional categories found in the top-level menu (Figure 3) are:

* Diagnose - to select a client for analysis, and display the generated diagnoses
* Explain - to explain the rationale behind the diagnoses and generated cues
* Print - to print reports
* Redefine - to add, change, and delete a diagnosis and its generated cue definitions, and
* Quit - to exit the system

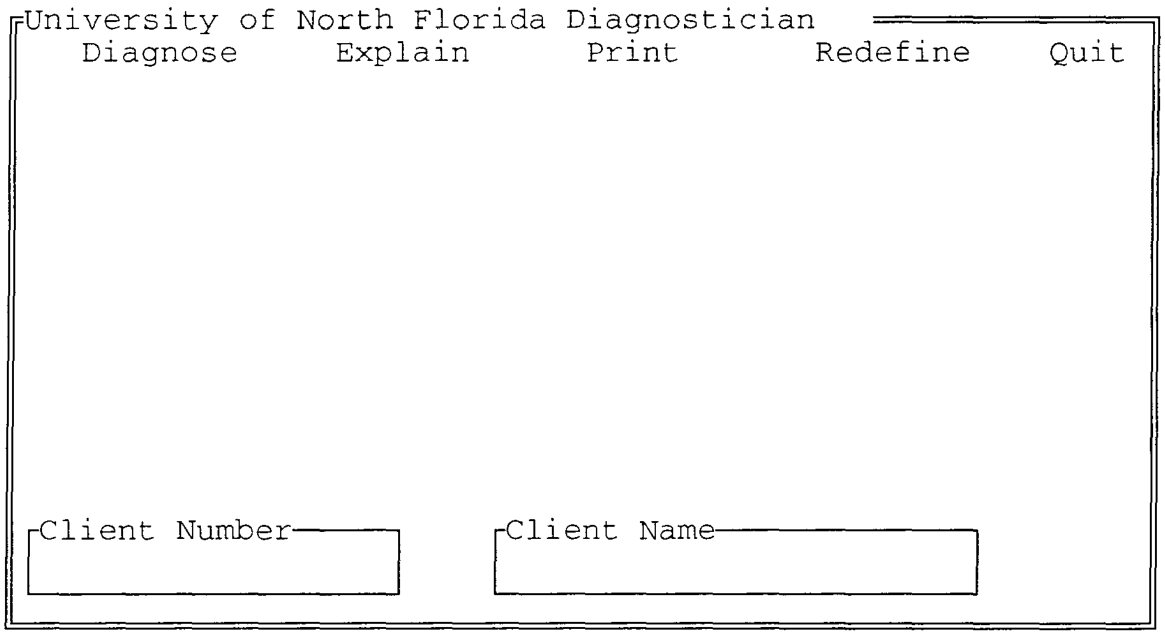


Figure 3: Top-Level Menu

4.2.2 File Maintenance

Diagnoses and/or the cues necessary to define a particular diagnosis may be dynamic and changeable over time. Additional nursing diagnoses may be proposed and accepted by the North American Nursing Diagnosis Association. As understanding of diagnostic processes increases, the individual diagnosis definitions are subject to change. To accommodate this potential for a changing environment, the system has been designed to allow additions, changes, and deletions of diagnoses and their definitions. This is done through an on-screen windowing environment that controls the necessary file maintenance.

Adding a diagnosis definition involves identifying the diagnosis by name and description and adding all defining cues with suitable weights. If a cue has been previously used in another definition, its description will be displayed after entry of the cue weight. Otherwise, the user enters a description for the new cue. Adding a previously unidentified cue will trigger a reminder message for the user to run a system utility that will check the input record to make sure the new cue is available from the assessment.

To change an existing diagnosis definition, the user selects the diagnosis to change and the diagnosis description and all current defining cues are displayed with their descriptions and weights. The user may add or delete cues and cue weights and descriptions may be changed as needed. Appropriate system actions are selected by positioning the computer's cursor over a selection box and pressing the "Enter" key. Selection boxes explicitly identify user choices with phrases such as "Add Cue," "Change Description," "Change Weight," and "Delete Cue."

Deleting a diagnosis definition requires the user to identify the diagnosis by name. A user may change her mind after the diagnosis has been named but before the definition is actually deleted without affecting the knowledge base.

4.2.3 Explanation

An explanation capability is included to illustrate the decision process taken by the inference engine and the cues used in proposing a particular diagnosis.

When the user selects "Explain" from the function selection menu, the system will request the category the user wants explained. Actual, possible, and potential diagnoses and generated cues are the categories available for explanation. Since multiple diagnoses or generated cues are possible, the system will separately list each one on the screen and identify every client cue used in determining it. As each item is fully explained, the system will pause until the user presses a key for the next item explanation.

4.3 Output

4.3.1 User Reports

User output from the system includes a set of four printed reports that describe 1) actual diagnoses, 2) possible diagnoses, 3) potential diagnoses, and 4) a general listing of cues presented by the client. Each report identifies the client by number and name at the top of the report.

The actual diagnosis report presents the confirmed diagnoses. It consists of a listing of the defining characteristics that were present in the client and used by the system for confirmation. For a diagnosis to qualify for this report, the aggregate weight of the defining characteristics equal or exceed the value of 100, the threshold for determination of an actual diagnosis.

The possible diagnosis section shows the possible diagnoses and lists the client's defining characteristics used in generating each diagnosis. Further, it identifies defining characteristics found in the diagnosis definitions but not exhibited by the client. This information may be used by the nurse to focus attention on other diagnosis specific information that may not have been gathered in the assessment.

The potential diagnosis section identifies the potential diagnoses and the risk factors found in the client that triggered the potential diagnosis. It also provides a listing of the defining characteristics necessary to confirm the diagnosis. This information alerts the nurse to the diagnoses and characteristic(s) that may occur if necessary nursing intervention steps are not taken.

The client data report consists of the client identification and all cues exhibited by that client. This report may be used for documentation of the client's health state at time of the assessment.

4.3.2 System Reports

System maintenance reports include printouts of each diagnosis definition available to the system and a listing of the client input data record.

The diagnosis definition printout identifies each diagnosis and all defining characteristics and risk factors used for the definition. Each cue is listed with their respective weights, or importance in defining the diagnosis. The source of the cue is further identified as either coming from the input record or being generated from other, lower level cues. For a generated cue, the lower level cues used for generation are identified along with their source.

The input data record report lists the format of the input cues exhibited by a client. The client data record is simply a character string that includes a positional indicator (the number "1") for each cue exhibited by the client. Every cue that has been used in a diagnosis or generated cue definition is listed with its position within the client data record identified. This report is used to document the interface with an assessment tool. If no automated assessment tool is available, it allows the client data record to be built through use of one of many available general purpose text editors.

4.4 Input

Input to the system, client data, consists of a preprocessed set of the client's characteristics and risk factors. For this project that data was prepared with a text editor but ultimately would come from an automated nursing assessment tool.

The client data format, however, is defined by the system itself and identified through the Input Data Record system report previously described. When a cue is used in a diagnosis definition, the system checks its own definition of the input record. If it finds that the referenced cue is not present through the input record, it checks its definitions for generated cues. Again, these cues may be generated from lower levels of cues that ultimately are based on cues available from the input record. If it fails to find the cue defined in either category, it will alert the user and request further definition. If the user indicates to the system that this cue is not a generated cue, the system will dynamically add a position to the input record and define it as reserved for that specific cue.

Refer to Appendix C, "University of North Florida Diagnostician - User's Manual" for a complete explanation of how to use all functions of the system.

CHAPTER 5

SYSTEM VALIDATION

5.1 Method

Two distinct types of validation are necessary when performing a validation test of an expert system. Content validation concerns testing the domain knowledge for accuracy and completeness. Process validation tests the process used in referencing the knowledge base and applying the knowledge in an appropriate manner. Content validation relates to the adequacy of the knowledge base while process validation is concerned with verifying the activities of the inference engine. For a complete validation of any expert system, both tests must be passed.

Since the primary goal of the project described in this thesis is the production of an expert inference engine, validation efforts should focus on validating the process. However, the process can only be validated in the context of a complete expert system, which would include a knowledge base. The intention, then, is validation of the expert system as a whole, and if this can be attained, validation of the process is assured.

Nevertheless, a distinction should be made between the nursing knowledge used in defining diagnoses and the process of applying that knowledge in arriving at a valid diagnosis. Inappropriate or incomplete representations in the knowledge base may result in unsatisfactory results, but that should not affect the validity of processing that knowledge. If the results obtained by a human expert using the knowledge contained in the knowledge base match those obtained by the process using that same knowledge base, then the process can be considered valid. This, after all, is the Turing Test which constitutes the test of the primary goal.

Validation of the expert system began by selecting a client population for which a finite number of diagnoses were defined. The population chosen was hospitalized, uncomplicated postpartum clients. The nursing expert identified and defined a set of fourteen nursing diagnoses that would encompass most of the problems normally associated with that population.

A total of ten representative clients were used that exhibited characteristics commonly found in the population (see Appendix A and B). The nursing expert determined relevant cues, both defining characteristics and risk factors, and proposed diagnoses that she expected for each client based on her knowledge and experience. The cues from these clients were made available to the expert system and it was allowed to generate all diagnoses that it could from the data.

Two iterations of testing were conducted. Deficiencies discovered during the first test were corrected and the second test was performed. The first test consisted of running the data from a set of four clients (clients 1 - 4, Appendix B) and analyzing the results. Problems encountered in the first test were addressed and the data from the original set of four clients were reprocessed along with data from an additional set of six new clients (clients 5 - 10, Appendix B).

A separate set of two hypothetical diagnosis definitions were developed specifically to test the process of applying knowledge found in the knowledge base. These definitions covered both broad and deep designs and involved much more complex cue relationships in order to more fully exploit the capabilities of the inference engine. Whereas the proposed nursing diagnosis definitions were fairly straightforward and simple in structure, the deep hypothetical definition contained many generated cues arranged in a complicated entanglement of interdependencies. The nursing diagnosis definitions contained a maximum of one level of generated cues where the hypothetical definitions contained as many as seven levels.

The weighting of the hypothetical cues were designed such that the presence of every cue, generated or otherwise, would be necessary to confirm the hypothetical diagnosis. The absence of even one minor cue would result in insufficient weighting to propose the diagnosis as an actual diagnosis. The diagnosis would still be designated as a possible diagnosis, however, given the presence of other cues in its definition. Figures 4 and 5 are diagrammatical representations of these two complex diagnosis definitions.

5.2 Results

The system was developed and tested with an IBM PC/XT with 640 kilobytes of internal memory and a 20 megagbyte internal disk drive, a relatively old and limited piece of hardware. Even so, a stress test of a client record that indicated every cue as present, an extremely remote possibility, required only a minute and forty seconds to generate all fourteen diagnoses. The same test, performed on an IBM PS/2 Model 80, took approximately ten seconds.

A total of 97 nursing diagnoses, 33 actual and 64 potential, were generated by the expert system for the 10 sample clients. All fourteen diagnoses were confirmed as either an actual diagnosis or a potential diagnosis for the sample population. An average of 3.3 actual and 6.4 potential diagnoses were confirmed for each client. The least number

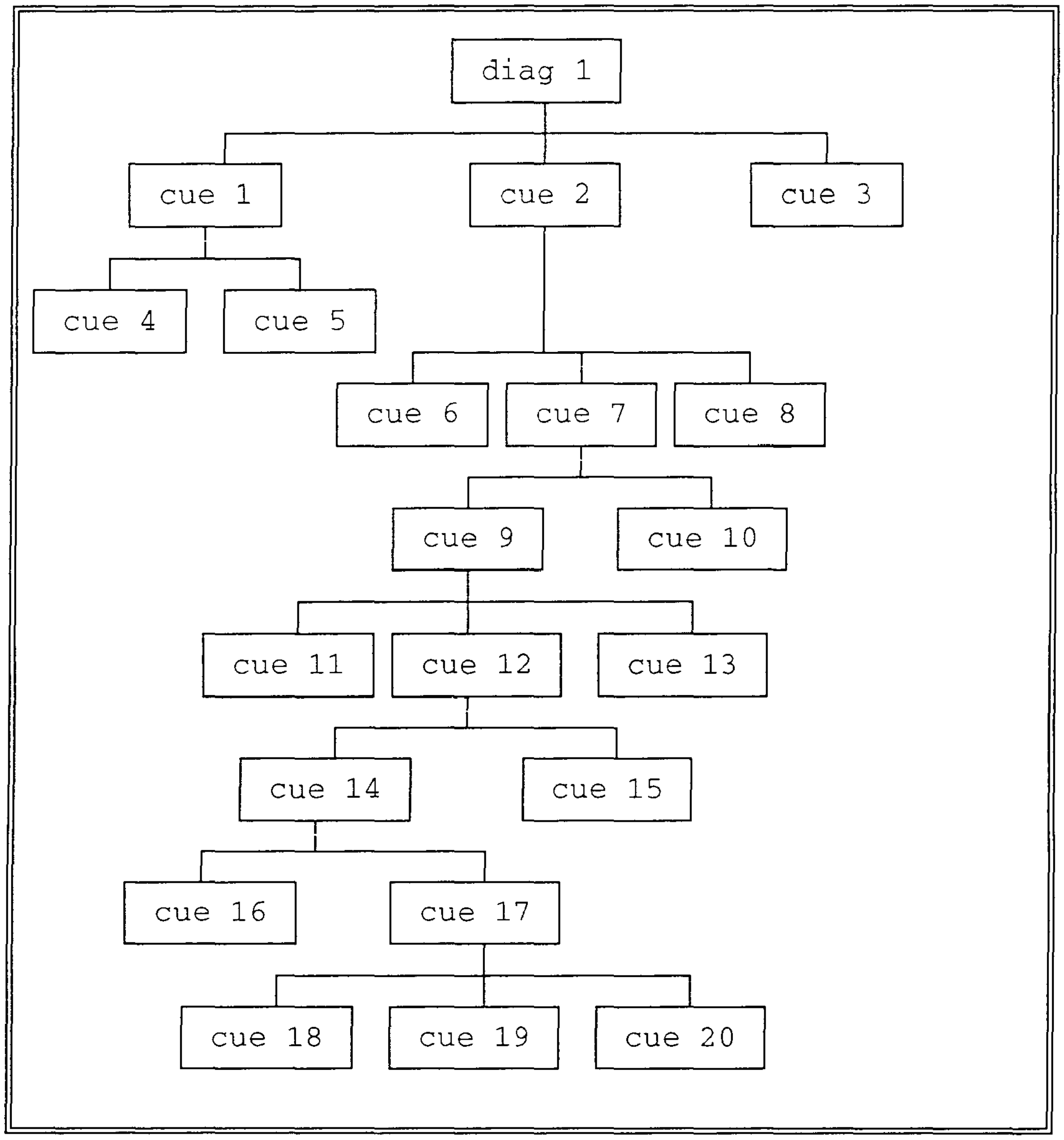


Figure 4: Deep Diagnosis Definition

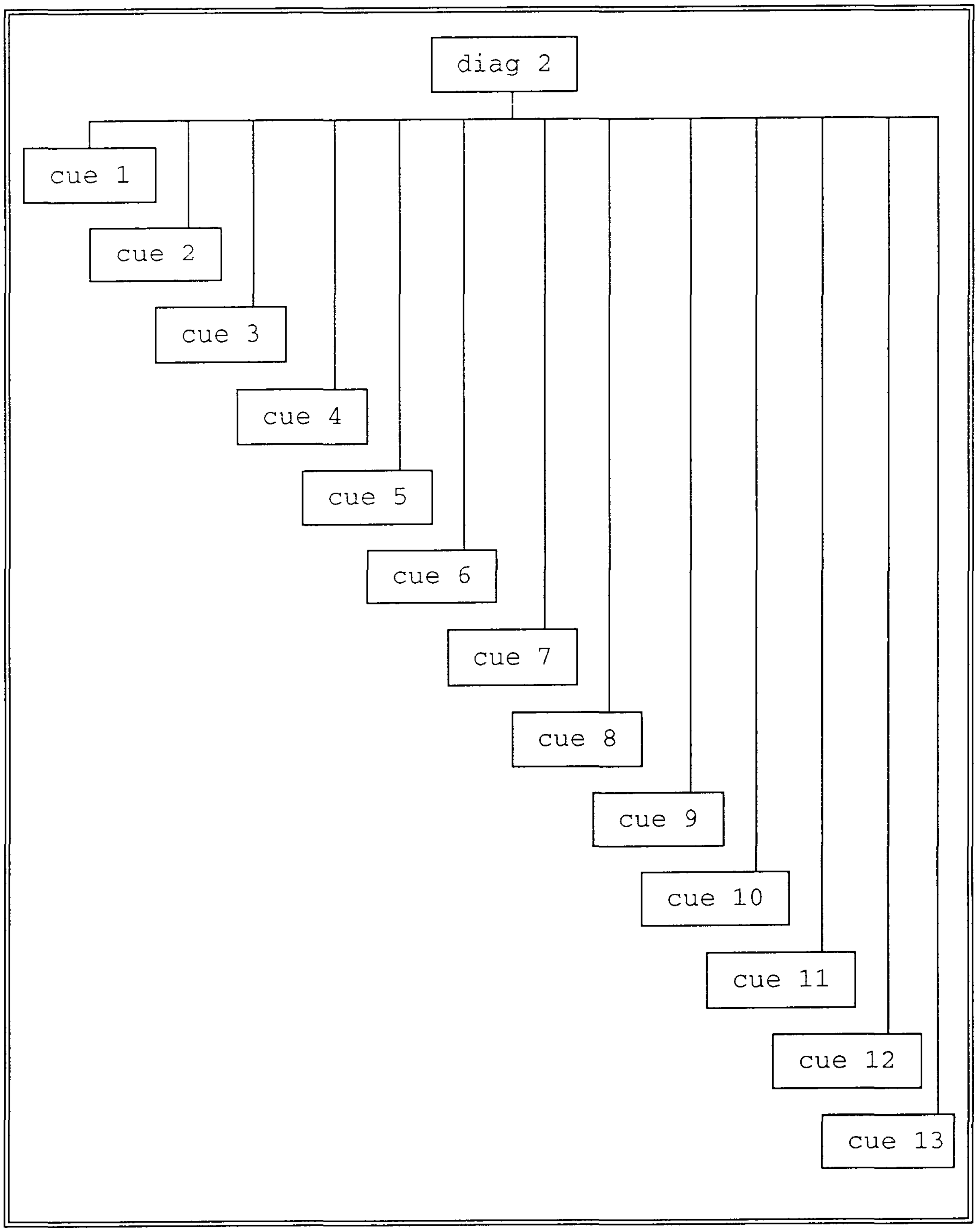


Figure 5: Broad Diagnosis Definition

of diagnoses confirmed for a single client was six - one actual and five potential diagnoses. The most for a single client was eleven, which occurred for two separate clients - three actual and eight potential diagnoses for one and four actual and seven potential diagnoses for another. No "possible" diagnoses were generated. In all test cases, enough information was available to confirm a diagnosis as either actual or potential or rule it out completely. Table 1 gives the final results for the individual test clients.

Only one diagnosis tested required a generated cue. The diagnosis "Altered Comfort" required the cue "Autonomic Response in Acute Pain" to be generated from a combination of other cues found in the client data record. This happened on only one occasion, but the generated cue was produced and listed in the diagnosis explanation. (Client 2, Appendix B).

Client	Actual Diag.	Possible Diag.	Potential Diag.	Total Diag.
1	3	0	7	10
2	3	0	8	11
3	6	0	3	9
4	3	0	7	10
5	3	0	7	10
6	3	0	7	10
7	4	0	7	11
8	1	0	5	6
9	2	0	8	10
10	5	0	5	10
Totals	33	0	64	97

Table 1: Validation Test Results

After the first try at validation, discrepancies were discovered in both the knowledge base (content) and the inference engine (process).

Content evaluation revealed that the initial definitions of several diagnoses resulted in the engine not confirming some diagnoses as expected and proposing other diagnoses that were unexpected.
Errors of omission in the expert system diagnosis definitions accounted for the majority of the discrepancies. These errors consisted of inadvertently omitting key defining characteristics or risk factors that the nursing expert reportedly used, perhaps subconsciously, in arriving at her "expected" diagnoses. The resultant list of proposed diagnoses, consequently, did not include all the diagnoses expected by the nursing expert.

One problem was encountered in the area of cue weighting where a potential diagnosis was generated because a single risk factor was defined with a weight of 100, reaching the threshold for diagnosis generation, when it actually should not have been that important. This resulted in the expert system generating a potential diagnosis that the nursing expert did not expect.

The last problem found in the knowledge base consisted of wording for a cue that was too general. This caused the nursing expert to indicate the cue as being present when it should not have been, thereby generating an unexpected potential diagnosis.

Process evaluation revealed that the expert system, on two occasions, generated a diagnosis as both possible and potential. Although enough defining characteristics and risk factors were present to justify this behavior, the nursing expert suggested that this was not valid and that the possible diagnosis should not appear if the system would also generate it as a potential diagnosis.

After adding the missing cues and "fine tuning" the system with the other changes, a second test was conducted which retested the original four clients and included an additional set of six new clients. This time the expert system matched every expected diagnosis for all ten clients except one. The one remaining discrepancy was caused by factors that fell outside the realm of the data available to the expert system. The system reported Ineffective Breast-feeding as a potential diagnosis for a client who had delivered an infant with a cleft lip (Client 3, Appendix B). The nursing expert did not expect that diagnosis because she knew through experience, or possibly common sense, that a baby with a cleft lip would not be breast-feeding.

The test of the hypothetical diagnosis definitions revealed no problems at all. The system confirmed every diagnosis expected when all cues were present. Generated cues were properly developed internally and documented through the explanation facilities.

When any one cue was removed from the assessment data, the system correctly identified the diagnosis as possible but not confirmed, as expected.

CHAPTER 6

CONCLUSIONS

The one remaining discrepancy between the results obtained from the expert system and those expected by the human expert nurse involved an area where the nurse's intuition and experience gave her an advantage over the expert system. The knowledge that a baby with a cleft lip would not be breast-feeding is intuitively obvious to an experienced nurse but escaped the notice of the system. Discounting the common sense aspect of that discrepancy, the system showed a high level of correlation to the human expert, given the knowledge base available to it. This would appear to pass the Turing Test and achieve the primary goal.

The more subjective subgoals outlined for this system were met in the following manner:

A) Diagnosis definitions were defined as tree-like structures where individual cues were weighted to correspond to their respective contributions toward designating a diagnosis. This closely resembles the model held by the expert nurse when determining a diagnosis.

B) Through the explanation facilities, the expert system is able to present a case for a particular diagnosis by listing the client cues used in determining the diagnosis. It also indicates the strength of the case by showing the sum of the weights of the cues used.

C) The expert system provides guidance for confirming possible or potential diagnoses by listing other defining characteristics found in the diagnosis definition but not found in the client's assessment.

The validation test did, however, suggest some important points to be considered in building and using any expert system for nursing. Confirming some previously cited barriers to development and implementation of expert systems, building a nursing knowledge base and providing assessment data requires extreme care.

Differences between expected and actual results of the initial validation test underscored the need for a clear understanding between the nursing expert and the knowledge engineer responsible for building the system. Even then, the "tacit dimension" of expert knowledge - that hidden knowledge that the expert uses but cannot tell - appears to be a significant obstacle that requires rigorous testing to overcome.

As with any computer system, its results are only as good as the input given it. Assessment of client cues is a major function that directly affects the outcome of any nursing diagnostic activity, whether it is done by an expert computer system or by an expert nurse.

6.1 Future directions

Through the addition of a more detailed set of cues for each diagnosis, and the addition of more diagnostic categories, the knowledge base could be enhanced to the point where it could serve as a real aid to the nurse. The addition of the rest of the diagnoses defined by the North American Nursing Diagnosis Association would significantly advance the capabilities of this system. The results of the validation test seem to indicate that the engine should perform well under the more punctilious conditions. Prospects are promising, then, for substantial benefit to nurses practicing in a clinical environment.

Another possible use could be as a computer-aided-instruction (CAI) tool in an academic environment. Knowledgeable faculty in nursing schools should be able to devise a plan that could incorporate a system such as this into a meaningful testing regimen or as a tool for reinforcement of student diagnostic skills.

Valid questions may be raised about the limits of the system. How many diagnoses can it handle? How many generated cues can it produce? How will execution speed be affected if more information is added? The answers to these questions are dependent on the capacities of the computer hardware used and, therefore, hard to answer.

There are no size limitations imposed by the software in defining diagnoses and generated cues. Theoretically, any number of diagnoses and generated cues may be defined and referenced. According to the Arity/Prolog reference manuals, a knowledge base that is too large to fit in internal memory may be segmented into "pages" and processed a page at a time with the overflow data stored on a disk drive. This implies that the size of the knowledge base is limited only by the amount of disk space available to the computer.

Processing speed, however, will be affected by the addition of more information. As more diagnoses and generated cues are defined, the forward chaining search procedure must reference correspondingly more information to find a candidate diagnosis or generated cue to test. Once a candidate is found, however, the backward chaining validation search through the client cues should show little change in speed. The implication is that there should be a graceful degradation of processing speed as more definitions are added - at least until enough definitions are defined to warrant "paging" part of the knowledge base onto a disk device. Then the additional requirement of paging information into and out of internal memory could cause a dramatic decrease in speed.

One limitation that will have to be overcome in a fully populated system is the number of cues that can be exhibited by one client. The maximum record size available in Arity Prolog is 255 bytes, allowing space for 224 distinct cues (after subtracting space for the client name and number). Should more cues be necessary, provisions must be made to allow multiple records for a single client. Currently, 122 cues of the 224 available per client (54%) are used in defining the fourteen test diagnoses.

The logical next step to make this a practical and useful tool is the addition of a robust assessment tool to provide client cues. Whether or not this tool utilizes artificial intelligence techniques, a mechanism is necessary to direct its focus. The broad range of possibilities in an unrestricted environment requires the assessment to narrow its focus as it progresses. Otherwise the sheer amount of data collection necessary would be prohibitive.

The CANDI system, under development at UCLA, appears to include such an automated assessment tool. As described earlier, this system uses a set of 30 screening questions, focusing on those areas where abnormal responses have been given and probes those areas in depth. This is the type of focusing assessment that is needed as a "front end" for the system described in this thesis.

In summary, the techniques and products demonstrated during the course of this project seem useful to the nursing profession. The expert engine has demonstrated its ability to diagnose clients from their cues and, under the proper circumstances, could and should prove itself worthwhile.

REFERENCES

[Alfaro86]
Alfaro, R., Application of Nursing Process: A Step-by-Step Guide, J. P. Lippincott Company, Philadelphia, PA, 1986.

[Aspinall81]
Aspinall, M. and C. Tanner, Decision-making for Patient Care, J. P. Lippincott Company, Philadelphia, Pa, 1981.

[Bailey88]
Bailey, D., "Computer Applications in Nursing", Computers in Nursing, 6, 5, (September/October, 1988).

[Bloom87]
Bloom, K., J. Leitner, and J. Solano, "Development of an Expert System Prototype to Generate Nursing Care Plans Based on Nursing Diagnosis", Computers in Nursing, 5, 4, (July/August, 1987).

[Carpenito87]
Carpenito, L., Handbook of Nursing Diagnosis, J. P. Lippincott Company, Philadelphia, PA, 1987.

[Carpenito89]
Carpenito, L, Nursing Diagnosis - Application to Clinical Practice, J. P. Lippincott Company, Philadelphia, PA, 1989.

[Chang88]
Chang, B. L., et al, "CANDI - A Knowledge-based System for Nursing Diagnosis", Computers in Nursing, 6, 1, (January/February, 1988).

[Chang89]
Chang, B. L., "Artificial Intelligence for Nursing Diagnosis Research and Education: Is it a Help or a Hype?", Classification of Nursing Diagnoses - Proceedings of the Eighth Conference, <Carroll-Johnson, R. M.>, ed., J. P. Lippincott, Philadelphia, PA, 1989.

[Chase88]
Chase, S. K., "Knowledge Representation in Expert Systems - Nursing Diagnosis Applications", Computers in Nursing, 6, 2, (March/April, 1988).

[Cuddigan89]
Cuddigan, J., et al, "Clinical Evaluation of an Artificial Intelligence-Based Nursing Diagnosis Consultant", Classification of Nursing Diagnoses - Proceedings of the Eighth Conference, <Carroll-Johnson, R. M.>, ed., J. P. Lippincott, Philadelphia, PA, 1989.

[Erickson83]
Erickson, H., E. Tomlin, M. Swain, Modeling and Role Modeling - A Theory and Paradigm For Nursing, Prentice Hall, Inc., Englewood Cliffs, NJ, 1983.

[Evans88]
Evans, S., "Expert Systems in Nursing Care: Issues and Expectations", Proceedings of the Twelfth Annual Symposium on Computer Applications in Medical Care, (November, 1988).

[Frenzel87]
Frenzel, L., Crash Course in Artificial Intelligence and Expert Systems, Howard W. Sams & Company, Indianapolis, IN, 1987.

[Gordon80]
Gordon, M., "Predictive Strategies in Diagnostic Tasks", Nursing Research, 29, 1, (January/February, 1980).

[Gordon87]
Gordon, M., Nursing Diagnosis, McGraw-Hill, Inc., New York, 1987.

[Hirsch89]
Hirsch, M., B. Chang and S. Gilbert, "A Computer Program to Support Patient Assessment and Clinical Decision-making in Nursing Education", Computers in Nursing, 7, 4, (July/August, 1989).

[Jones88]
Jones, J., "Clinical Reasoning in Nursing", Journal of Advanced Nursing, 13, (1988).

[Laborde84]
Laborde, J. M., "Expert Systems for Nursing?", Computers in Nursing, 2, 4, (July/August, 1984).

[Levine86]
Levine, R. I., et al, A Comprehensive Guide to AI and Expert Systems, Mcgraw-Hill Book Company, New York, 1986.

REFERENCES (continued)

[McCain65]
McCain, R., "Nursing by assessment: Not intuition", American Journal of Nursing, 65, 4, (1965), pp.82-84.

[McLane87]
Mclane, A., Classification of Nursing Diagnoses, The C. V. Mosby Company, St. Louis, 1987.

[Newman84]
Newman, M., "Nursing Diagnosis: Looking at the Whole", American Journal of Nursing, 84, (1984), pp. 1496-1499.

[Norris89]
Norris, J., "Development of Comprehensive Evaluation Research Procedures for a Computer Decision Support Consultant for Nurses", Proceedings of the Eighth Conference of the North American Nursing Diagnosis Association, (1989), pp. 264-269.

[Orem71]
Orem, D., Nursing: concepts of Practice, McGraw-Hill Book Company, New York, 1971.

[Ozbolt87]
Ozbolt, J. G., "Developing Decision Support Systems for Nursing", Computers in Nursing, 5, 3, (May/June, 1987).

[Putzier84]
Putzier, D. and K. Padrick, "Nursing Diagnosis: A Component of Nursing Process and Decision Making", Topics in Clinical Nursing, (January, 1984).

[Roth89]
Roth, K., J. DiStefano and B. Chang, "Candi - Development of the Automated Nursing Assessment Tool", Computers in Nursing, 7, 5, (September/October, 1989).

[Rowe88]
Rowe, N., Artificial Intelligence Through Prolog, Prentice Hall, Inc., Englewood Cliffs, NJ, 1988.

[Ryan85]
Ryan, S. A., "An Expert System for Nursing Practice - Clinical Decision Support", Computers in Nursing, 3, 2, (March/April, 1985).

[Schank88]
Schank, M., L. Doney, and J. Seizyk, "The Potential of Expert Systems in Nursing", Journal of Nursing Administration, 18, 6, (June, 1988).

REFERENCES (continued)

[Summers89]
Summers, S., et al, "Computerized Nursing Diagnosis and Documentation of Nursing Care in Inpatient Health Care Agencies", Proceedings of the Eighth Conference of the North American Nursing Diagnosis Association, 1, 1 (1989), pp. 270-274.

[Schnupp87]
Schnupp, P. and L. Bernhard, Productive Prolog Programming, Prentice Hall, Inc., Englewood Cliffs, NJ, 1987.

[Tanner87]
Tanner, C., et al, "Diagnostic Reasoning Strategies of Nurses and Nursing Students", Nursing Research, 36, 6, (November/December, 1987).

[Turing50]
Turing, A., "Computing Machinery and Intelligence", Mind, 59, 236, (October, 1950).

[Waterman86]
Waterman, D., A Guide to Expert Systems, Addison-Wesley Publishing Company, Reading, MA, 1986.

[Woolery90]
Woolery, L., "Expert Nurses and Expert Systems", Computers in Nursing, 8, 1, (January/February 1990).

Appendix A

DIAGNOSIS DEFINITIONS

Nursing diagnoses used in the knowledge base for testing the prototype.

d1	Colonic Constipation
d2	Ineffective Breast-Feeding
d3	Altered Comfort
d4	Altered Family Processes
d5	Altered Health Maintenance
d6	Infection
d7	Altered Nutrition: Less than body requirements
d8	Altered Nutrition: More than body requirements
d9	Altered Parenting
d10	Body Image Disturbance
d11	Altered Sexuality Patterns
d12	Sleep Pattern Disturbance
d13	Impaired Skin Integrity
d14	Urge Incontinence

Hypothetical diagnoses used in the knowledge base for testing the prototype

dt1	Test Diagnosis 1
dt2	Test Diagnosis 2

The following pages list the definitions for each of the above diagnoses.

```
'               University of North Florida                   '
'                       Diagnostician                         '

'DIAGNOSIS DEFINITION                                         '

'Diagnosis  'd1' Colonic Constipation'' is defined by the
following cues:'

c1' Decreased BM frequency'
'whose weight is : '100' and is defined by: 'input

c2' Hard, dry stool'
'whose weight is : '100' and is defined by: 'input

c3' Straining at stool'
'whose weight is : '75' and is defined by: 'input

c4' Painful defecation'
'whose weight is : '75' and is defined by: 'input

c5' Abdominal distention'
'whose weight is : '75' and is defined by: 'input

c6' Rectal pressure'
'whose weight is : '50' and is defined by: 'input

c7' Headache, appetite impairment'
'whose weight is : '50' and is defined by: 'input

c8' Abdominal pain'
'whose weight is : '50' and is defined by: 'input

r1' Pregnancy'
'whose weight is : '100' and is defined by: 'input

r2' Lack of exercise'
'whose weight is : '25' and is defined by: 'input

r4' Lack of privacy'
'whose weight is : '25' and is defined by: 'input

r5' Fear of rectal pain'
'whose weight is : '100' and is defined by: 'input

r30' Postpartum'
'whose weight is : '100' and is defined by: 'input
```

```
'               University of North Florida                    '
'                       Diagnostician                          '

'DIAGNOSIS DEFINITION                                          '

'Diagnosis  'd2' Ineffective Breast-Feeding'' is defined by
the following cues:'

c9' Actual or perceived inadequate milk supply'
'whose weight is : '25' and is defined by: 'input

c10' Infant inability to attach on to maternal breast
correctly'
'whose weight is : '75' and is defined by: 'input

c11' No observable signs of oxytocin release'
'whose weight is : '25' and is defined by: 'input

c12' Observable signs of inadequate infant intake'
'whose weight is : '75' and is defined by: 'input

c13' Nonsustained suckling at the breast'
'whose weight is : '50' and is defined by: 'input

c14' Insufficient emptying of each breast per feeding'
'whose weight is : '50' and is defined by: 'input

c15' Insufficient opportunity for suckling at the breast'
'whose weight is : '75' and is defined by: 'input

c16' Infant exhibiting fussiness and crying within the first
hour after breast-feeding; unresponsive to other comfort
measures'
'whose weight is : '50' and is defined by: 'input

c17' Infant arching and crying at the breast resisting
latching on'
'whose weight is : '75' and is defined by: 'input

r6' Breast anomaly'
'whose weight is : '100' and is defined by: 'input

r7' Infant anomaly/poor sucking reflex'
'whose weight is : '100' and is defined by: 'input

r8' Prematurity'
'whose weight is : '100' and is defined by: 'input
```

```
r9' Previous breast surgery'
'whose weight is : '25' and is defined by: 'input

r10' Maternal fatigue'
'whose weight is : '25' and is defined by: 'input

r11' Maternal anxiety'
'whose weight is : '25' and is defined by: 'input

r12' Maternal ambivalence toward breast-feeding'
'whose weight is : '100' and is defined by: 'input

r13' Inadequate Nutrition intake'
'whose weight is : '25' and is defined by: 'input

r14' Inadequate fluid intake'
'whose weight is : '25' and is defined by: 'input

r15' History of unsuccessful breast-feeding'
'whose weight is : '50' and is defined by: 'input

r16' Nonsupportive partner/family'
'whose weight is : '50' and is defined by: 'input

r17' Lack of knowledge - parenting'
'whose weight is : '75' and is defined by: 'input

r18' Ill mother'
'whose weight is : '75' and is defined by: 'input

r19' Ill infant'
'whose weight is : '75' and is defined by: 'input

r3' Breast-feeding'
'whose weight is : '50' and is defined by: 'input

r46' First-time breast-feeder'
'whose weight is : '100' and is defined by: 'input

r47' Sore nipples'
'whose weight is : '100' and is defined by: 'input

r48' Cracked nipples'
'whose weight is : '100' and is defined by: 'input
```

```
'              University of North Florida                   '
'                      Diagnostician                         '

'DIAGNOSIS DEFINITION                                        '

'Diagnosis  'd3' Altered Comfort'' is defined by the
following cues:'

c18' Client reports or demonstrates a discomfort'
'whose weight is : '100' and is defined by: 'input

c19' Autonomic response in acute pain'
'whose weight is : '75' and is defined by: '

    c22' Blood pressure increase in acute pain'
    'whose weight is : '35' and is defined by: 'input

    c23' Pulse increase in acute pain'
    'whose weight is : '35' and is defined by: 'input

    c24' Respirations increase in acute pain'
    'whose weight is : '35' and is defined by: 'input

    c25' Diaphoresis'
    'whose weight is : '25' and is defined by: 'input

    c26' Dilated pupils'
    'whose weight is : '25' and is defined by: 'input

c20' Guarded position'
'whose weight is : '50' and is defined by: 'input

c21' Crying, Moaning'
'whose weight is : '75' and is defined by: 'input

r20' Trauma (surgery, accidents)'
'whose weight is : '100' and is defined by: 'input

c64' Episiotomy'
'whose weight is : '100' and is defined by: 'input

c66' Cesarean Section'
'whose weight is : '100' and is defined by: 'input
```

```
'              University of North Florida                    '
'                    Diagnostician                            '

'DIAGNOSIS DEFINITION                                         '

'Diagnosis  'd4' Altered Family Processes'' is defined by
the following cues:'

c27' Family system does not adapt constructively to crisis'
'whose weight is : '50' and is defined by: 'input

c28' Family system does not communicate openly and
effectively between family members'
'whose weight is : '50' and is defined by: 'input

c29' Family does not meet physical needs of all its members'
'whose weight is : '50' and is defined by: 'input

c30' Family does not meet emotional needs of all its
members'
'whose weight is : '50' and is defined by: 'input

c31' Family does not meet spiritual needs of all its
members'
'whose weight is : '50' and is defined by: 'input

c32' Family does not express or accept a wide range of
feelings'
'whose weight is : '50' and is defined by: 'input

c33' Family does not seek or accept help appropriately'
'whose weight is : '50' and is defined by: 'input

r22' Birth of a child with defect'
'whose weight is : '100' and is defined by: 'input

r30' Postpartum'
'whose weight is : '100' and is defined by: 'input

c72' Lack of supportive partner/family'
'whose weight is : '100' and is defined by: 'input
```

```
                 University of North Florida                  '
'                        Diagnostician                        '

'DIAGNOSIS DEFINITION                                         '

'Diagnosis  'd5' Altered Health Maintenance'' is defined by
the following cues:'

r17' Lack of knowledge - parenting'
'whose weight is : '100' and is defined by: 'input

r23' Postpartum self-care'
'whose weight is : '50' and is defined by: 'input

r21' Primigravida'
'whose weight is : '50' and is defined by: 'input
```

```
'               University of North Florida                      '
'                       Diagnostician                            '

'DIAGNOSIS DEFINITION                                            '

'Diagnosis  'd6' Infection'' is defined by the following
cues:'

r24' Altered or insufficient leukocytes'
'whose weight is : '100' and is defined by: 'input

r25' Blood dyscrasias'
'whose weight is : '100' and is defined by: 'input

r26' Altered integumentary system'
'whose weight is : '100' and is defined by: 'input

r27' Presence of invasive lines (IVs, Foley catheter,
enteral feedings)'
'whose weight is : '100' and is defined by: 'input

r20' Trauma (surgery, accidents)'
'whose weight is : '100' and is defined by: 'input

r49' Episiotomy'
'whose weight is : '100' and is defined by: 'input

r50' Cesarean Section'
'whose weight is : '100' and is defined by: 'input
```

```
'                 University of North Florida                   '
'                        Diagnostician                          '

'DIAGNOSIS DEFINITION                                           '

'Diagnosis  'd7' Altered Nutrition: Less than body req.'' is
defined by the following cues:'

c34' Client reports or has inadequate food intake, with or
without weight loss'
'whose weight is : '100' and is defined by: 'input

c35' Actual or potential metabolic needs in excess of intake
with or without weight loss'
'whose weight is : '100' and is defined by: 'input

c36' Weight 10% - 20% below ideal for height and frame'
'whose weight is : '75' and is defined by: 'input

c37' Triceps skin fold, mid_arm circumference, and mid_arm
muscle circumference less than 60% standard measurement'
'whose weight is : '75' and is defined by: 'input

c38' Tachycardia on minimal exercise and bradycardia at
rest'
'whose weight is : '25' and is defined by: 'input

c39' Muscle weakness and tenderness'
'whose weight is : '25' and is defined by: 'input

c40' Mental irritability or confusion'
'whose weight is : '25' and is defined by: 'input

c41' Decreased serum albumin'
'whose weight is : '75' and is defined by: 'input

c42' Decreased serum transferrin or iron-binding capacity'
'whose weight is : '50' and is defined by: 'input

r28' Lack of knowledge - nutrition'
'whose weight is : '100' and is defined by: 'input

r29' Crash or fad diet'
'whose weight is : '100' and is defined by: 'input

r30' Postpartum'
'whose weight is : '50' and is defined by: 'input
```

```
'                University of North Florida                   '
'                        Diagnostician                          '

'DIAGNOSIS DEFINITION                                           '

'Diagnosis  'd8' Altered Nutrition: More than body req.'' is
defined by the following cues:'

c43' Overweight - more than 10% over ideal for height and
frame'
'whose weight is : '100' and is defined by: 'input

c44' Obese - more than 20% over ideal for height and frame'
'whose weight is : '100' and is defined by: 'input

c45' Triceps skin fold greater than 15mm (men) or 25mm
(women)'
'whose weight is : '100' and is defined by: 'input

c46' Reported undesirable eating patterns'
'whose weight is : '75' and is defined by: 'input

c47' Intake in excess of body requirements'
'whose weight is : '75' and is defined by: 'input

c48' Sedentary activity patterns'
'whose weight is : '25' and is defined by: 'input

r1' Pregnancy'
'whose weight is : '100' and is defined by: 'input
```

```
'               University of North Florida                    '
'                      Diagnostician                           '

'DIAGNOSIS DEFINITION                                          '

'Diagnosis  'd9' Altered Parenting'' is defined by the
following cues:'

c50' Inappropriate parenting behavior'
'whose weight is : '100' and is defined by: 'input

c51' Lack of parental attachment behavior'
'whose weight is : '100' and is defined by: 'input

c52' Frequent verbalization of dissatisfaction or
disappointment with infant/child'
'whose weight is : '75' and is defined by: 'input

c53' Verbalization of frustration of role'
'whose weight is : '50' and is defined by: 'input

c54' Verbalization of perceived or actual inadequacy'
'whose weight is : '50' and is defined by: 'input

c55' Diminished or inappropriate visual, tactile, or
auditory stimulation of infant'
'whose weight is : '25' and is defined by: 'input

c56' Evidence of abuse or neglect of child'
'whose weight is : '100' and is defined by: 'input

r31' Single parent'
'whose weight is : '50' and is defined by: 'input

r32' Adolescent parent'
'whose weight is : '50' and is defined by: 'input

r33' Child of unwanted pregnancy'
'whose weight is : '50' and is defined by: 'input

r34' Child of undesired sex'
'whose weight is : '50' and is defined by: 'input

r35' Child with undesired characteristics'
'whose weight is : '50' and is defined by: 'input

r36' Child with physical handicap'
'whose weight is : '75' and is defined by: 'input

r37' Child with mental handicap'
'whose weight is : '75' and is defined by: 'input

r38' Separation from nuclear family'
'whose weight is : '50' and is defined by: 'input
```

```
r39' Lack of extended family'
'whose weight is : '50' and is defined by: 'input

r17' Lack of knowledge - parenting'
'whose weight is : '50' and is defined by: 'input

r40' Unrealistic expectations of child by parent'
'whose weight is : '50' and is defined by: 'input

r41' Unrealistic expectations of self by parent'
'whose weight is : '50' and is defined by: 'input
```

```
'              University of North Florida                 '
'                     Diagnostician                        '

'DIAGNOSIS DEFINITION                                      '

'Diagnosis  'd10' Body Image Disturbance'' is defined by the
following cues:'

c57' Verbal or nonverbal negative response to actual or
perceived change in body structure and/or function'
'whose weight is : '100' and is defined by: 'input

r1' Pregnancy'
'whose weight is : '100' and is defined by: 'input

r30' Postpartum'
'whose weight is : '100' and is defined by: 'input
```

```
'              University of North Florida                  '
'                     Diagnostician                         '

'DIAGNOSIS DEFINITION                                       '

'Diagnosis  'd11' Altered Sexuality Patterns'' is defined by
the following cues:'

c58' Identification of sexual difficulties, limitations, or
changes'
'whose weight is : '100' and is defined by: 'input

r30' Postpartum'
'whose weight is : '100' and is defined by: 'input

c73' Separation from spouse'
'whose weight is : '100' and is defined by: 'input
```

```
'               University of North Florida                    '
'                       Diagnostician                          '

'DIAGNOSIS DEFINITION                                          '

'Diagnosis  'd12' Sleep Pattern Disturbance'' is defined by
the following cues:'

c59' Difficulty falling or remaining asleep'
'whose weight is : '100' and is defined by: 'input

c60' Fatigue on awakening or during the day'
'whose weight is : '50' and is defined by: 'input

c61' Dozing during the day'
'whose weight is : '50' and is defined by: 'input

c62' Agitation'
'whose weight is : '50' and is defined by: 'input

c63' Mood alterations'
'whose weight is : '50' and is defined by: 'input

r42' Hospitalization'
'whose weight is : '100' and is defined by: 'input

r30' Postpartum'
'whose weight is : '100' and is defined by: 'input
```

```
'                University of North Florida                   '
'                        Diagnostician                         '

'DIAGNOSIS DEFINITION                                          '

'Diagnosis  'd13' Impaired Skin Integrity'' is defined by
the following cues:'

c64' Episiotomy'
'whose weight is : '100' and is defined by: 'input

c65' Perineal Laceration'
'whose weight is : '100' and is defined by: 'input

c66' Cesarean Section'
'whose weight is : '100' and is defined by: 'input

c67' Denuded Skin'
'whose weight is : '50' and is defined by: 'input

c68' Erythema'
'whose weight is : '25' and is defined by: 'input

c69' Lesions'
'whose weight is : '25' and is defined by: 'input

c70' Pruritus'
'whose weight is : '25' and is defined by: 'input

c74' Cracked nipples'
'whose weight is : '100' and is defined by: 'input
```

```
'                 University of North Florida                 '
'                        Diagnostician                        '

'DIAGNOSIS DEFINITION                                         '

'Diagnosis  'd14' Urge Incontinence'' is defined by the
following cues:'

c71' Urgency followed by incontinence'
'whose weight is : '100' and is defined by: 'input

r43' Post-indwelling catheters'
'whose weight is : '50' and is defined by: 'input

r44' Loss of perineal tissue - Childbirth'
'whose weight is : '100' and is defined by: 'input

r45' Irritation to perineal area - poor personal hygiene'
'whose weight is : '50' and is defined by: 'input
```

```
'                 University of North Florida                  '
'                        Diagnostician                         '

'DIAGNOSIS DEFINITION                                          '

'Diagnosis  'dt1' test diagnosis 1'' is defined by the
following cues:'

ct1' test cue 1'
'whose weight is : '40' and is defined by: '

    ct4' test cue 4'
    'whose weight is : '50' and is defined by: 'input

    ct5' test cue 5'
    'whose weight is : '60' and is defined by: 'input

ct2' test cue 2'
'whose weight is : '40' and is defined by: '

    ct6' test cue 6'
    'whose weight is : '40' and is defined by: 'input

    ct7' test cue 7'
    'whose weight is : '40' and is defined by: '

        ct9' test cue 9'
        'whose weight is : '50' and is defined by: '

            ct11' test cue 11'
            'whose weight is : '40' and is defined by:
                                           'input

            ct12' test cue 12'
            'whose weight is : '40' and is defined by: '

                ct14' test cue 14'
                'whose weight is : '50' and is defined by: '

                    ct16' test cue 16'
                    'whose weight is : '40' and is defined
                                               by: 'input

                    ct17' test cue 17'
                    'whose weight is : '60' and is defined
                                               by: '

                        ct18' test cue 18'
                        'whose weight is : '40' and is
                                  defined by: 'input

                        ct19' test cue 19'
                        'whose weight is : '40' and is
                                  defined by: 'input
```

```
                          ct20' test cue 20'
                          'whose weight is : '40' and is
                                    defined by: 'input

                  ct15' test cue 15'
                  'whose weight is : '60' and is defined by:
                                                'input

              ct13' test cue 13'
              'whose weight is : '40' and is defined by:
                                            'input

          ct10' test cue 10'
          'whose weight is : '60' and is defined by: 'input

      ct8' test cue 8'
      'whose weight is : '40' and is defined by: 'input

  ct3' test cue 3'
  'whose weight is : '40' and is defined by: 'input
```

```
'               University of North Florida                      '
'                     Diagnostician                              '

'DIAGNOSIS DEFINITION                                            '

'Diagnosis  'dt2' test diagnosis 2'' is defined by the
following cues:'

ct4' test cue 4'
'whose weight is : '10' and is defined by: 'input

ct5' test cue 5'
'whose weight is : '10' and is defined by: 'input

ct6' test cue 6'
'whose weight is : '20' and is defined by: 'input

ct3' test cue 3'
'whose weight is : '10' and is defined by: 'input

ct8' test cue 8'
'whose weight is : '20' and is defined by: 'input

ct10' test cue 10'
'whose weight is : '10' and is defined by: 'input

ct11' test cue 11'
'whose weight is : '10' and is defined by: 'input

ct13' test cue 13'
'whose weight is : '10' and is defined by: 'input
```

APPENDIX B

Validating client scenarios, input data and output from the prototype expert system.

Client 1

Carole Evans is a 23 year old primigravida who is 14 hours post-delivery. She had a midline episiotomy and delivered a 7 pound, 11 ounce male. She is breast-feeding her infant, and states she is not sure she has enough milk to give him. The baby is observed to arch and cry at the breast. Carole holds him awkwardly. She complains of perineal pain and abdominal pain.

Expected Diagnoses (from the nursing expert)
- Ineffective Breast-feeding
- Altered Comfort
- Impaired Skin Integrity
- Potential Constipation
- Potential Altered Family Processes
- Potential Altered Health Maintenance
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Body Image Disturbance
- Potential Sleep Pattern Disturbance

Input Data (Cues from the assessment)
- Breast-feeding
- Abdominal pain
- Lack of knowledge - parenting
- Infant arching and crying at the breast resisting latching on
- Actual or perceived inadequate milk supply
- Client reports or demonstrates a discomfort
- Primigravida
- Postpartum self-care
- Postpartum
- Hospitalization
- Episiotomy
- First-time breast-feeder

Generated Diagnoses (from the prototype)
- Ineffective Breast-Feeding
- Altered Comfort
- Impaired Skin Integrity
- Potential Altered Health Maintenance
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance

Client 2

Barbara Johnston is a 16 year old, obese primigravida who is 12 hours postpartum. She is a single parent, had a cesarean section and delivered a 6 pound, 3 ounce female. She is bottle feeding. She has in IV of 1000 cc D5W running at 125 cc/hour and a Foley catheter. Her blood pressure is 140/86, pulse is 102, and respiration is 30, all elevated measurements. She is moaning and tossing in the bed.

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Altered Nutrition: More than body requirements
- Impaired Skin Integrity
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Altered Health Maintenance
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Altered Parenting
- Potential Body Image Disturbance
- Potential Sleep Pattern Disturbance

Input Data (Cues from the assessment)

- Crying, moaning
- Respirations increase in acute pain
- Pulse increase in acute pain
- Blood pressure increase in acute pain
- Primigravida
- Postpartum self-care
- Postpartum
- Obese - more than 20% over ideal for height and frame
- Adolescent parent
- Single parent
- Hospitalization
- Cesarean Section

Generated Diagnoses (from the prototype)

- Altered Comfort
- Altered Nutrition: More than body requirements
- Impaired Skin Integrity
- Potential Infection
- Potential Altered Health Maintenance
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance
- Potential Altered Parenting

Client 3

Mona Bradshaw is a 25 year old gravida 3 who is 48 hours postpartum. She delivered a 7 pound 3 ounce male over an intact perineum. She planned to breast-feed her infant. The infant has a bilateral cleft lip, and both Mona and her husband have refused to see the baby. Mona complains of rectal pressure, abdominal pain and inability to hold her urine. She states she has been unable to sleep since delivery.

Expected Diagnoses (from the nursing expert)
- Colonic Constipation
- Altered Comfort
- Altered Family Processes
- Altered Parenting
- Sleep Pattern Disturbance
- Urge Incontinence
- Potential Body Image Disturbance
- Potential Altered Sexuality Patterns

Input Data (Cues from the assessment)
- Abdominal pain
- Rectal pressure
- Infant anomaly/poor sucking reflex
- Client reports or demonstrates a discomfort
- Birth of a child with a defect
- Family does not seek or accept help appropriately
- Family does not express or accept a wide range of feelings
- Family does not communicate openly and effectively between members
- Family does not adapt constructively to crisis
- Postpartum
- Child with physical handicap
- Child with undesired characteristics
- Diminished or inappropriate visual, tactile, or auditory stimulation of infant
- Lack of parental attachment behavior
- Inappropriate parenting behavior
- Hospitalization
- Difficulty falling or remaining asleep
- Urgency followed by incontinence

Generated Diagnoses (from the prototype)
- Colonic Constipation
- Altered Comfort
- Altered Family Processes
- Altered Parenting
- Sleep Pattern Disturbance
- Urge Incontinence
- Potential Ineffective Breast-Feeding
- Potential Altered Sexuality Patterns
- Potential Body Image Disturbance

Client 4

Judy Lawrence is a 33 year old, underweight gravida 2 who is 24 hours post-delivery. She had a midline episiotomy and delivered a 5 pound, 11 ounce female. She is bottle feeding. She states that she wanted a boy, since she already has a girl at home. She is separated from her husband. She complains about being fat and states she is going to crash diet to lose her pregnancy weight.

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Impaired Skin Integrity
- Body Image Disturbance
- Potential Altered Nutrition: Less than body requirements
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Infection
- Potential Altered Parenting
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance

Input Data (Cues from the assessment)

- Nonsupportive partner/family
- Postpartum
- Crash or fad diet
- Weight 10% - 20% below ideal for height and frame
- Separation from nuclear family
- Child of undesired sex
- Single parent
- Verbal or nonverbal negative response to actual or perceived change in body structure and/or function
- Hospitalization
- Episiotomy

Generated Diagnoses (from the prototype)

- Altered Comfort
- Body Image Disturbance
- Impaired Skin Integrity
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Altered Nutrition: Less than body requirements
- Potential Altered Parenting

Client 5

Joy Davis is a 26 year old gravida 3 who delivered a 6 pound, 3 ounce female by cesarean section two days ago. She is breast-feeding her infant but says that she doesn't think she can continue since her nipples are cracked and sore. Her husband is in the Navy and is out to sea. He is not due home for three months. Joy is concerned about her ability to cope with three children alone.

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Impaired Skin Integrity
- Altered Sexuality Patterns
- Potential Ineffective Breast-Feeding
- Potential Infection
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance
- Potential Altered Parenting

Input Data (Cues from the assessment)

- Breast-feeding
- Maternal anxiety
- Client reports or demonstrates a discomfort
- Postpartum
- Separation from nuclear family
- Single parent
- Verbalization of perceived or actual inadequacy
- Hospitalization
- Cesarean section
- Sore nipples
- Cracked nipples

Generated Diagnoses (from the prototype)

- Altered Comfort
- Impaired Skin Integrity
- Altered Sexuality Patterns
- Potential Ineffective Breast-Feeding
- Potential Infection
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance
- Potential Altered Parenting

Client 6

Jane Palmer is a 14 year old primigravida who delivered a 5 pound, 6 ounce male 26 hours ago. She had a midline episiotomy and is bottle feeding her infant. Jane comments "He sure is ugly. I didn't want an ugly old boy. I wanted a pretty little girl so I could dress her up." Jane does not pick the infant up to feed him when he cries until the nursing staff tells her to. She says that her bottom is sore and she is afraid to have a bowel movement.

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Altered Parenting
- Impaired Skin Integrity
- Potential Colonic Constipation
- Potential Altered Health Maintenance
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Altered Family Processes
- Potential Body Image Disturbance

Input Data (Cues from the assessment)

- Fear of rectal pain
- Lack of knowledge - parenting
- Client reports or demonstrates a discomfort
- Primigravida
- Child with undesired characteristics
- Child of undesired sex
- Adolescent parent
- Evidence of abuse or neglect of child
- Diminished or inappropriate visual, tactile, or auditory stimulation of infant
- Frequent verbalization of dissatisfaction or disappointment with infant/child
- Lack of parental attachment behavior
- Inappropriate parenting behavior
- Hospitalization
- Episiotomy

Generated Diagnoses (from the prototype)

- Altered Comfort
- Altered Parenting
- Impaired Skin Integrity
- Potential Colonic Constipation
- Potential Altered Health Maintenance
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Altered Family Processes
- Potential Body Image Disturbance

Client 7

Jackie Bailey is a 23 year old, unmarried primigravida who delivered a 2 pound, 14 ounce male 36 hours ago by cesarean section. Her IV and Foley catheter were discontinued six hours ago. She has been eating well and has voided twice since the catheter was removed but complains of some urgency and urinary dribbling. Her son is in the NICU on a ventilator. Jackie never talks about her son, never inquires about him and has not been to see him in the nursery. She has had no visitors and states "My mom is dead and I don't know where my father is."

Expected Diagnoses (from the nursing expert)
- Altered Family Processes
- Altered Parenting
- Impaired Skin Integrity
- Urge Incontinence
- Potential Altered Comfort
- Potential Ineffective Breast-feeding
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Body Image Disturbance

Input Data (Cues from the assessment)
- Lack of supportive partner/family
- Prematurity
- Primigravida
- Postpartum
- Lack of extended family
- Separation from nuclear family
- Single parent
- Evidence of abuse or neglect of child
- Diminished or inappropriate visual, tactile, or auditory stimulation of infant
- Lack of parental attachment behavior
- Inappropriate parenting behavior
- Hospitalization
- Cesarean section
- Urgency followed by incontinence

Generated Diagnoses (from the prototype)
- Altered Family Processes
- Altered Parenting
- Impaired Skin Integrity
- Urge Incontinence
- Potential Altered Comfort
- Potential Infection
- Potential Ineffective Breast-feeding
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Body Image Disturbance

Client 8

Rhonda Jackson is a 34 year old gravida 4 who delivered a 7 pound, 8 ounce female 6 hours ago over an intact perineum. She and her husband are elated since they have three boys at home. Rhonda is beast-feeding her infant and does not anticipate problems since she breast-fed all of the boys as well. She complains of afterbirth cramps which are quite severe while the infant is nursing.

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance

Input Data (Cues from the assessment)

- Breast-feeding
- Client reports or demonstrates a discomfort
- Abdominal pain
- Postpartum
- Hospitalization

Generated Diagnoses (from the prototype)

- Altered Comfort
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance

Client 9

Judy Rivers is a 24 year old primigravida who delivered a 9 pound, 8 ounce female 24 hours ago. She is bottle feeding her infant. Judy is observed holding her infant awkwardly and shifting from side to side in the bed slowly. She states that her "stitches hurt" but she "doesn't want any medication for pain right now". She has been recently divorced from her husband of four years. She now lives with her parents. She has no job, but plans to get one when the baby is about 6 weeks old.

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Impaired Skin Integrity
- Potential Altered Health Maintenance
- Potential Altered Parenting
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance

Input Data (Cues from the assessment)

- Lack of knowledge - parenting
- Trauma (surgery, accidents)
- Client reports or demonstrates a discomfort
- Primigravida
- Postpartum
- Separation from nuclear family
- Single parent
- Hospitalization
- Irritation to perineal area / Poor personal hygiene
- Episiotomy

Generated Diagnoses (from the prototype)

- Altered Comfort
- Impaired Skin Integrity
- Potential Altered Health Maintenance
- Potential Altered Parenting
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Sleep Pattern Disturbance
- Potential Colonic Constipation
- Potential Altered Family Processes
- Potential Body Image Disturbance

Client 10

Pamela Shufnel is a 21 year old primigravida who delivered a 6 pound, 12 ounce male 36 hours ago. She had a midline episiotomy and is breast feeding her infant. She is observed crying and states "I'm so confused. My husband is insisting that we have the baby baptized by a Catholic priest. I am a Baptist - I don't believe in infant baptism." Pamela is also worried about her son's circumcision and breast-feeding, stating she doesn't know how to take care of a baby. "I just know my son is going to hate me. I don't have any milk. When am I going to have milk? He isn't getting anything to eat."

Expected Diagnoses (from the nursing expert)

- Altered Comfort
- Altered Family Processes
- Body Image Disturbance
- Sleep Pattern Disturbance
- Impaired Skin Integrity
- Potential Ineffective Breast-feeding
- Potential Altered Health Maintenance
- Potential Infection
- Potential Altered Sexuality Patterns
- Potential Colonic Constipation

Input Data (Cues from the assessment)

- Breast-feeding
- Lack of knowledge - parenting
- Nonsupportive partner/family
- Maternal anxiety
- Actual or perceived inadequate milk supply
- Crying, Moaning
- Primigravida
- Family does not express or accept a wide range of feelings
- Family does not meet spiritual needs of all its members
- Family does not meet emotional needs of all its members
- Family does not communicate openly and effectively between members
- Family does not adapt constructively to crisis
- Postpartum
- Verbal or nonverbal negative response to actual or perceived change in body structure and/or function
- Hospitalization
- Mood alterations
- Agitation
- Episiotomy
- First-time breast-feeder

Generated Diagnoses (from the prototype)

- Altered Comfort
- Altered Family Processes
- Body Image Disturbance
- Sleep Pattern Disturbance

Impaired Skin Integrity
Potential Ineffective Breast-feeding
Potential Altered Health Maintenance
Potential Infection
Potential Altered Sexuality Patterns
Potential Colonic Constipation

UNIVERSITY OF NORTH FLORIDA

DIAGNOSTICIAN

User's Manual

by Tom Edgar

TABLE OF CONTENTS

INTRODUCTION

The University of North Florida Diagnostician is a menu based expert system designed to aid in determining a set of diagnoses applicable to a specific client when presented with a list of that client's symptoms and risk factors. It will provide listings of actual, possible, and potential diagnoses as well as explanations of the rationale behind it's decisions. Diagnoses are defined within the system by providing the defining characteristics and risk factors (referred to as cues), weighted by their importance to the diagnosis. Cues may be provided as input items or they may be constructed from lower level cues. There is no limit to the number of levels available for constructing higher level cues from lower level cues.

This system (the DIAGNOSTICIAN) is designed to operate on the IBM family of personal computers or compatibles with 640 K of internal memory, a color monitor, hard disk drive and printer, and utilizing the MS-DOS or PC-DOS operating system. The original purpose of the DIAGNOSTICIAN is to provide assistance in determining nursing diagnoses. Therefore, all examples and explanations will be presented in the context of the nursing profession.

This manual will describe all procedures and techniques necessary for operation of the system.

GETTING STARTED

It is assumed that the user of the DIAGNOSTICIAN has purchased and executed a current user license with ARITY CORPORATION for operation of their ARITY/PROLOG interpreter on the target machine. The current version at this writing is version 5.1.

To begin operation of the DIAGNOSTICIAN, the user should create a subdirectory on the hard disk to contain the necessary programs and data files. Any legal directory name is acceptable, however, a meaningful name such as "DIAG" should be used.

Once the subdirectory is created, copy the files from the DIAGNOSTICIAN diskettes to the subdirectory.

To activate the DIAGNOSTICIAN, enter "API"(for Arity Prolog Interpreter) at the DOS prompt in the directory you created. The system will be loaded and the function selection menu will be presented. Depending on the type of hardware used, this initial loading will take from a few seconds to two minutes.

ORDER OF OPERATIONS

The DIAGNOSTICIAN is very flexible in its design and imposes few rules on the user. However, as in most computer systems, some rules must be followed. They are:

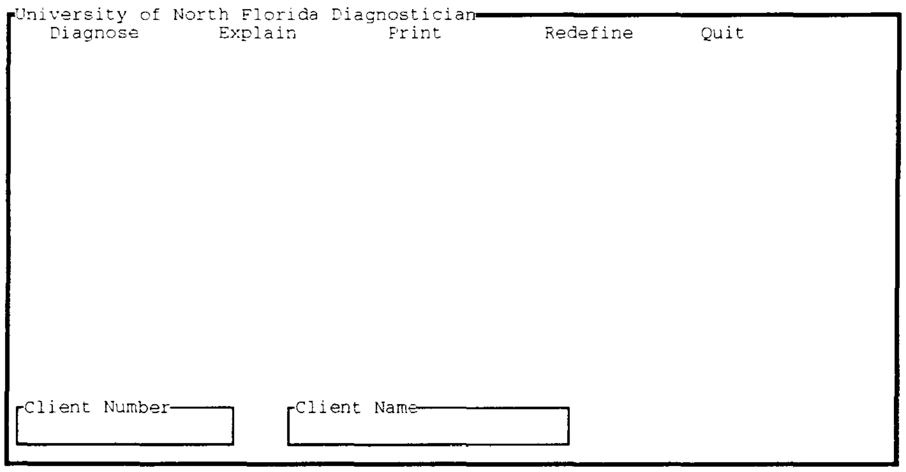
1. In order for the DIAGNOSTICIAN to develop a diagnosis from the client cues, the diagnosis must be defined within the system. Use the "Redefine" section to define all diagnoses and the relevant cues through the add, change, or delete functions provided. Save the definitions for future use by selecting the "Save redefinitions" function after defining diagnoses and cues.
2. A subject client must be selected before any meaningful diagnostic work can be done. Use the "Select Client" function under "Diagnose" to choose a client for diagnosis. This will provide the DIAGNOSTICIAN with a set of cues exhibited by that client and instruct it to perform its diagnostic activity.

After these two steps have been performed, there are no restrictions on the order of any of the other functions. Informative displays or reports may run and rerun in any order desired. However, if some basis for the diagnostic activity is changed, such as changing a diagnosis definition, the system should be "reset" by performing the

"Select Client" function again to incorporate the new conditions.

THE FUNCTION SELECTION MENU

The function selection menu allows the operator to choose the desired system function through pull-down menus. The top-level menu appears across the top of the screen listing the major functions available. Selection of a major function will activate a pull-down menu which provides more detailed sub-functions associated with the major function. The top-level menu appears as follows:



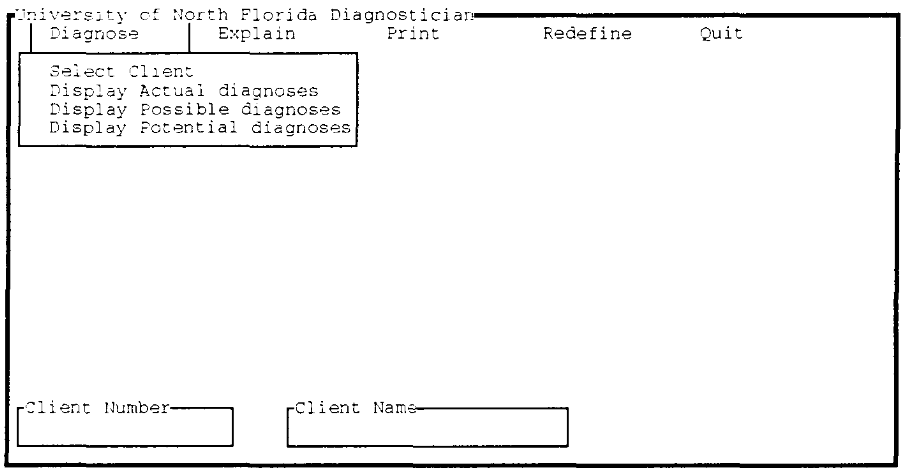
There are two methods for selecting the desired function: pointing and using accelerator keys. Both methods accomplish the same goal and the choice of which method to use is left to the user's preference. The first method, pointing, consists of using the arrow keys to move the highlight bar across the top of the screen to the desired major function and pressing the enter key or down arrow key.

To utilize the accelerator keys, the user selects the major function by pressing the highlighted letter of the function. The pull-down menu associated with the selection will appear just as it does when using pointing, as described above.

Once the pull-down menu is displayed for a particular major function, selection of a sub-function is performed in the same manner; that is, by pointing with the arrow keys or by using accelerator keys. The pull-down menus for each function selection appear in the section describing the function.

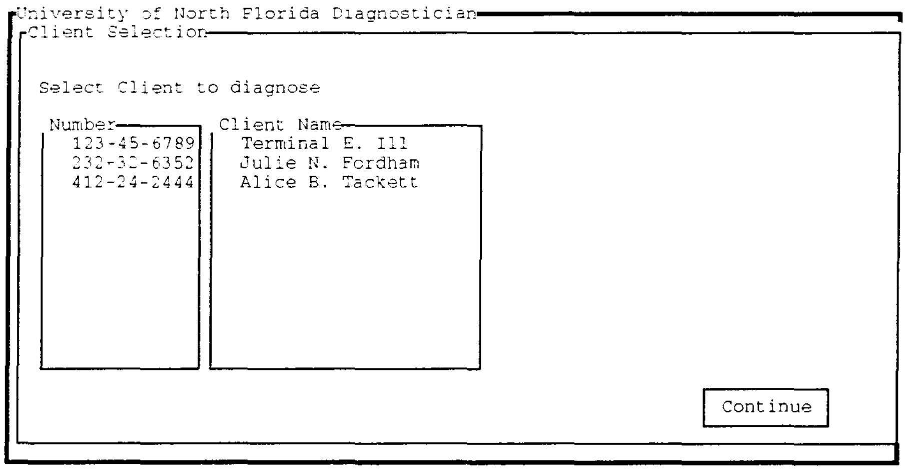
DIAGNOSE

The "Diagnose" function is the heart of the DIAGNOSTICIAN. It is used to select a client, generate diagnoses and display the results of its work. Please refer to the "Order of Operations" section of this manual for the proper sequence of selecting these options. The pull-down menu for this function appears as follows:


University of North Florida Diagnostician
Diagnose
Explain
Print
Redefine
Quit
Select Client
Display Actual diagnoses
Display Possible diagnoses
Display Potential diagnoses
Client Number
Client Name

SELECT CLIENT

This sub-function is used to choose a specific client and their corresponding set of cues to diagnose. Choosing this option will display the "Client Selection" menu which lists each client and their client number in a selection box as follows:



University of North Florida Diagnostician
Client Selection

Select Client to diagnose

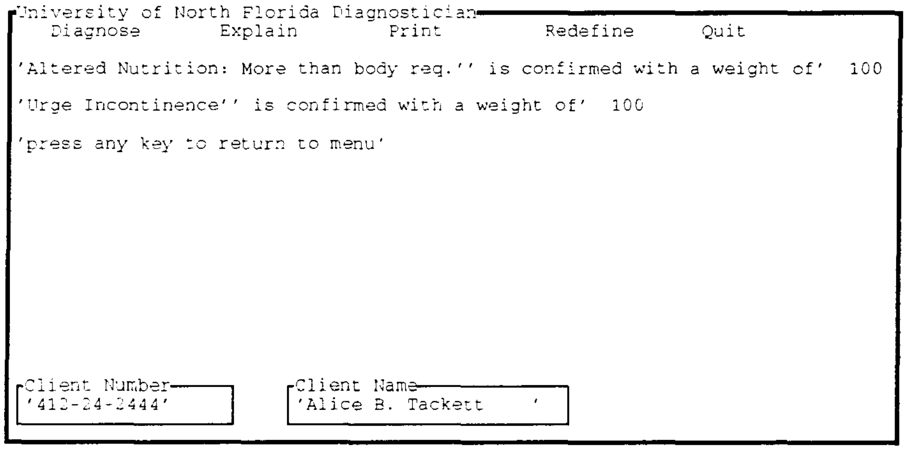
Number	Client Name
123-45-6789	Terminal E. Ill
232-32-6352	Julie N. Fordham
412-24-2444	Alice B. Tackett

Position the pointer to the desired client through use of the up and down arrows or Page Up and Page Down keys to display clients not listed in the first set of clients. When the pointer is at the correct client, press "Enter" or the Tab key to select that client. The cursor will then be positioned at the "CONTINUE" box. At this point, if the user should change her mind, she may return to the selection box by pressing the Tab key. Otherwise, exit the selection menu by pressing "Enter." There will be a short delay on exiting the "Client Selection" menu as the system references the client record and converts the cues found there into a form that the DIAGNOSTICIAN can use. It then will generate all diagnoses that it can from the set of client cues.

Once a client has been selected, the client number and name appear in the informational boxes at the bottom of the screen.

DISPLAY ACTUAL/POSSIBLE/POTENTIAL DIAGNOSES

Selection of any of these three sub-functions will display the respective diagnoses that the system has produced. The DIAGNOSTICIAN will display the diagnosis with the sum of the weights of the cues that produced it. The sum must exceed 100 for the system to generate actual and potential diagnoses. Possible diagnoses are generated if any cue is present that also appears as a cue in the diagnosis definition, regardless of weight. Refer to the "Redefine" section of this manual for further explanation of weighting and diagnosis definition procedures. If more than five diagnoses has been produced, the display will halt after the fifth has been shown and the message "more" will be displayed to indicate more diagnoses are available. Press any key to view the next diagnoses. At the end of the display, the message "press any key to return to menu" will appear. A sample display appears below.



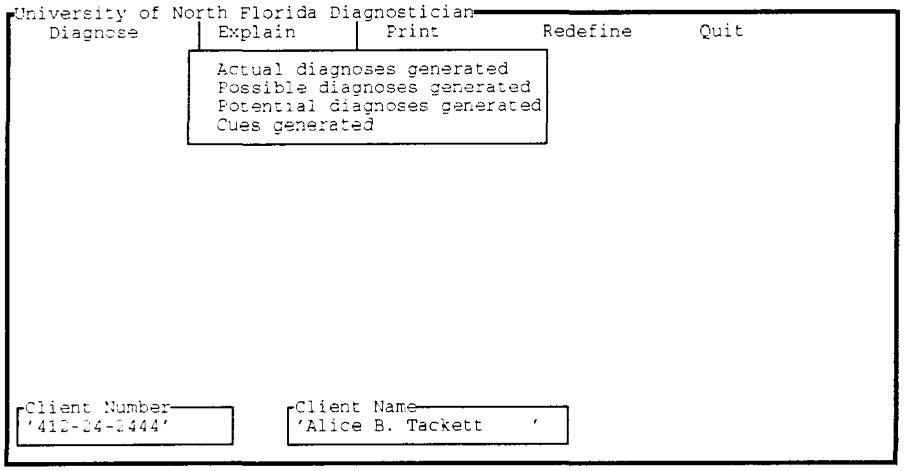
```
'Altered Nutrition: More than body req.'' is confirmed with a weight of'  100

'Urge Incontinence'' is confirmed with a weight of'  100

'press any key to return to menu'
```

Client Number
'413-24-3444'

Client Name
'Alice B. Tackett   '

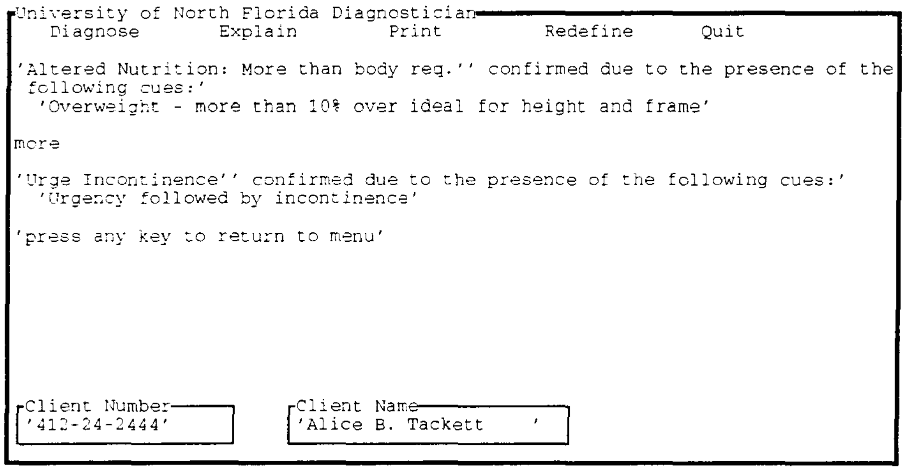
EXPLAIN

The pull-down menu for this function appears as follows:



```
University of North Florida Diagnostician
  Diagnose  | Explain   | Print      Redefine     Quit
            | Actual diagnoses generated
            | Possible diagnoses generated
            | Potential diagnoses generated
            | Cues generated

Client Number          Client Name
'412-24-2444'          'Alice B. Tackett    '
```

This function exists to provide a mechanism for the DIAGNOSTICIAN to explain its reasoning in producing diagnoses and high-level cues. It will list each generated diagnosis or high-level cue along with the client's defining cues that caused the generation. No user input is required other than selection of the type of explanation to display. If more than one diagnosis has been generated, the system will stop at the end of each diagnosis, display the "more" message, and wait for a key press to continue. A sample actual diagnosis explanation screen appears below.



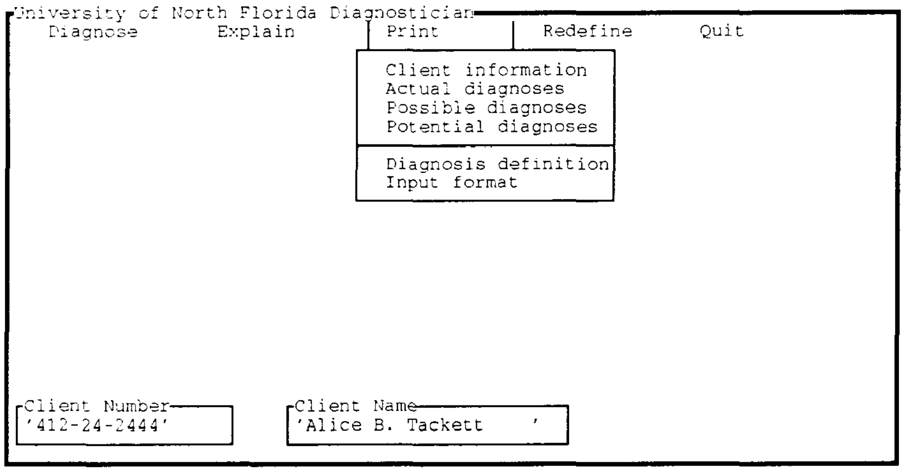
```
'Altered Nutrition: More than body req.'' confirmed due to the presence of the
 following cues:'
  'Overweight - more than 10% over ideal for height and frame'

more

'Urge Incontinence'' confirmed due to the presence of the following cues:'
  'Urgency followed by incontinence'

'press any key to return to menu'
```

Client Number: '412-24-2444'

Client Name: 'Alice B. Tackett '

PRINT

The PRINT function allows the DIAGNOSTICIAN to provide paper reports of the knowledge it possesses. The pull-down menu for this function is:



Client Number
'412-24-2444'
Client Name
'Alice B. Tackett '

Selection of any of the sub-functions will produce a report on the attached printer. Sample reports follow this section.

CLIENT INFORMATION

This report lists the client information consisting of the client's name, number, and all cues present for that client.

ACTUAL/POSSIBLE/POTENTIAL DIAGNOSES

These reports provide a listing of the diagnoses generated for the client. They explain the rationale behind its decisions by listing the client's cues used in generating each diagnoses. Further, for POSSIBLE and POTENTIAL diagnoses, the DIAGNOSTICIAN provides some guidance by listing the cues that should be observed (but are not currently present) to confirm the diagnosis. If high-level cues were generated in the process of developing a diagnosis, they are listed with the lower level cues used in their generation. There may be multiple levels of high-level cues listed.

DIAGNOSIS DEFINITION

When the user selects this report, the DIAGNOSTICIAN will request the user to enter the specific diagnosis to report as follows:

```
'Enter diagnosis to print  '
```

Enter the name of the diagnosis (d1, d12, etc.) to print and press "Enter". The system will print the diagnosis and all defining cues, weights, and descriptions for that diagnosis.

INPUT FORMAT

This is a technical report used by the system administrator. It lists the position in the input record for each of the cues used in diagnosis and high-level cue definitions. Its purpose is to define the interface between the DIAGNOSTICIAN and the assessment tool used in determining what cues are present for a particular client. Under normal, steady state, conditions, this report should not be necessary. However, should the user add or change diagnoses or high-level cue definitions, the input record for a client may change and this report will reflect the new client record definition. Not included in this report are the first two client data fields; client number - 11 characters and client name - 20 characters. Note: The client cue positions actually start with position 0, not 1, so the client record is one position longer than the sum of the cue positions, client number, and client name.

SAMPLE REPORTS

```
'                         University of North Florida                          '
'                          Diagnostician                                 '

'CLIENT INFORMATION '' for 413-24-2444  Alice B. Tackett    '

'Cues Present'

  'Breast anomaly'
  'Trauma (surgery, accidents)'
  'Family does not express or accept a wide range of feelings'
  'Overweight - more than 10% over ideal for height and frame'
  'Unrealistic expectations of self by parent'
  'Urgency followed by incontinence'
```

Sample Report - "Client Information"

```
'                          University of North Florida                          '
'                         Diagnostician                                  '

'GENERATED DIAGNOSES - ACTUAL     '' for 412-24-2444  Alice B. Tackett     '

********************************
'Altered Nutrition: More than body req.'' confirmed due to the presence of the following cues:'

  'Overweight - more than 10% over ideal for height and frame'

********************************
'Urge Incontinence'' confirmed due to the presence of the following cues:'

  'Urgency followed by incontinence'
```

Sample Report - "Generated Diagnoses - Actual"

```
'                        University of North Florida                          '
'                      Diagnostician                                   '

'GENERATED DIAGNOSES - POSSIBLE   '' for 412-24-2444  Alice B. Tackett    '


*********************************
'Altered Family Processes'' possible due to the presence of the following cues:'
  'Family does not express or accept a wide range of feelings'

'**** Other defining characteristics to observe for ''Altered Family Processes'

  'Family system does not adapt constructively to crisis'
  'Family system does not communicate openly and effectively between family members'
  'Family does not meet physical needs of all its members'
  'Family does not meet emotional needs of all its members'
  'Family does not meet spiritual needs of all its members'
  'Family does not seek or accept help appropriately'
  '**** End of other characteristics to observe'
```

Sample Report - "Generated Diagnoses - Possible"

```
'                        University of North Florida                         '
'                      Diagnostician                                  '

'GENERATED DIAGNOSES - POTENTIAL  '' for 412-24-2444  Alice B. Tackett     '


********************************
'Ineffective Breast-Feeding'' potential due to the presence of the
following cues:'
  'Breast anomaly'

'****  Defining characteristics to observe for ''Ineffective Breast-Feeding'

  'Actual or perceived inadequate milk supply'
  'Infant inability to attach on to maternal breast correctly'
  'No observable signs of oxytocin release'
  'Observable signs of inadequate infant intake'
  'Nonsustained suckling at the breast'
  'Insufficient emptying of each breast per feeding'
  'Insufficient opportunity for suckling at the breast'
  'Infant exhibiting fussiness and crying within the first hour after breast-feeding; unresponsive to
other comfort measures'
  'Infant arching and crying at the breast resisting latching on'
  '**** End of characteristics to observe'

********************************
'Altered Comfort'' potential due to the presence of the following cues:'
  'Trauma (surgery, accidents)'

'****  Defining characteristics to observe for ''Altered Comfort'

  'Client reports or demonstrates a discomfort'
  'Autonomic response in acute pain'
  'Guarded position'
  'Crying, Moaning'
  '**** End of characteristics to observe'

********************************
'Infection'' potential due to the presence of the following cues:'
  'Trauma (surgery, accidents)'

'****  Defining characteristics to observe for ''Infection'

  '**** End of characteristics to observe'
```

Sample Report - "Generated Diagnoses - Potential"

```
'                        University of North Florida                    '
'                      Diagnostician                           '

'DIAGNOSIS DEFINITION                                           '

'Diagnosis  'd2' Ineffective Breast-Feeding'' is defined by the following cues:'

c9' Actual or perceived inadequate milk supply'
'whose weight is : '25' and is defined by: 'input

c10' Infant inability to attach on to maternal breast correctly'
'whose weight is : '75' and is defined by: 'input

c11' No observable signs of oxytocin release'
'whose weight is : '25' and is defined by: 'input

c12' Observable signs of inadequate infant intake'
'whose weight is : '75' and is defined by: 'input

c13' Nonsustained suckling at the breast'
'whose weight is : '50' and is defined by: 'input

c14' Insufficient emptying of each breast per feeding'
'whose weight is : '50' and is defined by: 'input

c15' Insufficient opportunity for suckling at the breast'
'whose weight is : '75' and is defined by: 'input

c16' Infant exhibiting fussiness and crying within the first hour after breast-feeding; unresponsive
to other comfort measures'
'whose weight is : '50' and is defined by: 'input

c17' Infant arching and crying at the breast resisting latching on'
'whose weight is : '75' and is defined by: 'input

r6' Breast anomaly'
'whose weight is : '100' and is defined by: 'input

r7' Infant anomaly/poor sucking reflex'
'whose weight is : '100' and is defined by: 'input

r8' Prematurity'
'whose weight is : '100' and is defined by: 'input
```

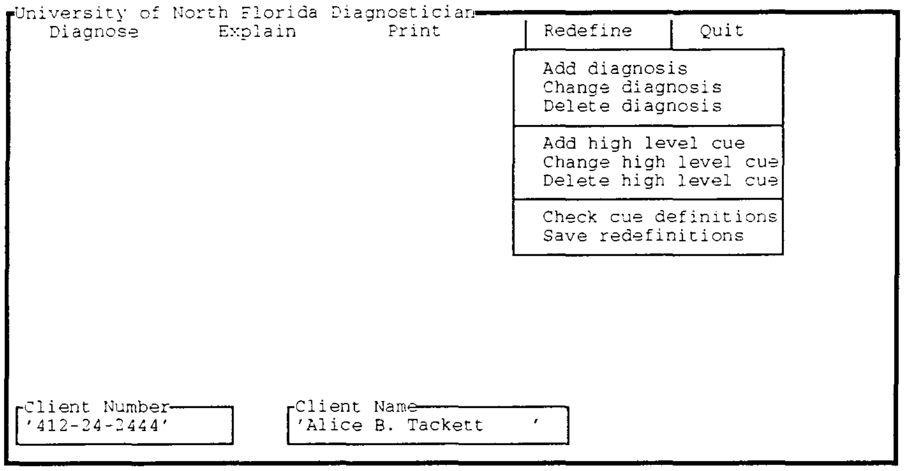
Sample Report - "Diagnosis Definition"

```
'                        University of North Florida                    '
'                      Diagnostician                           '

'INPUT RECORD FORMAT                                           '


'Position'  1  'is reserved for'  r5' Fear of rectal pain'

'Position'  2  'is reserved for'  r4' Lack of privacy'

'Position'  3  'is reserved for'  r3' Post delivery'

'Position'  4  'is reserved for'  r2' Lack of exercise'

'Position'  5  'is reserved for'  r1' Pregnancy'

'Position'  6  'is reserved for'  c8' Abdominal pain'

'Position'  7  'is reserved for'  c7' Headache, appetite impairment'

'Position'  8  'is reserved for'  c6' Rectal pressure'

'Position'  9  'is reserved for'  c5' Abdominal distention'

'Position'  10  'is reserved for'  c4' Painful defecation'

'Position'  11  'is reserved for'  c3' Straining at stool'

'Position'  12  'is reserved for'  c2' Hard, dry stool'

'Position'  13  'is reserved for'  c1' Decreased BM frequency'

'Position'  14  'is reserved for'  r19' Ill infant'

'Position'  15  'is reserved for'  r18' Ill mother'

'Position'  16  'is reserved for'  r17' Lack of knowledge - parenting'

'Position'  17  'is reserved for'  r16' Nonsupportive partner/family'

'Position'  18  'is reserved for'  r15' History of unsuccessful breast-feeding'

'Position'  19  'is reserved for'  r14' Inadequate fluid intake'
```

Sample Report - "Input Record Format"

REDEFINE

This section is used to add, change, and delete diagnosis and high-level cue definitions. The following is the display for the pull-down menu:



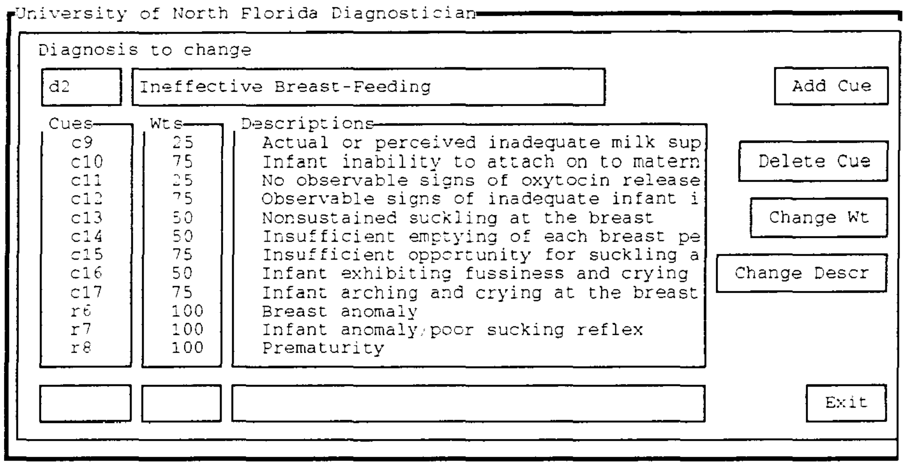
```
University of North Florida Diagnostician
  Diagnose      Explain      Print       | Redefine | Quit
                                         Add diagnosis
                                         Change diagnosis
                                         Delete diagnosis
                                         Add high level cue
                                         Change high level cue
                                         Delete high level cue
                                         Check cue definitions
                                         Save redefinitions

Client Number          Client Name
'412-24-2444'          'Alice B. Tackett     '
```

These definitions comprise the "knowledge base" from which the DIAGNOSTICIAN gains an understanding of how to determine the correct diagnosis, given a set of client cues. A diagnosis or high-level cue is defined by listing its defining cues and their relative importance by assigning weights to the cues. The weighting threshold is 100, meaning that if the sum of the weights of the cues exhibited by a client is equal to or greater than 100, that diagnosis or high-level cue is present. There are two classes of cues in defining a diagnosis; defining characteristics and risk factors. In the nursing profession, defining characteristics are the clinical criteria that validate the presence of a diagnosis. Risk factors are clinical and personal situations that can change health status or influence problem development for the client. In the DIAGNOSTICIAN, the two classes of cues are differentiated by a coding convention in the cue name. Cues that begin with a small letter "c" (i.e. c1, c2, c3, etc.) are defined as defining characteristics. Those that begin with a small "r" (i.e. r1, r2, etc.) are defined as risk factors. The DIAGNOSTICIAN uses the two types of cues in very different ways. Defining characteristics are used in developing actual diagnoses if the sum of their weights is greater than or equal to 100, or possible diagnoses if the sum of the weights is less than 100. Risk factors are used to develop potential diagnoses if the sum of their weights equals or exceeds 100.

The same algorithm holds for developing high-level cues with the exception that there is no "possible" designation. That is, either a high-level cue exists because the sum of its lower level cue weights equals or exceeds 100, or it doesn't.

Selection of the add, change, or delete functions for either diagnoses or high-level cues will initiate very similar appearing screens. User actions and responses, correspondingly, are also very similar and, for clarity, only one set of instructions will be presented here.

Select the add, change, or delete function and the redefinition screen will appear. A sample screen for the "Change Diagnosis" function appears on the following page.


University of North Florida Diagnostician

Diagnosis to change

d2 | Ineffective Breast-Feeding

Cues	Wts	Descriptions
c9	25	Actual or perceived inadequate milk sup
c10	75	Infant inability to attach on to matern
c11	25	No observable signs of oxytocin release
c12	75	Observable signs of inadequate infant i
c13	50	Nonsustained suckling at the breast
c14	50	Insufficient emptying of each breast pe
c15	75	Insufficient opportunity for suckling a
c16	50	Infant exhibiting fussiness and crying
c17	75	Infant arching and crying at the breast
r6	100	Breast anomaly
r7	100	Infant anomaly/poor sucking reflex
r8	100	Prematurity

Cursor movement around this screen is accomplished through the use of the Tab key. When the cursor is positioned at a box that is not the one the user wants, press the Tab key to move ahead one box or the Shift_Tab to back up one box. The cursor will initially be positioned in the top left box for entry of the diagnosis or high-level cue name. This is a name of up to four characters that begin with one of the following letters:

d - for diagnoses

c - for defining characteristic cues

r - for risk factor cues

Enter the name for the diagnosis or high-level cue. The cursor is then placed at the adjacent box which contains the description of the diagnosis or high-level cue. If the diagnosis or high-level cue already exists, its description will appear in this box and the user may replace the current description with another at this time. Be aware that changing this description will cause the new description to be used throughout the entire system. If the description box is blank, enter a description of up to two hundred characters.

ADD

When the user adds a diagnosis or high-level cue, the middle boxes containing the cue names, weights and cue descriptions will initially be blank. After entering the diagnosis or high-level cue description, the cursor will be positioned at the "Add Cue" box. To add a cue, press "Enter" here and the cursor will be positioned at the box at the bottom of the screen below the "Cues" column for entry of the first cue name. Enter the first cue name for this diagnosis or high-level cue and press "Enter." The cursor will move to the box at the bottom of the screen below the "Wts" column for entry of the first cue's weight. On entry of this piece of data the DIAGNOSTICIAN will add the cue name and weight in the appropriate columns and search its knowledge base for a description for that cue. If it finds the cue already defined, for example, after it has been defined as part of another diagnosis, it will display the cue's description in the "Description" column. Regardless of whether the description is found or not, the cursor is positioned at the box at the bottom of the screen under the "Description" column for entry of a description for the cue. If the

DIAGNOSTICIAN finds and displays a description, and the user accepts that description, press "Enter" without entering any characters in this field. Otherwise, a new description of up to two hundred characters may be defined for the cue by typing it now. As is the case with the diagnosis description, changing this description will make the new description available throughout the system. Upon completion of this entry, the cursor is positioned at the "Add Cue" box for the addition of other cues. Repeat this procedure until all cues are defined. When finished with the definition, move the cursor to the "Exit" box with the Tab key and press "Enter" to exit.

CHANGE

The change function will cause the DIAGNOSTICIAN to reference its knowledge base and display the defining cues of the diagnosis or high-level cue in the appropriate columns. To add cues to the definition, proceed in the same manner as described above. To delete a cue, change a cue's weight, or change a cue's description, move the cursor to the "Cues" column by using the Tab key. Then select the appropriate cue by moving the pointer with the up or down arrow keys or the Page Up or Page Down keys. When the pointer is positioned correctly, press "Enter" or the Tab key to move the cursor to the boxes on the right of the screen. Use the Tab key to move the cursor to the desired function and press "Enter." If "Delete Cue" is selected the cue is removed from the diagnosis or high-level cue definition and the cue columns are adjusted. If "Change Wt" or "Change Descr" is selected, the cursor is positioned under the desired column for entry of the new information. On completion of the new entry, the columns will reflect the changed information. When the user is finished changing the definition, move the cursor to the "Exit" box and press "Enter."

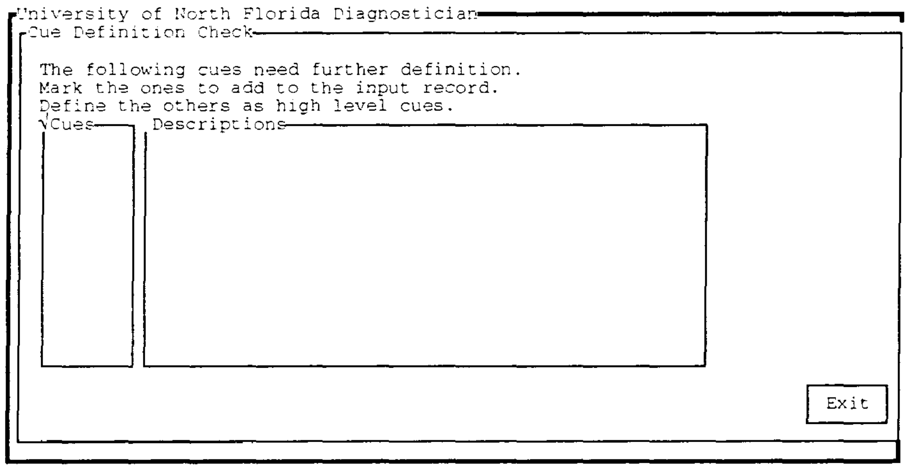
DELETE

The user may delete a previously defined definition with this function. The only user input required, other than the selection from the function selection menu, is the name associated with the diagnosis or high-level cue to delete. On entry of the diagnosis or high-level cue name, the DIAGNOSTICIAN will display the definition and the cursor will be positioned at the "Exit" box. Should the user make a mistake and desire not to delete this diagnosis or high-level cue, move the cursor back to the top left box with the TAB key and enter the correct diagnosis or high-level cue to delete. Should the user decide not to delete anything at all after having already entered a name, enter "d0" for the diagnosis or high-level cue name. This will instruct the system to disregard the delete command. To accept the deletion and exit the screen, press "Enter" at the "Exit" box.

CHECK CUE DEFINITIONS / ADD CUE TO INPUT RECORD

This function will check the knowledge base to assure that all cues used in a definition are available through either the input record or as a high-level cue. It will display the "Cue Definition Check" screen and present any cues that have been used in a definition but not defined. This screen appears on the following page. If the user has added any cues that need further definition, a message will appear at the bottom, right corner of the function selection menu. It acts as a reminder to use this function and looks like this:

```
Check Cue
Definitions
```



University of North Florida Diagnostician
Cue Definition Check

The following cues need further definition.
Mark the ones to add to the input record.
Define the others as high level cues.

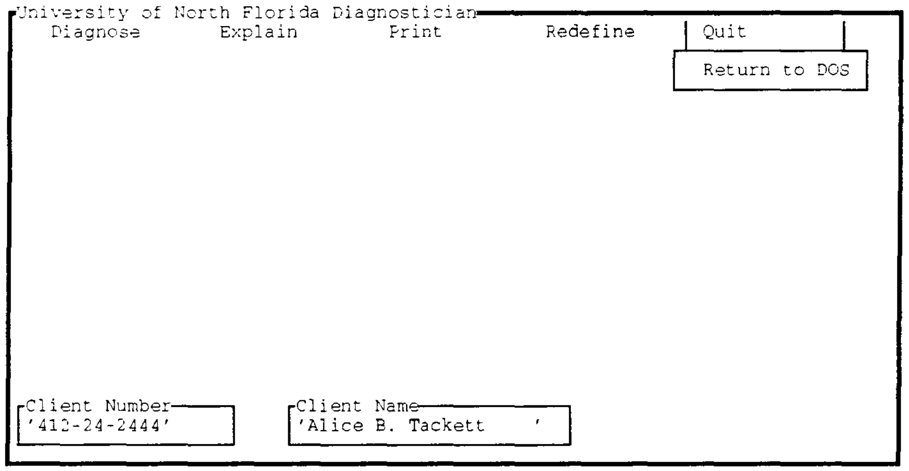
Cues that appear on this screen require further definition as to their origin. If the cue will be an input item, the cue should be marked as part of the input record. Position the pointer to the cue and press the Space Bar to mark it. A check mark will appear next to the cue to indicate that it is marked for addition to the input record. The Space Bar is used as a toggle switch and can be used to "unmark" previously marked cues. All cues not marked should be defined as high-level cues through the "Add high level cue" function. After all input cues have been marked, move the cursor to the "Exit" box with the Tab key and press "Enter". The marked cues will be automatically added to the input record. The user should print a new "Input Record" report to see the positions of the new cues in the input record. It is strongly recommended that the user select this function after adding or changing any diagnosis or high-level cue definitions.

SAVE REDEFINITIONS

After adding, changing, or deleting definitions the user should select this function to record the new definitions for future use. The new definitions are available in the current session only unless they are saved. No user input is required other than the selection of this function.

QUIT

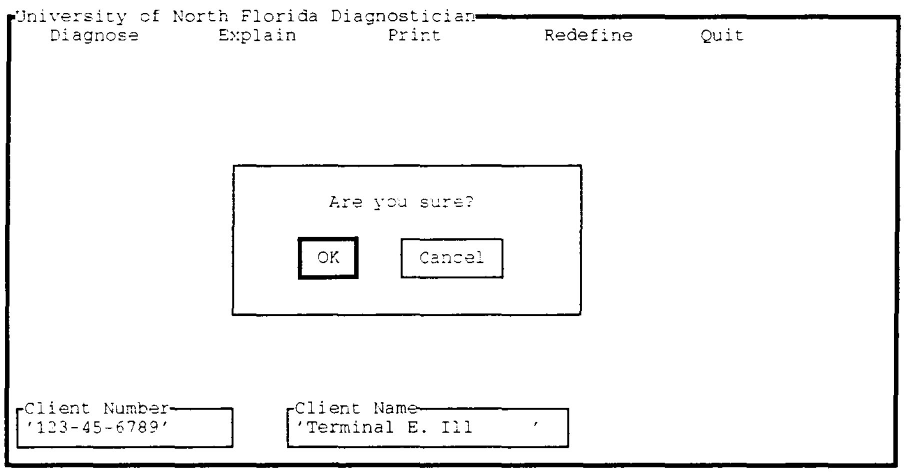
When the user wants to exit the session, the "Quit" function should be selected. The "Quit" pull-down menu is:



Client Number
'412-24-2444'

Client Name
'Alice B. Tackett '

Selection of the "Return to DOS" box will display a confirmation box as follows:



```
'123-45-6789'
'Terminal E. Ill     '
```

Press "Enter" to return to the DOS operating system.

UNIVERSITY OF NORTH FLORIDA

DIAGNOSTICIAN

Programmer's Manual

by Tom Edgar

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The University of North Florida Diagnostician (Diagnostician) is a menu-driven, expert system for generation of diagnoses. It is written in Arity Prolog, version 5.1, using a windowing environment for all screen handling and user interface. The Prolog interpreter resides atop Microsoft's DOS operating system. The system was developed under DOS version 3.2 but has been successfully tested with DOS version 4.01. In its present configuration, it requires 640 K of internal memory, a hard disk, monitor, and printer. Although designed for use with a color monitor, it will operate satisfactorily with a single color monitor (tested with a Hercules controller).

It is assumed that programmers working with this system are proficient in both the DOS operating system and Arity's version of Prolog. If not, a suggested list of reading material includes the "MS-DOS Operating System Reference Manual" for a review of DOS, and both "Using the ARITY/Prolog Interpreter and Compiler" and "The ARITY/Prolog Reference Manual" for procedures available through Prolog.

CHAPTER 2

DESIGN

The DIAGNOSTICIAN is built through three main modules of Prolog rules and facts - the client data base, the nursing knowledge base, and the inference engine.

2.1 The Client Data Base

The client data base is a set of client data records, each having specific data fields for the client number, client name, and client cues. The client cues are developed from an assessment of the client's condition and are assumed to be available to this system.

The client data record is a character string record that may be manipulated through most general purpose text editors. Both the client number and name are used only for identification. The client cues are represented by an indicator (the character "1") in specific positions defined by the DIAGNOSTICIAN to represent specific cues. The presence of the "1" indicator means that the cue defined for that position is present, whereas the absence of the "1" means that cue is not present.

The DIAGNOSTICIAN has the capability of adding positions to the client data record for additional cues when necessary. This procedure is handled by the inference engine when new cues are used in defining diagnoses or generated cues and when those cues are not already defined in the client data record.

The format for the current client data record is listed in Appendix A.

2.2 The Nursing Knowledge Base

The nursing knowledge base consists of information specifically for the nursing application. It includes diagnosis definitions, generated (high-level) cue definitions, and client data record processing information.
Diagnoses are defined through Prolog facts in two formats as follows:

```
     diagnose(diagnosis_name, cue_name, cue_weight).
                               and
     diag_descr(diagnosis_name, diagnosis_description).
```

The first format identifies each cue that is included in a diagnosis definition and the weight of the cue. A diagnosis, most probably, will be defined by multiple cues and this format will serve to record as many cues as necessary. Diagnosis_name is a label beginning with the character "d" that identifies the specific diagnosis. Diagnosis_names used in the prototype system are of the form d1, d2, d3, etc. Cue_name is a label in the format c1, c2, etc. for defining characteristic cues and r1, r2, etc. for risk factor cues. Cue_weight is a number to indicate the relative importance, or weight, of the cue to the diagnosis definition.

The second format gives the diagnosis a description of up to two hundred characters. The description is a character string that may contain anything the user feels is important to describe the diagnosis. In the prototype, this description takes a form such as "Ineffective Breast-feeding," etc.

Since generated (high-level) cues are not available directly from the client data record, they are generated during the inference process when diagnosing a client. They are defined in the same manner as a diagnosis. These cue definitions are represented by the following Prolog fact:

```
cue(generated_cue_name, cue_name, cue_weight)
```

Generated_cue_name identifies the name by which the cue is known. Again, it is of the form c1, c2, etc. for defining characteristic cues and r1, r2, etc. for risk factor cues. As in the definitions for a diagnosis, cue_name represents a cue used in defining the generated cue. Cue_weight determines the relative importance of the cue to the generated cue. Cue_name may also represent a generated cue. It is possible to include in the definition of a generated cue, other generated cues that are built from still other, lower level cues - that is, generated cue definitions may be cascaded where one generated cue may be defined by other, lower level generated cues. There is no theoretical limit to the number of levels of generated cues. The only limitations imposed will be those imposed by the availability of hardware resources.

All client cues, whether available from the client data record or generated, include one other Prolog fact definition. The cue description fact takes the format:

```
cue(cue_name, cue_description)
```

Again, as in the diagnosis definition, cue_description is a character string of up to two hundred characters.

Client data record processing information consists of Prolog rules, or predicates, that identify the position of each cue within the client record and a Prolog fact to indicate the last record position used.

The cue position predicates take the form:

```
    assert_input(cue_position) :- assertz(cue(cue_name)).
```

where cue_position is the position within cue portion of the client data record where the cue is found, and cue_name is the name of the cue found there.

These predicates are used to generate other Prolog facts during the processing of the client data record. The other Prolog facts are generated from the "assertz" part of the rule and indicate that a particular cue is exhibited by the client. It takes the form "cue(cue_name)." when the assert_input predicate is activated.

Cue_position includes an unstated offset of 32 characters to account for an eleven character client number, a twenty character client name, and one character for position 0 of the client cue portion of the record. A cue_position of 1, for instance, actually points to the 33rd position of the client data record.

The last bit of information contained in the nursing knowledge base is an indication of the last cue position used. It is formatted as follows:

```
last_input_position(position_number).
```

where position_number is the last client cue position defined for use.

As new cues are defined, the last_input_position fact is referenced to determine the end of the client data record. Positions for newly defined cues are added to the end of the record. It is expected that the client data record will reflect the new format after new cues have been added. This means that the client data record must include a "1" or some other character, such as a space, in the new position for the record to be processed correctly.

This discussion of the nursing knowledge base is included for information only. No programmer interaction is required to manipulate the data contained therein. All definitions, formats, and other information are controlled by the inference engine through the REDEFINITION function available through the top level function selection menu. The subfunction, SAVE REDEFINITIONS will save the current state of all information referenced in this discussion.

The current state of the nursing knowledge base is listed in Appendix B.

2.3 The Inference Engine

The design of the DIAGNOSTICIAN's inference engine is modular for ease of updating and for clarity of functionality. Sections are segregated by the different user selectable functions available from the top level menu. Every procedure related to a specific function is self-contained in that portion of the program designated for the function so the programmer does not have to search through the entire program for relevant modules and procedures.

Several top-level functions involve screen handling through Arity's "dialog box" routines. These are temporary window based environments where normal program control may be interrupted at specific points to perform various program functions.

The inference process is controlled through a hybrid forward-backward chaining mechanism. Cues available from the client data record are processed in a forward chaining manner to find a diagnosis that mentions it in its diagnosis definition. Control then switches to a backward chaining process where an attempt is made to verify the diagnosis definition through the client cues. If enough client cues exist for verification, the diagnosis is presented as an actual, possible, or potential diagnosis.

CHAPTER 3

PROGRAM MODULES

This chapter is intended to provide some insight into the programming of the inference engine for the DIAGNOSTICIAN. For a thorough understanding of the program, the program source listing should be referenced in concert with this document. The inference engine, for the purpose of this document, is defined as all programming not otherwise referenced, i.e., in the nursing knowledge base.

There are two source files containing the programming for the inference engine - "prolog.ini" and "diag.ari".

"prolog.ini" is an initiation routine, consulted immediately after activation of the Prolog interpreter. The code found there consults both the nursing knowledge base and diagnosis program, sets the operating environment through window definitions, and activates the top-level function selection menu.

The second source file, "diag.ari", contains the rest of the programming for the DIAGNOSTICIAN. This program will be explored in depth in the following sections.

3.1 Screen Handling

Virtually all screen handling is accomplished through a menu and the dialog boxes available from Arity Prolog.

3.1.1 Menu

The menu is defined near the beginning of the program and functions as described in the ARITY/PROLOG REFERENCE MANUAL, chapter 13. Five major functional categories are divided into respective subfunctions, each selectable through pointing or through "accelerator keys." The menu is controlled by the "do_top_menu" predicate. This predicate activates the menu and processes the user selection through a "case" statement. It also will display a message to alert the user to "check cue definitions" when a new cue has been used in either a diagnosis or generated cue definition.

3.1.2 Dialog Boxes

Dialog boxes are described in chapter 12 of the ARITY/PROLOG REFERENCE MANUAL. In the DIAGNOSTICIAN, these boxes are used to interact with the user in 1) defining or redefining diagnoses and generated cues, 2) selecting a client and 3) checking cue definitions. Each of the boxes are defined at the beginning of the program.

Every dialog box is controlled through a set of predicates related to the function of the box. Each set of predicates contains specific programming for the selected function. Processing is intercepted at the initialization of the dialog box, between fields within the box, and when action push buttons are selected to perform various operations.

Each set of predicates contains processing for handling special keys when pressed by the user. The ENTER key, UP ARROW, DOWN ARROW, PAGE UP, and PAGE DOWN are handled by predicates that control program action when that key is pressed.

3.1.2.1 Defining/Redefining Diagnoses and Generated Cues

The dialog box controlling these functions is named "change_cues." It contains edit fields, text (message) boxes, list boxes, and push buttons. This box is used by six different functions. Each function uses a separate set of predicates to control the operation of the dialog box. The functions (and predicate names) are:

1) add diagnosis - (get_msg_ad),
2) change diagnosis - (get_msg_cd),
3) delete diagnosis - (get_msg_dd),
4) add generated (high level) cue - (get_msg_ac),
5) change generated (high level) cue - (get_msg_cc),
6) delete generated (high level) cue - (get_msg_dc)

As new cues are used in definitions, a check is made to determine if the cue has been previously used. If not, a "new_cue" fact is recorded in the knowledge base to alert the user to run the "Check Cue Definitions" function.

3.1.2.2 Errata

Three unfortunate situations exist in using the dialog boxes, specifically in using the list boxes.

First, when using the "Change Diagnosis" or "Change High Level Cue" functions, the previously defined cue names, weights, and descriptions should be displayed in the respective list boxes. If the user decides to change either the weight or description of the first cue displayed, the selected list box will be offset by one item after the change has been made. The new weight or description is processed correctly, however. To correct the display, press the PAGE UP key repeatedly until the "indicator" is pointing to the beginning of each list.

Second, when numerous cues are used in a definition, multiple pages of data for the list boxes are necessary. As the user pages through the cue data with the PAGE UP or PAGE DOWN keys, the weight and description list boxes may become uncoordinated with the cue name list box. This can result in a possible misreading of the weights and descriptions in relation to the cue names. For example, the indicators for each list box may be pointing to the correct item but the items pointed to may be at the top of the cue name list box and midway down the list boxes for the weights and descriptions. This situation can be corrected, as above, by pressing the PAGE UP or PAGE DOWN key repeatedly until the indicators line up at the top or bottom of the list.

Third, an empty list box may not be updated and displayed. When defining a new diagnosis or generated cue, for instance, the list boxes for the defining cues should be empty. However, since the system will not allow this, a null or empty string must be supplied as the first entry in each box. The null strings may be replaced, however, when cue information is available for the list boxes.

3.1.2.3 Client Selection

Selecting a client for diagnosis is controlled through the dialog box named "cselect" and predicates named "get_msg_sel." Processing begins by abolishing all facts related to any previous diagnostic activity. During initialization of the dialog box the client data base file is read and client numbers and names are displayed. On selection of a client, the system will display the name and number in the appropriate windows and proceed with its diagnostic activity. This activity will be traced in the "Diagnosis" section.

3.1.2.4 Check Cue Definitions

When new cues are added to a diagnosis or generated cue definition, provisions must be made to ensure the availability of the cue data. The cue definition check routine ensures that all cues used in a definition are available to the system either through the client data record or through generation of the cue from other cues. This checking process is controlled by the "definition" dialog box and "get_msg_def" predicates.

The first thing that must be determined is whether the cue data is already available. The input record definition predicates are checked for every cue used in every definition. Any cues not found in the input record are checked for definition as a generated cue. All cues failing both tests are listed in the "definition" dialog box.

The user indicates the cues to add to the input record by choosing, or placing a check mark at the cue. All other cues should be defined as a generated cue.

3.2 Diagnosis

Client diagnosis is the final process of the "Client Selection" function.

The diagnostic activity begins by referencing the client data base record and converting all cues found there into Prolog facts. This is performed by the "process_cdb" predicate called from the "get_client_rec" predicate. All client cues then become facts in the form "cue(cue_name)" for later reference.

Control is then passed to the "produce_diag" predicate where the first activity is production of generated cues with the "develop_cues" predicate.

The "develop_cues" predicate checks each client cue against the definitions for generated cues found in the knowledge base. This is the first implementation of the hybrid chaining control in that the client cue is matched, if possible, to a cue used in defining a generated cue through forward chaining. If a match is found, backward chaining is performed through the "findall", "get_cue_wt_c", and "sum_wts" predicates to determine if enough cues exist to produce the generated cue - that is, if the combined weight of the cues present equals or exceeds 100. If so, the generated cue is asserted by the "assert_cue" predicate in the same form as any other client cue. The "develop_cues" predicate is called a second time to reference any generated cues used in defining higher level generated cues. By calling the "develop_cues" predicate a second time, eight levels of generated cues have been properly processed in testing.

Actual diagnoses are next produced with the "produce_act_diag" predicate. This predicate operates in a similar manner to the "develop_cues" predicate just discussed. The hybrid chaining control is used to find a diagnosis that mentions a client cue as a defining characteristic and the diagnosis is confirmed through other client cues. If the diagnosis can be confirmed with an aggregate cue weight of 100 or more, it is recorded with the "diag_act" predicate. Otherwise, it is recorded as a possible diagnosis with the "diag_pos" predicate. Checks are made along the way to prevent duplicates.

Potential diagnoses are next developed by the "produce_pot_diag" predicate. This predicate operates the same as "produce_act_diag" except it uses risk factor cues (beginning with the character "r") and does not produce any possible diagnoses. A potential diagnosis is recorded with the "diag_pot" predicate. If a diagnosis qualifies as a potential diagnosis, a check is made to determine if it also appears as a possible diagnosis, and, if so, the possible diagnosis predicate (diag_pos) is retracted.

3.3 Displays

3.3.1 Display Diagnoses

Display of actual, possible, and potential diagnoses are handled by the "display_act," "display_pos," and "display_pot" predicates. They reference the knowledge base for diagnoses produced and recorded by the "produce_diag" predicate.

Diagnoses are displayed with the combined weight of the client cues used for their generation. A counter is used so that the display may be paused after five diagnoses have been displayed. The "keyb" predicate allows a key to be pressed to continue.

3.3.2 Explain Diagnoses and Generated Cues

Explanations of actual, possible, and potential diagnoses and generated cues are provided by the "explain_act_diag," "explain_pos_diag," "explain_pot_diag," and "explain_cue" predicates. Each will reference the knowledge base to retrieve and explain the requested item by listing the client cues used in development of the item. The display will be paused after each diagnosis or generated cue and await a key press from the user to continue. Cascaded generated cues will be traced all the way back to foundation cues available from the client data record.

3.4 Printed Reports

All printed reports are produced by redirecting the standard output from the screen to a disk file through the "stdout" predicate. A call to the DOS PRINT command is made through the "shell" predicate to print the file. Since the DOS PRINT command requires identification of the printer, the file "PRNFILE" containing the string "PRN" is directed into the command for printer identification.

3.4.1 Print Client Information

This report prints the client identification and cues found in the client data record through the "print_client" predicate. It will trace generated cues back to the foundation cues found in the client record.

3.4.2 Print Diagnoses

Actual, possible, and potential diagnoses are printed by the "print_act," "print_pos," and "print_pot" predicates. These predicates will print the diagnoses and all defining cues found in the client data record. "print_pos" and "print_pot" also will print all defining characteristic cues used in the diagnosis definitions but not found in the client data record. The predicates "p_other_cues_pos" and "p_other_cues_pot" are used to print these other defining characteristic cues for possible and potential diagnoses, respectively.

3.4.3 Print Definitions

Diagnosis definitions are printed by the "print_def" predicate. It will request input from the user for the specific diagnosis to print and will print the diagnosis with all cues used in its definition. Again, generated cues are traced back to the foundation cues found in the client data record.

3.4.4 Print Input Record Format

The format of the input record is printed by the "print_input" predicate. It references data found in the nursing knowledge base to print each cue and its position in the input record. Not listed on this report is the client number and name which occupy the first 31 positions of the record (position 0-11 is client number, position 12-31 is client name). Also not listed is position 0 of the client cues (in position 32 of the record). Therefore position 1 listed on the report is, in reality, position 33 in the record.

3.5 Save Redefinitions

The current state of the nursing knowledge base may be saved through the "save_db" predicate. This procedure will save all predicates used for diagnosis and generated cue definitions and all input record format predicates. They will be saved in a file named "nurse.ari."

3.6 Quit

The "quit_prolog" predicate uses the "halt" predicate to halt all Prolog operations and return to DOS.

Appendix I

```
'                University of North Florida                   '
'                       Diagnostician                          '

'INPUT RECORD FORMAT                                           '
```

'Position' 1 'is reserved for' r5' Fear of rectal pain'

'Position' 2 'is reserved for' r4' Lack of privacy'

'Position' 3 'is reserved for' r3' Breast-feeding'

'Position' 4 'is reserved for' r2' Lack of exercise'

'Position' 5 'is reserved for' r1' Pregnancy'

'Position' 6 'is reserved for' c8' Abdominal pain'

'Position' 7 'is reserved for' c7' Headache, appetite impairment'

'Position' 8 'is reserved for' c6' Rectal pressure'

'Position' 9 'is reserved for' c5' Abdominal distention'

'Position' 10 'is reserved for' c4' Painful defecation'

'Position' 11 'is reserved for' c3' Straining at stool'

'Position' 12 'is reserved for' c2' Hard, dry stool'

'Position' 13 'is reserved for' c1' Decreased BM frequency'

'Position' 14 'is reserved for' r19' Ill infant'

'Position' 15 'is reserved for' r18' Ill mother'

'Position' 16 'is reserved for' r17' Lack of knowledge - parenting'

'Position' 17 'is reserved for' r16' Nonsupportive partner/family'

'Position' 18 'is reserved for' r15' History of unsuccessful breast-feeding'

'Position' 19 'is reserved for' r14' Inadequate fluid intake'

'Position' 20 'is reserved for' r13' Inadequate Nutrition intake'

'Position' 21 'is reserved for' r12' Maternal ambivalence'

'Position' 22 'is reserved for' r11' Maternal anxiety'

'Position' 23 'is reserved for' r10' Maternal fatigue'

'Position' 24 'is reserved for' r9' Previous breast surgery'

'Position' 25 'is reserved for' r8' Prematurity'

'Position' 26 'is reserved for' r7' Infant anomaly/poor sucking reflex'

'Position' 27 'is reserved for' r6' Breast anomaly'

'Position' 28 'is reserved for' c17' Infant arching and crying at the breast resisting latching on'

'Position' 29 'is reserved for' c16' Infant exhibiting fussiness and crying within the first hour after breast-feeding; unresponsive to other comfort measures'

'Position' 30 'is reserved for' c15' Insufficient opportunity for suckling at the breast'

'Position' 31 'is reserved for' c14' Insufficient emptying of each breast per feeding'

'Position' 32 'is reserved for' c13' Nonsustained suckling at the breast'

'Position' 33 'is reserved for' c12' Observable signs of inadequate infant intake'

'Position' 34 'is reserved for' c11' No observable signs of oxytocin release'

'Position' 35 'is reserved for' c10' Infant inability to attach on to maternal breast correctly'

'Position' 36 'is reserved for' c9' Actual or perceived inadequate milk supply'

'Position' 37 'is reserved for' r20' Trauma (surgery, accidents)'

'Position' 38 'is reserved for' c21' Crying, Moaning'

'Position' 39 'is reserved for' c20' Guarded position'

'Position' 40 'is reserved for' c18' Client reports or demonstrates a discomfort'

'Position' 41 'is reserved for' c26' Dialated pupils'

'Position' 42 'is reserved for' c25' Diaphoresis'

'Position' 43 'is reserved for' c24' Respirations increase in acute pain'

'Position' 44 'is reserved for' c23' Pulse increase in acute pain'

'Position' 45 'is reserved for' c22' Blood pressure increase in acute pain'

'Position' 46 'is reserved for' r22' Birth of a child with defect'

'Position' 47 'is reserved for' r21' Primigravida'

'Position' 48 'is reserved for' c33' Family does not seek or accept help appropriately'

'Position' 49 'is reserved for' c32' Family does not express or accept a wide range of feelings'

'Position' 50 'is reserved for' c31' Family does not meet spiritual needs of all its members'

'Position' 51 'is reserved for' c30' Family does not meet emotional needs of all its members'

'Position' 52 'is reserved for' c29' Family does not meet physical needs of all its members'

'Position' 53 'is reserved for' c28' Family system does not communicate openly and effectively between family members'

'Position' 54 'is reserved for' c27' Family system does not adapt constructively to crisis'

'Position' 55 'is reserved for' r23' Postpartum self-care'

'Position' 56 'is reserved for' r27' Presence of invasive lines (IVs, Foley catheter, enteral feedings)'

'Position' 57 'is reserved for' r26' Altered integumentary system'

'Position' 58 'is reserved for' r25' Blood dyscrasias'

'Position' 59 'is reserved for' r24' Altered or insufficient leukocytes'

'Position' 60 'is reserved for' r30' Postpartum'

'Position' 61 'is reserved for' r29' Crash or fad diet'

'Position' 62 'is reserved for' r28' Lack of knowledge - nutrition'

'Position' 63 'is reserved for' c42' Decreased serum transferrin or iron-binding capacity'

'Position' 64 'is reserved for' c41' Decreased serum albumin'

'Position' 65 'is reserved for' c40' Mental irritability or confusion'

'Position' 66 'is reserved for' c39' Muscle weakness and tenderness'

'Position' 67 'is reserved for' c38' Tachycardia on minimal excercise and bradycardia at rest'

'Position' 68 'is reserved for' c37' Triceps skin fold, mid_arm circumference, and mid_arm muscle circumference less than 60% standard measurement'

'Position' 69 'is reserved for' c36' Weight 10% - 20% below ideal for height and frame'

'Position' 70 'is reserved for' c35' Actual or potential metabolic needs in excess of intake with or without weight loss'

```
'Position'  71  'is reserved for'  c34' Client reports or
     has inadequate food intake, with or without weight loss'
```

'Position' 72 'is reserved for' c48' Sedentary activity patterns'

'Position' 73 'is reserved for' c47' Intake in excess of body requirements'

'Position' 74 'is reserved for' c46' Reported undesirable eating patterns'

'Position' 75 'is reserved for' c45' Triceps skin fold greater than 15mm (men) or 25mm (women)'

'Position' 76 'is reserved for' c44' Obese - more than 20% over ideal for height and frame'

'Position' 77 'is reserved for' c43' Overweight - more than 10% over ideal for height and frame'

'Position' 78 'is reserved for' r41' Unrealistic expectations of self by parent'

'Position' 79 'is reserved for' r40' Unrealistic expectations of child by parent'

'Position' 80 'is reserved for' r39' Lack of extended family'

'Position' 81 'is reserved for' r38' Separation from nuclear family'

'Position' 82 'is reserved for' r37' Child with mental handicap'

'Position' 83 'is reserved for' r36' Child with physical handicap'

'Position' 84 'is reserved for' r35' Child with undesired characteristics'

'Position' 85 'is reserved for' r34' Child of undesired sex'

'Position' 86 'is reserved for' r33' Child of unwanted pregnancy'

'Position' 87 'is reserved for' r32' Adolescent parent'

'Position' 88 'is reserved for' r31' Single parent'

'Position' 89 'is reserved for' c56' Evidence of abuse or neglect of child'

'Position' 90 'is reserved for' c55' Diminished or inappropriate visual, tactile, or auditory stimulation of infant'

'Position' 91 'is reserved for' c54' Verbalization of perceived or actual inadequacy'

'Position' 92 'is reserved for' c53' Verbalization of frustration of role'

'Position' 93 'is reserved for' c52' Frequent verbalization of dissatisfaction or disappointment with infant/child'

'Position' 94 'is reserved for' c51' Lack of parental attachment behavior'

'Position' 95 'is reserved for' c50' Inappropriate parenting behavior'

'Position' 96 'is reserved for' c57' Verbal or nonverbal negative response to actual or perceived change in body structure and/or function'

'Position' 97 'is reserved for' c58' Identification of sexual difficulties, limitations, or changes'

'Position' 98 'is reserved for' r42' Hospitalization'

'Position' 99 'is reserved for' c63' Mood alterations'

'Position' 100 'is reserved for' c62' Agitation'

'Position' 101 'is reserved for' c61' Dozing during the day'

'Position' 102 'is reserved for' c60' Fatigue on awakening or during the day'

'Position' 103 'is reserved for' c59' Difficulty falling or remaining asleep'

'Position' 104 'is reserved for' c70' Pruritus'

'Position' 105 'is reserved for' c69' Lesions'

'Position' 106 'is reserved for' c68' Erythema'

'Position' 107 'is reserved for' c67' Denuded Skin'

'Position' 108 'is reserved for' c66' Cesarean Section'

'Position' 109 'is reserved for' c65' Perineal Laceration'

'Position' 110 'is reserved for' c64' Episiotomy'

'Position' 111 'is reserved for' r45' Irritation to perineal area - poor personal hygiene'

'Position' 112 'is reserved for' r44' Loss of perineal tissue - Childbirth'

'Position' 113 'is reserved for' r43' Post-indwelling catheters'

'Position' 114 'is reserved for' c71' Urgency followed by incontinence'

'Position' 115 'is reserved for' r46' First-time breast-feeder'

'Position' 116 'is reserved for' r48' Cracked nipples'

'Position' 117 'is reserved for' r47' Sore nipples'

'Position' 118 'is reserved for' c74' Cracked nipples'

'Position' 119 'is reserved for' c73' Separation from spouse'

'Position' 120 'is reserved for' c72' Lack of supportive partner/family'

'Position' 121 'is reserved for' r50' Cesarean Section'

'Position' 122 'is reserved for' r49' Episiotomy'

Appendix II

The Nurse knowledge base.

```
last_input_position(122).

assert_input(1) :-
    assertz(cue(r5)).
assert_input(2) :-
    assertz(cue(r4)).
assert_input(3) :-
    assertz(cue(r3)).
assert_input(4) :-
    assertz(cue(r2)).
assert_input(5) :-
    assertz(cue(r1)).
assert_input(6) :-
    assertz(cue(c8)).
assert_input(7) :-
    assertz(cue(c7)).
assert_input(8) :-
    assertz(cue(c6)).
assert_input(9) :-
    assertz(cue(c5)).
assert_input(10) :-
    assertz(cue(c4)).
assert_input(11) :-
    assertz(cue(c3)).
assert_input(12) :-
    assertz(cue(c2)).
assert_input(13) :-
    assertz(cue(c1)).
assert_input(14) :-
    assertz(cue(r19)).
assert_input(15) :-
    assertz(cue(r18)).
assert_input(16) :-
    assertz(cue(r17)).
assert_input(17) :-
    assertz(cue(r16)).
assert_input(18) :-
    assertz(cue(r15)).
assert_input(19) :-
    assertz(cue(r14)).
assert_input(20) :-
    assertz(cue(r13)).
assert_input(21) :-
    assertz(cue(r12)).
```

```
assert_input(22) :-
    assertz(cue(r11)).
assert_input(23) :-
    assertz(cue(r10)).
assert_input(24) :-
    assertz(cue(r9)).
assert_input(25) :-
    assertz(cue(r8)).
assert_input(26) :-
    assertz(cue(r7)).
assert_input(27) :-
    assertz(cue(r6)).
assert_input(28) :-
    assertz(cue(c17)).
assert_input(29) :-
    assertz(cue(c16)).
assert_input(30) :-
    assertz(cue(c15)).
assert_input(31) :-
    assertz(cue(c14)).
assert_input(32) :-
    assertz(cue(c13)).
assert_input(33) :-
    assertz(cue(c12)).
assert_input(34) :-
    assertz(cue(c11)).
assert_input(35) :-
    assertz(cue(c10)).
assert_input(36) :-
    assertz(cue(c9)).
assert_input(37) :-
    assertz(cue(r20)).
assert_input(38) :-
    assertz(cue(c21)).
assert_input(39) :-
    assertz(cue(c20)).
assert_input(40) :-
    assertz(cue(c18)).
assert_input(41) :-
    assertz(cue(c26)).
assert_input(42) :-
    assertz(cue(c25)).
assert_input(43) :-
    assertz(cue(c24)).
assert_input(44) :-
    assertz(cue(c23)).
assert_input(45) :-
    assertz(cue(c22)).
assert_input(46) :-
    assertz(cue(r22)).
```

```
assert_input(47) :-
    assertz(cue(r21)).
assert_input(48) :-
    assertz(cue(c33)).
assert_input(49) :-
    assertz(cue(c32)).
assert_input(50) :-
    assertz(cue(c31)).
assert_input(51) :-
    assertz(cue(c30)).
assert_input(52) :-
    assertz(cue(c29)).
assert_input(53) :-
    assertz(cue(c28)).
assert_input(54) :-
    assertz(cue(c27)).
assert_input(55) :-
    assertz(cue(r23)).
assert_input(56) :-
    assertz(cue(r27)).
assert_input(57) :-
    assertz(cue(r26)).
assert_input(58) :-
    assertz(cue(r25)).
assert_input(59) :-
    assertz(cue(r24)).
assert_input(60) :-
    assertz(cue(r30)).
assert_input(61) :-
    assertz(cue(r29)).
assert_input(62) :-
    assertz(cue(r28)).
assert_input(63) :-
    assertz(cue(c42)).
assert_input(64) :-
    assertz(cue(c41)).
assert_input(65) :-
    assertz(cue(c40)).
assert_input(66) :-
    assertz(cue(c39)).
assert_input(67) :-
    assertz(cue(c38)).
assert_input(68) :-
    assertz(cue(c37)).
assert_input(69) :-
    assertz(cue(c36)).
assert_input(70) :-
    assertz(cue(c35)).
assert_input(71) :-
    assertz(cue(c34)).
```

```
assert_input(72) :-
    assertz(cue(c48)).
assert_input(73) :-
    assertz(cue(c47)).
assert_input(74) :-
    assertz(cue(c46)).
assert_input(75) :-
    assertz(cue(c45)).
assert_input(76) :-
    assertz(cue(c44)).
assert_input(77) :-
    assertz(cue(c43)).
assert_input(78) :-
    assertz(cue(r41)).
assert_input(79) :-
    assertz(cue(r40)).
assert_input(80) :-
    assertz(cue(r39)).
assert_input(81) :-
    assertz(cue(r38)).
assert_input(82) :-
    assertz(cue(r37)).
assert_input(83) :-
    assertz(cue(r36)).
assert_input(84) :-
    assertz(cue(r35)).
assert_input(85) :-
    assertz(cue(r34)).
assert_input(86) :-
    assertz(cue(r33)).
assert_input(87) :-
    assertz(cue(r32)).
assert_input(88) :-
    assertz(cue(r31)).
assert_input(89) :-
    assertz(cue(c56)).
assert_input(90) :-
    assertz(cue(c55)).
assert_input(91) :-
    assertz(cue(c54)).
assert_input(92) :-
    assertz(cue(c53)).
assert_input(93) :-
    assertz(cue(c52)).
assert_input(94) :-
    assertz(cue(c51)).
assert_input(95) :-
    assertz(cue(c50)).
assert_input(96) :-
    assertz(cue(c57)).
```

```
assert_input(97) :-
    assertz(cue(c58)).
assert_input(98) :-
    assertz(cue(r42)).
assert_input(99) :-
    assertz(cue(c63)).
assert_input(100) :-
    assertz(cue(c62)).
assert_input(101) :-
    assertz(cue(c61)).
assert_input(102) :-
    assertz(cue(c60)).
assert_input(103) :-
    assertz(cue(c59)).
assert_input(104) :-
    assertz(cue(c70)).
assert_input(105) :-
    assertz(cue(c69)).
assert_input(106) :-
    assertz(cue(c68)).
assert_input(107) :-
    assertz(cue(c67)).
assert_input(108) :-
    assertz(cue(c66)).
assert_input(109) :-
    assertz(cue(c65)).
assert_input(110) :-
    assertz(cue(c64)).
assert_input(111) :-
    assertz(cue(r45)).
assert_input(112) :-
    assertz(cue(r44)).
assert_input(113) :-
    assertz(cue(r43)).
assert_input(114) :-
    assertz(cue(c71)).
assert_input(115) :-
    assertz(cue(r46)).
assert_input(116) :-
    assertz(cue(r48)).
assert_input(117) :-
    assertz(cue(r47)).
assert_input(118) :-
    assertz(cue(c74)).
assert_input(119) :-
    assertz(cue(c73)).
assert_input(120) :-
    assertz(cue(c72)).
assert_input(121) :-
    assertz(cue(r50)).
```

```
assert_input(122) :-
    assertz(cue(r49)).

diagnose(d1,c1,100).
diagnose(d1,c2,100).
diagnose(d1,c3,75).
diagnose(d1,c4,75).
diagnose(d1,c5,75).
diagnose(d1,c6,50).
diagnose(d1,c7,50).
diagnose(d1,c8,50).
diagnose(d1,r1,100).
diagnose(d1,r2,25).
diagnose(d1,r4,25).
diagnose(d1,r5,100).
diagnose(d2,c9,25).
diagnose(d2,c10,75).
diagnose(d2,c11,25).
diagnose(d2,c12,75).
diagnose(d2,c13,50).
diagnose(d2,c14,50).
diagnose(d2,c15,75).
diagnose(d2,c16,50).
diagnose(d2,c17,75).
diagnose(d2,r6,100).
diagnose(d2,r7,100).
diagnose(d2,r8,100).
diagnose(d2,r9,25).
diagnose(d2,r10,25).
diagnose(d2,r11,25).
diagnose(d2,r12,100).
diagnose(d2,r13,25).
diagnose(d2,r14,25).
diagnose(d2,r15,50).
diagnose(d2,r16,50).
diagnose(d2,r17,75).
diagnose(d2,r18,75).
diagnose(d2,r19,75).
diagnose(d3,c18,100).
diagnose(d3,c19,75).
diagnose(d3,c20,50).
diagnose(d3,c21,75).
diagnose(d3,r20,100).
diagnose(d4,c27,50).
diagnose(d4,c28,50).
diagnose(d4,c29,50).
diagnose(d4,c30,50).
diagnose(d4,c31,50).
diagnose(d4,c32,50).
diagnose(d4,c33,50).
diagnose(d4,r22,100).
diagnose(d5,r17,100).
diagnose(d6,r24,100).
diagnose(d6,r25,100).
diagnose(d6,r26,100).
diagnose(d6,r27,100).
diagnose(d6,r20,100).
diagnose(d7,c34,100).
diagnose(d7,c35,100).
diagnose(d7,c36,75).
diagnose(d7,c37,75).
diagnose(d7,c38,25).
diagnose(d7,c39,25).
diagnose(d7,c40,25).
diagnose(d7,c41,75).
diagnose(d7,c42,50).
diagnose(d7,r28,100).
diagnose(d7,r29,100).
diagnose(d8,c43,100).
diagnose(d8,c44,100).
diagnose(d8,c45,100).
diagnose(d8,c46,75).
diagnose(d8,c47,75).
diagnose(d8,c48,25).
diagnose(d8,r1,100).
diagnose(d9,c50,100).
diagnose(d9,c51,100).
diagnose(d9,c52,75).
diagnose(d9,c53,50).
diagnose(d9,c54,50).
diagnose(d9,c55,25).
diagnose(d9,c56,100).
diagnose(d9,r31,50).
diagnose(d9,r32,50).
diagnose(d9,r33,50).
diagnose(d9,r34,50).
diagnose(d9,r35,50).
diagnose(d9,r36,75).
diagnose(d9,r37,75).
diagnose(d9,r38,50).
diagnose(d9,r39,50).
diagnose(d9,r17,50).
diagnose(d9,r40,50).
diagnose(d9,r41,50).
diagnose(d10,c57,100).
diagnose(d10,r1,100).
diagnose(d11,c58,100).
diagnose(d11,r30,100).
diagnose(d12,c59,100).
diagnose(d12,c60,50).
diagnose(d12,c61,50).
diagnose(d12,c62,50).
diagnose(d12,c63,50).
diagnose(d12,r42,100).
diagnose(d12,r30,100).
diagnose(d13,c64,100).
diagnose(d13,c65,100).
diagnose(d13,c66,100).
diagnose(d13,c67,50).
diagnose(d13,c68,25).
diagnose(d13,c69,25).
diagnose(d13,c70,25).
diagnose(d14,c71,100).
diagnose(d14,r43,50).
diagnose(d14,r44,100).
diagnose(d14,r45,50).
diagnose(d1,r30,100).
diagnose(d2,r3,50).
diagnose(d2,r46,100).
diagnose(d4,r30,100).
diagnose(d10,r30,100).
diagnose(d5,r23,50).
diagnose(d5,r21,50).
diagnose(d7,r30,50).
diagnose(d4,c72,100).
diagnose(d11,c73,100).
diagnose(d13,c74,100).
diagnose(d2,r47,100).
diagnose(d2,r48,100).
diagnose(d6,r49,100).
diagnose(d6,r50,100).
diagnose(d3,c64,100).
diagnose(d3,c66,100).

diag_descr(d8,$Altered Nutrition: More than body req.$).
diag_descr(d9,$Altered Parenting$).
diag_descr(d12,$Sleep Pattern Disturbance$).
diag_descr(d14,$Urge Incontinence$).
diag_descr(d10,$Body Image Disturbance$).
diag_descr(d5,$Altered Health Maintenance$).
diag_descr(d7,$Altered Nutrition: Less than body req.$).
diag_descr(d4,$Altered Family Processes$).
diag_descr(d11,$Altered Sexuality Patterns$).
diag_descr(d13,$Impaired Skin Integrity$).
diag_descr(d6,$Infection$).
diag_descr(d1,$Colonic Constipation$).
diag_descr(d2,$Ineffective Breast-Feeding$).
diag_descr(d3,$Altered Comfort$).

cue(c19,c22,35).
cue(c19,c23,35).
cue(c19,c24,35).
cue(c19,c25,25).
cue(c19,c26,25).

cue(c1,$Decreased BM frequency$).
cue(c2,$Hard, dry stool$).
cue(c3,$Straining at stool$).
cue(c4,$Painful defecation$).
cue(c5,$Abdominal distention$).
cue(c6,$Rectal pressure$).
cue(c7,$Headache, appetite impairment$).
cue(c8,$Abdominal pain$).
cue(r1,$Pregnancy$).
cue(r2,$Lack of exercise$).
cue(r4,$Lack of privacy$).
cue(r5,$Fear of rectal pain$).
cue(c9,$Actual or perceived inadequate milk supply$).
cue(c10,$Infant inability to attach on to maternal breast
correctly$).
cue(c11,$No observable signs of oxytocin release$).
cue(c12,$Observable signs of inadequate infant intake$).
cue(c13,$Nonsustained suckling at the breast$).
cue(c14,$Insufficient emptying of each breast per feeding$).
cue(c15,$Insufficient opportunity for suckling at the
breast$).
cue(c16,$Infant exhibiting fussiness and crying within the
first hour after breast-feeding; unresponsive to other comfort
measures$).
cue(c17,$Infant arching and crying at the breast resisting
latching on$).
cue(r6,$Breast anomaly$).
cue(r7,$Infant anomaly/poor sucking reflex$).
cue(r8,$Prematurity$).
cue(r9,$Previous breast surgery$).
cue(r10,$Maternal fatigue$).
cue(r11,$Maternal anxiety$).
cue(r13,$Inadequate Nutrition intake$).
cue(r14,$Inadequate fluid intake$).
cue(r15,$History of unsuccessful breast-feeding$).
cue(r16,$Nonsupportive partner/family$).
cue(r18,$Ill mother$).
cue(r19,$Ill infant$).
cue(c18,$Client reports or demonstrates a discomfort$).
cue(c20,$Guarded position$).
cue(c21,$Crying, Moaning$).
cue(r20,$Trauma (surgery, accidents)$).
cue(c22,$Blood pressure increase in acute pain$).
cue(c23,$Pulse increase in acute pain$).
cue(c24,$Respirations increase in acute pain$).
cue(c25,$Diaphoresis$).
cue(c26,$Dilated pupils$).
cue(c19,$Autonomic response in acute pain$).
cue(c27,$Family system does not adapt constructively to
crisis$).
cue(c28,$Family system does not communicate openly and
effectively between family members$).
cue(c29,$Family does not meet physical needs of all its
members$).
cue(c30,$Family does not meet emotional needs of all its
members$).
cue(c31,$Family does not meet spiritual needs of all its
members$).
cue(c32,$Family does not express or accept a wide range of
feelings$).
cue(c33,$Family does not seek or accept help appropriately$).
cue(r22,$Birth of a child with defect$).
cue(r23,$Postpartum self-care$).
cue(r17,$Lack of knowledge - parenting$).
cue(r24,$Altered or insufficient leukocytes$).
cue(r25,$Blood dyscrasias$).
cue(r26,$Altered integumentary system$).
cue(r27,$Presence of invasive lines (IVs, Foley catheter,
enteral feedings)$).
cue(c34,$Client reports or has inadequate food intake, with or
without weight loss$).
cue(c35,$Actual or potential metabolic needs in excess of
intake with or without weight loss$).
cue(c36,$Weight 10% - 20% below ideal for height and frame$).
cue(c37,$Triceps skin fold, mid_arm circumference, and mid_arm
muscle circumference less than 60% standard measurement$).
cue(c38,$Tachycardia on minimal excercise and bradycardia at
rest$).
cue(c39,$Muscle weakness and tenderness$).
cue(c40,$Mental irritability or confusion$).
cue(c41,$Decreased serum albumin$).
cue(c42,$Decreased serum transferrin or iron-binding
capacity$).
cue(r28,$Lack of knowledge - nutrition$).
cue(r29,$Crash or fad diet$).
cue(c43,$Overweight - more than 10% over ideal for height and
frame$).
cue(c44,$Obese - more than 20% over ideal for height and
frame$).
cue(c45,$Triceps skin fold greater than 15mm (men) or 25mm
(women)$).
cue(c46,$Reported undesirable eating patterns$).
cue(c47,$Intake in excess of body requirements$).
cue(c48,$Sedentary activity patterns$).
cue(c50,$Inappropriate parenting behavior$).
cue(c51,$Lack of parental attachment behavior$).
cue(c52,$Frequent   verbalization   of   dissatisfaction   or
disappointment with infant/child$).
cue(c53,$Verbalization of frustration of role$).
cue(c54,$Verbalization of perceived or actual inadequacy$).
cue(c55,$Diminished  or  inappropriate  visual,  tactile,  or
auditory stimulation of infant$).
cue(c56,$Evidence of abuse or neglect of child$).
cue(r31,$Single parent$).
cue(r32,$Adolescent parent$).
cue(r33,$Child of unwanted pregnancy$).
cue(r34,$Child of undesired sex$).
cue(r35,$Child with undesired characteristics$).
cue(r36,$Child with physical handicap$).
cue(r37,$Child with mental handicap$).
cue(r38,$Separation from nuclear family$).
cue(r39,$Lack of extended family$).
cue(r40,$Unrealistic expectations of child by parent$).
cue(r41,$Unrealistic expectations of self by parent$).
cue(c57,$Verbal or nonverbal negative response to actual or
perceived change in body structure and/or function$).
cue(c58,$Identification of sexual difficulties, limitations,
or changes$).
cue(r30,$Postpartum$).
cue(c59,$Difficulty falling or remaining asleep$).
cue(c60,$Fatigue on awakening or during the day$).
cue(c61,$Dozing during the day$).
cue(c62,$Agitation$).
cue(c63,$Mood alterations$).
cue(r42,$Hospitalization$).
cue(c64,$Episiotomy$).
cue(c65,$Perineal Laceration$).
cue(c66,$Cesarean Section$).
cue(c67,$Denuded Skin$).
cue(c68,$Erythema$).
cue(c69,$Lesions$).
cue(c70,$Pruritus$).
cue(c71,$Urgency followed by incontinence$).
cue(r43,$Post-indwelling catheters$).
cue(r44,$Loss of perineal tissue - Childbirth$).
cue(r45,$Irritation   to   perineal   area   -   poor   personal
hygiene$).
cue(r3,$Breast-feeding$).
cue(r46,$First-time breast-feeder$).
cue(r21,$Primigravida$).
cue(c72,$Lack of supportive partner/family$).
cue(c73,$Separation from spouse$).
cue(c74,$Cracked nipples$).
cue(r47,$Sore nipples$).
cue(r48,$Cracked nipples$).
cue(r49,$Episiotomy$).
cue(r50,$Cesarean Section$).
cue(r12,$Maternal ambivalence toward breast-feeding$).
```

APPENDIX E

SOURCE PROGRAM LISTINGS

```
/*                          University of North Florida
                                   Diagnostician

                   A Master of Computer and Information Sciences
                                  Thesis Project

Development Vehicle: Arity Prolog, version 5.1
Source File:  prolog.ini


       Initial file loaded and executed when prolog interpreter
       is initiated.
*/
:-
%      ************************************************************
[-diag],                   % load diagnostic subsystem
[-nurse],                  % load diagnoses, hi_cues and input format

%      ************************************************************

%  window definitions

%      ************************************************************

define_window(background,'University of North Florida Diagnostician',
              (0,0),(24,79),(112,-23)),
define_window(foreground,'',(1,1),(20,78),(112,0)),
define_window(cnumber,'Client Number',(21,1),(23,20),(112,23)),
define_window(cname,'Client Name',(21,25),(23,50),(112,23)),
define_window(message,'',(21,60),(23,78),(112,0)),

%      ************************************************************

current_window(_,background),
current_window(_,cnumber),
current_window(_,cname),
current_window(_,message),
current_window(_,foreground),
do_top_menu.                            % execute top level menu
```

```
/*
                        University of North Florida
                              Diagnostician

                   A Master of Computer and Information Sciences
                                Thesis Project


Development Vehicle: Arity Prolog
Source File:            diag.ari
*/
%     *************************************************************

%  dialog box definitions

%     *************************************************************
begin_dialog(change_cues,'',(1,1),(23,79),(112,23),16,popup).
ctrl(text,0,$                    $,(0,1),78,25). % 1 - function text
ctrl(efield,1,_,(1,1),(78,23),5,$$).          % 2 - diag field
ctrl(efield,1,_,(1,9),(78,23),40,$$).         % 3 - diag description
ctrl(push,1,$Add Cue$,(1,66),(74,30),add_cue).       % 4
ctrl(list_box,1,$Cues$,(4,1),(17,9),(78,23),radio,(1,1),cues). % 5
ctrl(list_box,0,$Wts$,(4,10),(17,17),(78,23),radio,(1,1),wts). % 6
ctrl(list_box,0,$Descriptions$,(4,18),(17,60),(78,23),radio,(1,1),descrips).
ctrl(efield,1,_,(18,1),(78,23),6,$$).   % 8 - cue field
ctrl(efield,1,_,(18,10),(78,23),5,$$).  % 9 - wt field for add_cue
ctrl(efield,1,_,(18,18),(78,23),40,$$). % 10 - cue description field
ctrl(push,1,$Delete Cue$,(5,63),(74,30),delete_cue). % 11
ctrl(push,1,$Change Wt$,(8,64),(74,30),change_wt).   % 12
ctrl(push,1,$Change Descr$,(11,61),(74,30),change_descr). % 13
ctrl(push,1,$Exit$,(18,69),(74,30),exit).            % 14
ctrl(efield,1,_,(18,10),(78,23),5,$$).  % wt field for change_wts % 15
end_dialog(change_cues).


%     *************************************************************
begin_dialog(definition,'Cue Definition Check',(1,1),(23,79),(112,23),16,popup).
ctrl(text,0,$The following cues need further definition.$,(1,1),78,45). % 1
ctrl(text,0,$Mark the ones to add to the input record.$,(2,1),78,45).    % 2
ctrl(text,0,$Define the others as high level cues.$,(3,1),78,45).  % 3
ctrl(list_box,1,$Cues$,(4,1),(17,9),(78,23),choice,(1,1),cues).          % 4
ctrl(list_box,0,$Descriptions$,(4,10),(17,60),(78,23),radio,(1,1),descrips).
ctrl(push,1,$Exit$,(18,69),(74,30),exit).           % 6
end_dialog(definition).


%     *************************************************************
begin_dialog(cselect,'Client Selection',(1,1),(23,79),(112,23),16,popup).
ctrl(text,0,$Select Client to diagnose$,(2,1),78,30).                   % 1
ctrl(list_box,1,$Number$,(4,1),(17,15),(78,23),radio,(1,1),number).      % 2
ctrl(list_box,0,$Client Name$,(4,16),(17,40),(78,23),radio,(1,1),name). % 3
ctrl(push,1,$Continue$,(18,60),(74,30),exit).                     % 4
end_dialog(cselect).

%     *************************************************************

%  menu definition

%     *************************************************************
begin_menu(top_menu,75,colors((23,64),(23,64),(55,71),(78,23))).
item($ ~Diagnose    $,
  [item($~Select Client$,select_client),
   item($Display ~Actual diagnoses $,diag_act),
   item($Display ~Possible diagnoses$,diag_pos),
   item($Display P~otential diagnoses$,diag_pot)]).
item($ ~Explain     $,
  [item($~Actual diagnoses generated$,explain_act_diag),
   item($~Possible diagnoses generated$,explain_pos_diag),
   item($P~otential diagnoses generated$,explain_pot_diag),
   item($~Cues generated$,explain_cue)]).
item($ ~Print      $,
   [item($~Client information$,print_client),
    item($~Actual diagnoses$,print_act),
    item($~Possible diagnoses$,print_pos),
    item($P~otential diagnoses$,print_pot),
    break,
    item($~Diagnosis definition$,print_def),
    item($~Input format$,print_input)]).
item($ ~Redefine   $,
   [item($~Add diagnosis$,add_diag),
    item($~Change diagnosis$,change_diag),
    item($~Delete diagnosis$,delete_diag),
    break,
    item($Add ~high level cue$,add_cue),
    item($Change h~igh level cue$,change_cue),
    item($Delete hi~gh level cue$,delete_cue),
    break,
    item($Ch~eck cue definitions$,check_cue),
    item($~Save redefinitions$,save)]).
item($ ~Quit        $,
    [item($~Return to DOS$,quit_prolog)]).
end_menu(top_menu).


%      ***************************************************************

%  menu processing

%      ***************************************************************
do_top_menu :-
  cls,
  current_window(_,message),            % check for new cue definition
  cls,
  ifthen(new_cue,                  % if new_cue present
        (display('CHECK CUE'),          % display reminder
         nl,
         display('DEFINITIONS'))),
  current_window(_,foreground),
  send_menu_msg(activate(top_menu,(0,0)),Selection),    % activate menu
  case([
        Selection=select_client->select_client,
        Selection=diag_act->display_act,
        Selection=diag_pos->display_pos,
        Selection=diag_pot->display_pot,
      Selection=explain_act_diag->explain_act_diag,
      Selection=explain_pos_diag->explain_pos_diag,
      Selection=explain_pot_diag->explain_pot_diag,
      Selection=explain_cue->explain_cue,
      Selection=print_client->print_client,
        Selection=print_act->print_act,
        Selection=print_pos->print_pos,
        Selection=print_pot->print_pot,
        Selection=print_def->print_def,
        Selection=print_input->print_input,
      Selection=add_diag->add_diag,
      Selection=change_diag->change_diag,
      Selection=delete_diag->delete_diag,
      Selection=add_cue->add_hi_cue,
      Selection=change_cue->change_hi_cue,
      Selection=delete_cue->delete_hi_cue,
      Selection=check_cue->check_cue,
        Selection=save->save_db,
      Selection=quit_prolog->quit_prolog
       ]),
 do_top_menu.                          % recurse


%      ***************************************************************

%  menu selection - Add Diagnosis

%      ***************************************************************
```

```
add_diag :-                                 % add new diagnosis
  dialog_run(change_cues,get_msg_ad).          % activate dialog box

get_msg_ad(command(nobutton,ok),change_cues) :-      % Enter key pressed
  which_control(Old),                          % get current control number
  New is Old + 1,                           % proceed to next control
  send_dialog_msg(get_msg_ad,next_tabstop(Old,New),change_cues). % reroute

get_msg_ad(init_dialog,change_cues) :-         % initialize dialog box
  [!
   send_control_msg(text_set(_,$Diagnosis to add$),1,change_cues),
   send_control_msg(ef_set_text(_,$$),2,change_cues),       % clear edit fields
   send_control_msg(ef_set_text(_,$$),3,change_cues),
   send_control_msg(ef_set_text(_,$$),8,change_cues),
   send_control_msg(ef_set_text(_,$$),9,change_cues),
   send_control_msg(ef_set_text(_,$$),10,change_cues),
   send_control_msg(ef_set_text(_,$$),15,change_cues),
   send_control_msg(lb_clear,5,change_cues),          % clear list boxes
   send_control_msg(lb_clear,6,change_cues),
   send_control_msg(lb_clear,7,change_cues),
   send_control_msg(lb_insert_string($$,0),5,change_cues), % insert null str
   send_control_msg(lb_insert_string($$,0),6,change_cues),
   send_control_msg(lb_insert_string($$,0),7,change_cues)
  !],
   fail.

get_msg_ad(next_ctrl(2,15),change_cues) :-     % diag field - backward
  send_dialog_msg(get_msg_ad,next_ctrl(2,14),change_cues). % reroute to Exit

get_msg_ad(next_ctrl(2,_),change_cues) :-      % leaving diag name field
 [!
  get_diag_from_e(Diag),                    % get diag name
  ifthen(diag_descr(Diag,Diag_descr),          % display diag desciption
        (send_control_msg(ef_set_text(_,Diag_descr),3,change_cues),
         send_control_msg(update,3,change_cues)))
 !],
  fail.

get_msg_ad(next_ctrl(3,4),change_cues) :-      % diag description field- fwd
 [!
  get_diag_from_e(Diag),                    % get diag name & descrip
  send_control_msg(ef_set_text(Diag_descr,Diag_descr),3,change_cues),
  ifthen(diag_descr(Diag,_),                   % if descrip already present
         retract(diag_descr(Diag,_))),         % delete it
  assertz(diag_descr(Diag,Diag_descr))         % add it
 !],
  fail .

get_msg_ad(next_ctrl(8,5),change_cues) :-      % cue name field - backward
  send_dialog_msg(get_msg_ad,next_ctrl(8,4),change_cues).   % reroute

get_msg_ad(next_ctrl(9,10),change_cues) :-     % cue weight fld - forward
 [!                                         % record new diag/cue rule
  send_control_msg(ef_set_text(Ecue,$$),8,change_cues),   % get new cue
  string_term(Ecue,Cue),                    % convert to term
  send_control_msg(ef_set_text(Ewt,$$),9,change_cues),    % get wt
  string_term(Ewt,Wt),                         % convert to term
  send_control_msg(lb_add_string(Cue),5,change_cues),   % add cue name
  send_control_msg(lb_add_string(Wt),6,change_cues), % add cue wt
  get_diag_from_e(Diag),                    % get diag name
  ifthenelse(Wt > 0,                           % edit wt > 0
             assertz(diagnose(Diag,Cue,Wt)),   % record new rule or
             assertz(diagnose(Diag,Cue,0))),   % else record edited rule
  ifthenelse(cue(Cue,Descrip),              % display descrip or set flag
       send_control_msg(lb_add_string(Descrip),7,change_cues),
       (send_control_msg(lb_add_string(no_description),7,change_cues),
        assertz(new_cue))),                 % reminder to check cue definitions
  send_control_msg(lb_get_count(Count),5,change_cues),      % get # in list
  send_control_msg(lb_set_index(_,Count),5,change_cues), % set index to end
  send_control_msg(lb_set_index(_,Count),6,change_cues), % set index to end
  send_control_msg(lb_set_index(_,Count),7,change_cues), % set index to end
  send_control_msg(update,5,change_cues),           % update controls
```

```
   send_control_msg(update,6,change_cues),
   send_control_msg(update,7,change_cues),
   send_control_msg(update,8,change_cues),
   send_control_msg(update,9,change_cues)
  !],
   fail.

get_msg_ad(next_ctrl(10,11),change_cues) :-    % cue description field - fwd
  send_control_msg(ef_get_length(L),10,change_cues), % get lgth of input
  ifthen(L>0,                               % if new description has been entered
   (send_control_msg(ef_set_text(Descrip,$$),10,change_cues), % get descrip
    get_cue_text(Cue,Ndx),                  % get index from cue column
    send_control_msg(lb_delete_string(Ndx,_),7,change_cues), % delete old dsc
    Indx is Ndx - 1,                        % insert new cue descrip
    send_control_msg(lb_insert_string(Descrip,Indx),7,change_cues),
    send_control_msg(lb_set_index(_,Ndx),7,change_cues), % reset index
    send_control_msg(update,7,change_cues),
    send_control_msg(update,10,change_cues),
    ifthen(cue(Cue,_),                      % if descrip already present
           retract(cue(Cue,_))), % delete it
    assertz(cue(Cue,Descrip)))), % add new description
  send_dialog_msg(get_msg_ad,next_ctrl(10,4),change_cues). % reroute

get_msg_ad(next_ctrl(4,5),change_cues) :-      % Add button - forward
  send_dialog_msg(get_msg_ad,next_ctrl(4,14),change_cues). % reroute to Exit

get_msg_ad(next_ctrl(14,13),change_cues) :-    % Exit button - backward
  send_dialog_msg(get_msg_ad,next_ctrl(14,4),change_cues). % reroute to Add

get_msg_ad(next_ctrl(14,15),change_cues) :-     % Exit button - forward
  send_dialog_msg(get_msg_ad,next_ctrl(14,2),change_cues). % reroute to diag

get_msg_ad(command(_,add_cue),change_cues) :- % Add cue pushbutton
  send_dialog_msg(get_msg_ad,next_ctrl(4,8),change_cues).


get_msg_ad(command(_,exit),change_cues) :-     % Exit pushbutton
  !,                                      % cut
  exit_dbox(change_cues).                 % exit box

get_msg_ad(Msg,Key) :-                         % default dialog box functions
  def_dialog_fn(Msg,Key).


%      ****************************************************************

%  menu selection processing - Change Diagnosis

%      ****************************************************************

change_diag :-                            % change diagnosis
  dialog_run(change_cues,get_msg_cd).     % activate dialog box

get_msg_cd(command(nobutton,ok),change_cues) :-      % Enter key pressed
  which_control(Old),                     % get current control number
  ifthenelse(Old == 15,                   % if last one
             New is 1,                    % restart
             New is Old + 1),             % else add 1
  send_dialog_msg(get_msg_cd,next_tabstop(Old,New),change_cues). % reroute

get_msg_cd(init_dialog,change_cues) :-  % initialize dialog box
 [!
  send_control_msg(text_set(_,$Diagnosis to change$),1,change_cues),
  send_control_msg(ef_set_text(_,$$),2,change_cues), % clear edit fields
  send_control_msg(ef_set_text(_,$$),3,change_cues),
  send_control_msg(ef_set_text(_,$$),8,change_cues),
  send_control_msg(ef_set_text(_,$$),9,change_cues),
  send_control_msg(ef_set_text(_,$$),10,change_cues),
  send_control_msg(ef_set_text(_,$$),15,change_cues),
  send_control_msg(lb_clear,5,change_cues),          % clear list boxes
  send_control_msg(lb_clear,6,change_cues),
  send_control_msg(lb_clear,7,change_cues),
  send_control_msg(lb_insert_string($$,0),5,change_cues), % insert null strng
  send_control_msg(lb_insert_string($$,0),6,change_cues),
  send_control_msg(lb_insert_string($$,0),7,change_cues)
 !],
  fail.

get_msg_cd(next_ctrl(2,15),change_cues) :-     % diag name - backward
  send_dialog_msg(get_msg_cd,next_ctrl(2,14),change_cues). % reroute to Exit

get_msg_cd(next_ctrl(2,_),change_cues) :-      % leaving diag field
 [!                                   % get/display cues/descrip
  get_diag_from_e(Diag),              % get diag name
  ifthen(diag_descr(Diag,Diag_descr),          % display diag descrip
       (send_control_msg(ef_set_text(_,Diag_descr),3,change_cues),
        send_control_msg(update,3,change_cues))),
  send_control_msg(lb_clear,5,change_cues),    % clear boxes
  send_control_msg(lb_clear,6,change_cues),
  send_control_msg(lb_clear,7,change_cues),
  add_to_list_d,                      % add cues to boxes
  send_control_msg(update,5,change_cues),      % update controls
  send_control_msg(update,6,change_cues),
  send_control_msg(update,7,change_cues)
 !],
  fail.

get_msg_cd(next_ctrl(3,4),change_cues) :-      % diag description - fwd
 [!
  get_diag_from_e(Diag),              % get diag name
  send_control_msg(ef_set_text(Diag_descr,Diag_descr),3,change_cues), % descr
  ifthen(diag_descr(Diag,_),                   % if descrip already present
         retract(diag_descr(Diag,_))),         % delete it
  assertz(diag_descr(Diag,Diag_descr))         % add it
 !],
  fail.

get_msg_cd(next_ctrl(5,8),change_cues) :-      % cue name box - fwd
  send_dialog_msg(get_msg_cd,next_ctrl(5,11),change_cues). % reroute to Delete

get_msg_cd(next_ctrl(9,10),change_cues) :-     % cue wt field - fwd
 [!                                   % record new diag/cue rule
  send_control_msg(ef_set_text(Ecue,$$),8,change_cues),     % get new cue
  string_term(Ecue,Cue),                       % convert to term
  send_control_msg(ef_set_text(Ewt,$$),9,change_cues),      % get new cue wt
  string_term(Ewt,Wt),                                 % convert to term
  send_control_msg(lb_add_string(Cue),5,change_cues),       % add cue to list box
  send_control_msg(lb_add_string(Wt),6,change_cues), % add wt to list box
  get_diag_from_e(Diag),              % get diag name
  ifthenelse(Wt>0,                    % edit wt > 0
             assertz(diagnose(Diag,Cue,Wt)),   % record new diag/cue rule
             assertz(diagnose(Diag,Cue,0))),   % else record edited rule
  ifthenelse(cue(Cue,Descrip),                 % if cue descrip present
         send_control_msg(lb_add_string(Descrip),7,change_cues), % display it
         (send_control_msg(lb_add_string(no_description),7,change_cues),
        assertz(new_cue))),                    % else set reminder flag
  send_control_msg(lb_get_count(Count),5,change_cues),      % get # in list
  send_control_msg(lb_set_index(_,Count),5,change_cues), % set index to end
  send_control_msg(lb_set_index(_,Count),6,change_cues),
  send_control_msg(lb_set_index(_,Count),7,change_cues),
  send_control_msg(update,5,change_cues),              % update controls
  send_control_msg(update,6,change_cues),
  send_control_msg(update,7,change_cues),
  send_control_msg(update,8,change_cues),
  send_control_msg(update,9,change_cues)
 !],
  fail.

get_msg_cd(next_ctrl(10,11),change_cues) :-    % cue descrip field - fwd
  send_control_msg(ef_get_length(L),10,change_cues), % get lgth of input
  ifthen(L>0,                                  % if new descrip entered
   (send_control_msg(ef_set_text(Descrip,$$),10,change_cues),      % get text
    get_cue_text(Cue,Ndx),                     % get index of cue
    send_control_msg(lb_delete_string(Ndx,_),7,change_cues),       % delete desc
```

```
     Indx is Ndx - 1,                             % add descrip to box
     send_control_msg(lb_insert_string(Descrip,Indx),7,change_cues),
     send_control_msg(update,7,change_cues),    % update controls
     send_control_msg(update,10,change_cues),
     ifthen(cue(Cue,_),                           % if descrip already present
            retract(cue(Cue,_))),           % delete it
     assertz(cue(Cue,Descrip)))),           % add it
   send_dialog_msg(get_msg_cd,next_ctrl(10,5),change_cues). % reroute to box

 get_msg_cd(next_ctrl(11,10),change_cues) :-    % Delete Cue - backward
   send_dialog_msg(get_msg_cd,next_ctrl(11,5),change_cues). % reroute to box

 get_msg_cd(next_ctrl(14,15),change_cues) :-     % Exit - forward
   send_dialog_msg(get_msg_cd,next_ctrl(14,2),change_cues). % reroute to diag

 get_msg_cd(next_ctrl(15,2),change_cues) :-     % cue weight field - forward
   get_diag_from_e(Diag),                    % get diag name
   get_cue_text(Cue,Ndx),                    % get cue index
   retract(diagnose(Diag,Cue,_)),            % delete old diag/cue rule
   send_control_msg(ef_set_text(Ewt,$$),15,change_cues), % get new wt
   string_term(Ewt,Wt),                           % convert to term
   ifthenelse(Wt>0,                          % edit wt > 0
              assertz(diagnose(Diag,Cue,Wt)),   % record new diag/cue rule
              assertz(diagnose(Diag,Cue,0))),   % else record edited rule
   send_control_msg(lb_delete_string(Ndx,_),6,change_cues), % delete old wt
   Indx is Ndx - 1,                 % insert new cue weight in wt box
   ifthenelse(Wt>0,
          send_control_msg(lb_insert_string(Wt,Indx),6,change_cues),
          send_control_msg(lb_insert_string(0,Indx),6,change_cues)),
   send_control_msg(update,6,change_cues),      % update control
   send_dialog_msg(get_msg_cd,next_ctrl(15,5),change_cues). % reroute to box

 get_msg_cd(char(0,80),change_cues) :-    % down arrow
  [!
   send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
   Newndx is Ndx + 1,                                % add 1 to index
   send_control_msg(lb_set_index(_,Newndx),6,change_cues), % reset index
   send_control_msg(lb_set_index(_,Newndx),7,change_cues), % reset index
   send_control_msg(update,6,change_cues),      % update controls
   send_control_msg(update,7,change_cues)
  !],
   fail.

 get_msg_cd(char(0,72),change_cues) :-    % up arrow
  [!
   send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
   ifthenelse((Ndx - 1) < 1,              % if (index - 1) < 1
              Newndx is 1,        % set index to 1
              Newndx is Ndx - 1), % else subtract 1 from index
   send_control_msg(lb_set_index(_,Newndx),6,change_cues), % reset index
   send_control_msg(lb_set_index(_,Newndx),7,change_cues), % reset index
   send_control_msg(update,6,change_cues),      % update controls
   send_control_msg(update,7,change_cues)
  !],
   fail.

 get_msg_cd(char(0,81),change_cues) :-    % page down
  [!
   send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
   send_control_msg(lb_get_count(Count),5,change_cues),      % get # in box
   ifthenelse((Ndx + 12) > Count, % if next page > # in box
              Newndx is Count,          % set index to end
              Newndx is Ndx + 12),      % else set index to next page
   send_control_msg(lb_set_index(_,Newndx),6,change_cues), % reset index
   send_control_msg(lb_set_index(_,Newndx),7,change_cues), % reset index
   send_control_msg(update,6,change_cues),           % update controls
   send_control_msg(update,7,change_cues)
  !],
   fail.

 get_msg_cd(char(0,73),change_cues) :-    % page up
  [!
```

```
   send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
   ifthenelse((Ndx - 12) < 1,          % if previous page < beginning
             Newndx is 1,        % set index to beginning
             Newndx is Ndx - 12),      % else set index to prev page
   send_control_msg(lb_set_index(_,Newndx),6,change_cues), % reset index
   send_control_msg(lb_set_index(_,Newndx),7,change_cues), % reset index
   send_control_msg(update,6,change_cues),      % update controls
   send_control_msg(update,7,change_cues)
  !],
   fail.

get_msg_cd(command(_,add_cue),change_cues) :-  % Add Cue pushbutton
   send_dialog_msg(get_msg_cd,next_ctrl(4,8),change_cues). % reroute - cue fld


get_msg_cd(command(_,change_wt),change_cues) :-      % Change Wt pushbutton
   send_dialog_msg(get_msg_cd,next_ctrl(12,15),change_cues). % reroute- wt fld

get_msg_cd(command(_,change_descr),change_cues) :-  % Change Descr pushbutton
   send_dialog_msg(get_msg_cd,next_ctrl(13,10),change_cues). % reroute-descrip

get_msg_cd(command(_,delete_cue),change_cues) :-      % Delete Cue button
   get_diag_from_e(Diag),          % get diag name
   get_cue_text(Cue,Ndx),          % get cue name/index
   retract(diagnose(Diag,Cue,_)), % delete old diag/cue rule
   send_control_msg(lb_delete_string(Ndx,_),5,change_cues), % delete from box
   send_control_msg(lb_delete_string(Ndx,_),6,change_cues),
   send_control_msg(lb_delete_string(Ndx,_),7,change_cues),
   send_control_msg(update,5,change_cues),      % update controls
   send_control_msg(update,6,change_cues),
   send_control_msg(update,7,change_cues),
   send_dialog_msg(get_msg_cd,next_ctrl(11,5),change_cues). % reroute - box

get_msg_cd(command(_,exit),change_cues) :-     % Exit pushbutton
   !,
   exit_dbox(change_cues).              % exit dialog box

get_msg_cd(Msg,Key) :-                         % default dialog box functions
   def_dialog_fn(Msg,Key).

add_to_list_d :-            % add previous cues to boxes
   get_diag_from_e(Diag),   % get diag name
   diagnose(Diag,Cue,Wt),   % get cue name/wt to add
   send_control_msg(lb_add_string(Cue),5,change_cues), % add to list boxes
   send_control_msg(lb_add_string(Wt),6,change_cues),
   ifthenelse(cue(Cue,Descrip),    % if descrip exists
      send_control_msg(lb_add_string(Descrip),7,change_cues), % add it
      send_control_msg(lb_add_string(no_description),7,change_cues)), % else
   fail.                        % backtrack for other cues

add_to_list_d.                  % guarantee success

get_diag_from_e(Diag) :-   % get diag name from edit field
   send_control_msg(ef_set_text(Ediag,Ediag),2,change_cues),
   string_term(Ediag,Diag).       % convert to term

get_cue_text(Cue,Ndx) :-   % get cue name and index from list box
   send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues),
   send_control_msg(lb_get_text(Ndx,Cue),5,change_cues).


%      ***********************************************************

%  menu selection - Delete Diagnosis

%      ***********************************************************

delete_diag :-                  % delete diagnosis
  dialog_run(change_cues,get_msg_dd).   % activate dialog box

get_msg_dd(command(nobutton,ok),change_cues) :-      % Enter key pressed
  which_control(Old),           % get current control
  New is Old + 1,          % increment
  send_dialog_msg(get_msg_dd,next_tabstop(Old,New),change_cues). % reroute

get_msg_dd(init_dialog,change_cues) :-  % initialize dialog box
  [!
   send_control_msg(text_set(_,$Diagnosis to delete$),1,change_cues),
   send_control_msg(ef_set_text(_,$$),2,change_cues),       % clear edit fields
   send_control_msg(ef_set_text(_,$$),3,change_cues),
   send_control_msg(ef_set_text(_,$$),8,change_cues),
   send_control_msg(ef_set_text(_,$$),9,change_cues),
   send_control_msg(ef_set_text(_,$$),10,change_cues),
   send_control_msg(ef_set_text(_,$$),15,change_cues),
   send_control_msg(lb_clear,5,change_cues),         % clear list boxes
   send_control_msg(lb_clear,6,change_cues),
   send_control_msg(lb_clear,7,change_cues),
   send_control_msg(lb_insert_string($$,0),5,change_cues), % insert null strg
   send_control_msg(lb_insert_string($$,0),6,change_cues),
   send_control_msg(lb_insert_string($$,0),7,change_cues)
  !],
   fail.

get_msg_dd(next_ctrl(2,15),change_cues) :-     % diag name field - backward
  send_dialog_msg(get_msg_dd,next_ctrl(2,14),change_cues). % reroute - Exit

get_msg_dd(next_ctrl(2,3),change_cues) :-      % diag field - forward
  send_control_msg(ef_set_text(Ediag,$$),2,change_cues), % get diag name
  send_control_msg(update,2,change_cues),
  string_term(Ediag,Diag),                     % convert to term
  retract_diag(Diag),                          % delete all diag rules
  send_dialog_msg(get_msg_dd,next_ctrl(2,14),change_cues). % reroute to Exit

get_msg_dd(next_ctrl(14,13),change_cues) :-    % Exit button - backward
  send_dialog_msg(get_msg_dd,next_ctrl(14,2),change_cues). % reroute - diag

get_msg_dd(next_ctrl(14,15),change_cues) :-    % Exit button - forward
  send_dialog_msg(get_msg_dd,next_ctrl(14,2),change_cues). % reroute - diag

get_msg_dd(command(_,exit),change_cues) :-     % Exit puhsbutton
  !,
  exit_dbox(change_cues).               % exit dialog box

get_msg_dd(Msg,Key) :-                         % default dialog box functions
  def_dialog_fn(Msg,Key).

retract_diag(Diag) :-                          % retract all diagnosis rules
  retract(diag_descr(Diag,_)),                 % retract diag discription
  retract(diagnose(Diag,_,_)),                 %  and all cues
  fail.                                        % backtrack for others

retract_diag(Diag).                     % guarantee success


%     ****************************************************************

%  menu selection - Add High-level Cue

%     ****************************************************************

add_hi_cue :-                                  % add new hi_cue
  dialog_run(change_cues,get_msg_ac).          % activate dialog box

get_msg_ac(command(nobutton,ok),change_cues) :-      % Enter key pressed
  which_control(Old),                          % get current control #
  New is Old + 1,                       % increment
  send_dialog_msg(get_msg_ac,next_tabstop(Old,New),change_cues). % reroute

get_msg_ac(init_dialog,change_cues) :-         % initialize dialog box
  [!
   send_control_msg(text_set(_,$High level cue to add$),1,change_cues),
   send_control_msg(ef_set_text(_,$$),2,change_cues),       % clear edit fields
   send_control_msg(ef_set_text(_,$$),3,change_cues),
   send_control_msg(ef_set_text(_,$$),8,change_cues),
    send_control_msg(ef_set_text(_,$$),9,change_cues),
    send_control_msg(ef_set_text(_,$$),10,change_cues),
    send_control_msg(ef_set_text(_,$$),15,change_cues),
    send_control_msg(lb_clear,5,change_cues),          % clear list boxes
    send_control_msg(lb_clear,6,change_cues),
    send_control_msg(lb_clear,7,change_cues),
    send_control_msg(lb_insert_string($$,0),5,change_cues), % insert null strg
    send_control_msg(lb_insert_string($$,0),6,change_cues),
    send_control_msg(lb_insert_string($$,0),7,change_cues)
   !],
    fail.

 get_msg_ac(next_ctrl(2,15),change_cues) :-     % hi_cue name - backward
   send_dialog_msg(get_msg_ac,next_ctrl(2,14),change_cues). % reroute - Exit

 get_msg_ac(next_ctrl(2,_),change_cues) :-      % leaving hi_cue name field
  [!
   get_diag_from_e(Hi_cue),                     % get hi_cue name
   ifthen(cue(Hi_cue,Descrip),                  % display description
         (send_control_msg(ef_set_text(_,Descrip),3,change_cues),
          send_control_msg(update,3,change_cues)))
  !],
   fail.

 get_msg_ac(next_ctrl(3,4),change_cues) :-      % hi_cue description field
  [!
   get_diag_from_e(Hi_cue),                     % get hi_cue name
   send_control_msg(ef_set_text(Descrip,Descrip),3,change_cues),    % get new des
   ifthen(cue(Hi_cue,_),                        % if descrip already present
          retract(cue(Hi_cue,_))),              % delete it
   assertz(cue(Hi_cue,Descrip))                 % add new description
  !],
   fail.


 get_msg_ac(next_ctrl(8,5),change_cues) :-      % cue name field - backward
   send_dialog_msg(get_msg_ac,next_ctrl(8,4),change_cues). % reroute to Add

 get_msg_ac(next_ctrl(9,10),change_cues) :-% cue wt field - forward
  [!
   send_control_msg(ef_set_text(Ecue,$$),8,change_cues), % get cue name
   string_term(Ecue,Cue),                       % convert to term
   send_control_msg(ef_set_text(Ewt,$$),9,change_cues),       % get cue wt
   string_term(Ewt,Wt),                                % convert to term
   send_control_msg(lb_add_string(Cue),5,change_cues),        % add cue to list box
   send_control_msg(lb_add_string(Wt),6,change_cues), % add wt to list box
   get_diag_from_e(Hi_cue),                            % get hi_cue name
   ifthenelse(Wt > 0,                           % if cue wt > 0
              assertz(cue(Hi_cue,Cue,Wt)),      % record hi_cue/cue rule
              assertz(cue(Hi_cue,Cue,0))),      % else record edited rule
   ifthenelse(cue(Cue,Descrip),                 % display descrip
         send_control_msg(lb_add_string(Descrip),7,change_cues),
         (send_control_msg(lb_add_string(no_description),7,change_cues),
          assertz(new_cue))),      % record reminder flag check cues
   send_control_msg(lb_get_count(Count),5,change_cues),       % get # in list
   send_control_msg(lb_set_index(_,Count),5,change_cues), % set index to end
   send_control_msg(lb_set_index(_,Count),6,change_cues),
   send_control_msg(lb_set_index(_,Count),7,change_cues),
   send_control_msg(update,5,change_cues),             % update controls
   send_control_msg(update,6,change_cues),
   send_control_msg(update,7,change_cues),
   send_control_msg(update,9,change_cues)
  !],
   fail.

get_msg_ac(next_ctrl(10,11),change_cues) :-    % cue descrip - forward
  send_control_msg(ef_get_length(L),10,change_cues), % get input lngth
  ifthen(L>0,                            % if new descrip entered
   (send_control_msg(ef_set_text(Descrip,$$),10,change_cues), % get descrip
    get_cue_text(Cue,Ndx),               % get hi_cue name
    send_control_msg(lb_delete_string(Ndx,_),7,change_cues), % delete old
    Indx is Ndx - 1,                     % add new descrip
    send_control_msg(lb_insert_string(Descrip,Indx),7,change_cues),
    send_control_msg(update,7,change_cues),    % update controls
    send_control_msg(update,10,change_cues),
    ifthen(cue(Cue,_),                       % if descrip exists
           retract(cue(Cue,_))),  % delete it
    assertz(cue(Cue,Descrip)))),  % add descrip
  send_dialog_msg(get_msg_ac,next_ctrl(10,4),change_cues). % reroute - Add

get_msg_ac(next_ctrl(4,5),change_cues) :-      % Add Cue - forward
  send_dialog_msg(get_msg_ac,next_ctrl(4,14),change_cues). % reroute - Exit

get_msg_ac(next_ctrl(14,13),change_cues) :-    % Exit - backward
  send_dialog_msg(get_msg_ac,next_ctrl(14,4),change_cues). % reroute - Add

get_msg_ac(next_ctrl(14,15),change_cues) :-     % Exit - forward
  send_dialog_msg(get_msg_ac,next_ctrl(14,2),change_cues). % reroute - hi_cue

get_msg_ac(command(_,add_cue),change_cues) :-  % Add Cue pushbutton
  send_dialog_msg(get_msg_ac,next_ctrl(4,8),change_cues). % reroute - cue nam

get_msg_ac(command(_,exit),change_cues) :-     % Exit
  !,
  exit_dbox(change_cues).               % exit dialog box

get_msg_ac(Msg,Key) :-                         % default dialog box functions
  def_dialog_fn(Msg,Key).


%      ***************************************************************

%  menu selection - Change High-level Cue

%      ***************************************************************

change_hi_cue :-                        % change hi_cue cues
  dialog_run(change_cues,get_msg_cc).          % activate dialog box

get_msg_cc(command(nobutton,ok),change_cues) :-      % Enter key pressed
  which_control(Old),                          % get current control
  ifthenelse(Old == 15,                        % if last one
             New is 1,                         % set to first
             New is Old + 1),                  % else increment
  send_dialog_msg(get_msg_cc,next_tabstop(Old,New),change_cues). % reroute

get_msg_cc(init_dialog,change_cues) :-         % initialize dialog box
 [!
  send_control_msg(text_set(_,$High level cue to change$),1,change_cues),
  send_control_msg(ef_set_text(_,$$),2,change_cues), % clear edit fields
  send_control_msg(ef_set_text(_,$$),3,change_cues),
  send_control_msg(ef_set_text(_,$$),8,change_cues),
  send_control_msg(ef_set_text(_,$$),9,change_cues),
  send_control_msg(ef_set_text(_,$$),10,change_cues),
  send_control_msg(ef_set_text(_,$$),15,change_cues),
  send_control_msg(lb_clear,5,change_cues),           % clear list boxes
  send_control_msg(lb_clear,6,change_cues),
  send_control_msg(lb_clear,7,change_cues),
  send_control_msg(lb_insert_string($$,0),5,change_cues), % insert null strg
  send_control_msg(lb_insert_string($$,0),6,change_cues),
  send_control_msg(lb_insert_string($$,0),7,change_cues)
 !],
  fail.

get_msg_cc(next_ctrl(2,15),change_cues) :-     % hi_cue name - backward
  send_dialog_msg(get_msg_cc,next_ctrl(2,14),change_cues). % reroute-Exit

get_msg_cc(next_ctrl(2,_),change_cues) :-      % leaving hi_cue name
 [!
  get_diag_from_e(Hi_cue),                     % get hi_cue name
  ifthen(cue(Hi_cue,Descrip),                  % display description
         (send_control_msg(ef_set_text(_,Descrip),3,change_cues),
          send_control_msg(update,3,change_cues))),
  send_control_msg(lb_clear,5,change_cues),    % clear boxes
  send_control_msg(lb_clear,6,change_cues),
  send_control_msg(lb_clear,7,change_cues),
  add_to_list_c,                          % add all cues to boxes
  send_control_msg(update,5,change_cues),      % update controls
  send_control_msg(update,6,change_cues),
  send_control_msg(update,7,change_cues)
 !],
  fail.

get_msg_cc(next_ctrl(3,4),change_cues) :-      % hi_cue descrip - forward
 [!
  get_diag_from_e(Hi_cue),                     % get hi_cue name
  send_control_msg(ef_set_text(Descrip,Descrip),3,change_cues),
  ifthen(cue(Hi_cue,_),                        % if descrip exists
        retract(cue(Hi_cue,_))),               % delete it
  assertz(cue(Hi_cue,Descrip))                 % add new descrip
 !],
  fail.


get_msg_cc(next_ctrl(5,8),change_cues) :-      % cue name box - forward
  send_dialog_msg(get_msg_cc,next_ctrl(5,11),change_cues). % reroute - Delete

get_msg_cc(next_ctrl(9,10),change_cues) :- % cue wt field - forward
 [!
  send_control_msg(ef_set_text(Ecue,$$),8,change_cues), % get new cue name
  string_term(Ecue,Cue),                       % convert to term
  send_control_msg(ef_set_text(Ewt,$$),9,change_cues),  % get new cue wt
  string_term(Ewt,Wt),                                 % convert to term
  send_control_msg(lb_add_string(Cue),5,change_cues),       % add cue to list box
  send_control_msg(lb_add_string(Wt),6,change_cues),  % add wt to list box
  get_diag_from_e(Hi_cue),                     % get hi_cue name
  ifthenelse(Wt>0,         % if cue wt > 0
             assertz(cue(Hi_cue,Cue,Wt)),      % record new hi_cue/cue rule
             assertz(cue(Hi_cue,Cue,0))),      % else record edited rule
  ifthenelse(cue(Cue,Descrip),                 % display cue description
        send_control_msg(lb_add_string(Descrip),7,change_cues),
        (send_control_msg(lb_add_string(no_description),7,change_cues),
        assertz(new_cue))),      % set reminder flag to check cue definitions
  send_control_msg(lb_get_count(Count),5,change_cues),      % get # in list
  send_control_msg(lb_set_index(_,Count),5,change_cues), % set index to end
  send_control_msg(lb_set_index(_,Count),6,change_cues),
  send_control_msg(lb_set_index(_,Count),7,change_cues),
  send_control_msg(update,5,change_cues),              % update controls
  send_control_msg(update,6,change_cues),
  send_control_msg(update,7,change_cues),
  send_control_msg(update,8,change_cues),
  send_control_msg(update,9,change_cues)
 !],
  fail.

get_msg_cc(next_ctrl(10,11),change_cues) :-    % cue descrip field - fwd
  send_control_msg(ef_get_length(L),10,change_cues), % lgth of input
  ifthen(L>0,                                  % if new descrip entered
   (send_control_msg(ef_set_text(Descrip,$$),10,change_cues), % get descrip
    get_cue_text(Cue,Ndx),                     % get cue name/index
    send_control_msg(lb_delete_string(Ndx,_),7,change_cues), % delete old
    Indx is Ndx - 1,                           % insert new descrip in box
    send_control_msg(lb_insert_string(Descrip,Indx),7,change_cues),
    send_control_msg(update,7,change_cues),    % update controls
    send_control_msg(update,10,change_cues),
    ifthen(cue(Cue,_),                         % if descrip exists
          retract(cue(Cue,_))),        % delete it
    assertz(cue(Cue,Descrip)))),       % record new descrip
  send_dialog_msg(get_msg_cc,next_ctrl(10,5),change_cues). % reroute - cues


get_msg_cc(next_ctrl(11,10),change_cues) :-    % Delete Cue - backward
  send_dialog_msg(get_msg_cc,next_ctrl(11,5),change_cues). % reroute-cue box

get_msg_cc(next_ctrl(14,15),change_cues) :-     % Exit - forward
  send_dialog_msg(get_msg_cc,next_ctrl(14,2),change_cues). % reroute-hi_cue
```

```
get_msg_cc(next_ctrl(15,2),change_cues) :-     % cue wt field - backward
  get_diag_from_e(Hi_cue),                     % get hi_cue name
  get_cue_text(Cue,Ndx),              % get cue name/index
  retract(cue(Hi_cue,Cue,_)),                  % delete old hi_cue/cue rule
  send_control_msg(ef_set_text(Ewt,$$),15,change_cues), % get new cue wt
  string_term(Ewt,Wt),                         % convert to term
  ifthenelse(Wt>0,                    % edit wt > 0
             assertz(cue(Hi_cue,Cue,Wt)),      % record new hi_cue/cue rule
             assertz(cue(Hi_cue,Cue,0))),      % else record edited rule
  send_control_msg(lb_delete_string(Ndx,_),6,change_cues), % delete old wt
  Indx is Ndx - 1,                    % display new cue wt
  ifthenelse(Wt>0,
         send_control_msg(lb_insert_string(Wt,Indx),6,change_cues),
         send_control_msg(lb_insert_string(0,Indx),6,change_cues)),
  send_control_msg(update,6,change_cues),      % update control
  send_dialog_msg(get_msg_cc,next_ctrl(15,5),change_cues). % reroute-cue box

get_msg_cc(char(0,80),change_cues) :-   % down arrow
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
  Newndx is Ndx + 1,
  send_control_msg(lb_set_index(_,Newndx),6,change_cues), % set new index
  send_control_msg(lb_set_index(_,Newndx),7,change_cues),
  send_control_msg(update,6,change_cues),            % update controls
  send_control_msg(update,7,change_cues)
 !],
  fail.

get_msg_cc(char(0,72),change_cues) :-   % up arrow
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
  ifthenelse((Ndx - 1) < 1,           % if (index - 1) < 1
             Newndx is 1,        % set index to 1
             Newndx is Ndx - 1), % else decrement
  send_control_msg(lb_set_index(_,Newndx),6,change_cues), % set new index
  send_control_msg(lb_set_index(_,Newndx),7,change_cues),
  send_control_msg(update,6,change_cues),            % update controls
  send_control_msg(update,7,change_cues)
 !],
  fail.

get_msg_cc(char(0,81),change_cues) :-   % page down
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
  send_control_msg(lb_get_count(Count),5,change_cues),      % get # in box
  ifthenelse((Ndx + 12) > Count, % if new index past end
             Newndx is Count,          % set index to end
             Newndx is Ndx + 12),      % else set index to next page
  send_control_msg(lb_set_index(_,Newndx),6,change_cues), % set new index
  send_control_msg(lb_set_index(_,Newndx),7,change_cues),
  send_control_msg(update,6,change_cues),            % update controls
  send_control_msg(update,7,change_cues)
 !],
  fail.

get_msg_cc(char(0,73),change_cues) :-               % page up
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),5,change_cues), % get index
  ifthenelse((Ndx - 12) < 1,                  % if new index < 1
             Newndx is 1,              % set new index to 1
             Newndx is Ndx - 12),             % else set to prev page
  send_control_msg(lb_set_index(_,Newndx),6,change_cues), % set new index
  send_control_msg(lb_set_index(_,Newndx),7,change_cues),
  send_control_msg(update,6,change_cues),     % update controls
  send_control_msg(update,7,change_cues)
 !],
  fail.

get_msg_cc(command(_,add_cue),change_cues) :-  % Add Cue pushbutton
  send_dialog_msg(get_msg_cc,next_ctrl(4,8),change_cues). % reroute-cue field
```

```
get_msg_cc(command(_,change_wt),change_cues) :-              % Change Wt
  send_dialog_msg(get_msg_cc,next_ctrl(12,15),change_cues). % reroute-wt fld

get_msg_cc(command(_,change_descr),change_cues) :-    % Change Descrip
  send_dialog_msg(get_msg_cc,next_ctrl(12,10),change_cues). % reroute-descr

get_msg_cc(command(_,delete_cue),change_cues) :-      % Delete Cue
  get_diag_from_e(Hi_cue),               % get hi_cue name
  get_cue_text(Cue,Ndx),         % get cue name/index
  retract(cue(Hi_cue,Cue,_)),            % delete old hi_cue/cue rule
  send_control_msg(lb_delete_string(Ndx,_),5,change_cues), % delete from box
  send_control_msg(lb_delete_string(Ndx,_),6,change_cues),
  send_control_msg(lb_delete_string(Ndx,_),7,change_cues),
  send_control_msg(update,5,change_cues),      % update controls
  send_control_msg(update,6,change_cues),
  send_control_msg(update,7,change_cues),
  send_dialog_msg(get_msg_cc,next_ctrl(11,5),change_cues). % reroute-cue box

get_msg_cc(command(_,exit),change_cues) :-            % Exit
  !,
  exit_dbox(change_cues).                         % exit dialog box

get_msg_cc(Msg,Key) :-                            % default dialog box functions
  def_dialog_fn(Msg,Key).


add_to_list_c :-                   % add previous cues to list boxes
  get_diag_from_e(Hi_cue),               % get hi_cue name
  cue(Hi_cue,Cue,Wt),                    % get cue name/wt
  send_control_msg(lb_add_string(Cue),5,change_cues),       % add to list box
  send_control_msg(lb_add_string(Wt),6,change_cues),
  ifthenelse(cue(Cue,Descrip),           % display cue description
      send_control_msg(lb_add_string(Descrip),7,change_cues),
      send_control_msg(lb_add_string(no_description),7,change_cues)),
  fail.

add_to_list_c.                           % guarantee success


%      ****************************************************************

%  menu selection - Delete High-level Cue

%      ****************************************************************

delete_hi_cue :-                   % delete hi_cue
  dialog_run(change_cues,get_msg_dc).   % activate dialog box

get_msg_dc(command(nobutton,ok),change_cues) :-      % Enter key pressed
  which_control(Old),                    % get current control
  New is Old + 1,                % increment
  send_dialog_msg(get_msg_dc,next_tabstop(Old,New),change_cues). % reroute

get_msg_dc(init_dialog,change_cues) :-  % initialize dialog box
  [!
   send_control_msg(text_set(_,$High level cue to delete$),1,change_cues),
   send_control_msg(ef_set_text(_,$$),2,change_cues),       % clear edit fields
   send_control_msg(ef_set_text(_,$$),3,change_cues),
   send_control_msg(ef_set_text(_,$$),8,change_cues),
   send_control_msg(ef_set_text(_,$$),9,change_cues),
   send_control_msg(ef_set_text(_,$$),10,change_cues),
   send_control_msg(ef_set_text(_,$$),15,change_cues),
   send_control_msg(lb_clear,5,change_cues),          % clear list boxes
   send_control_msg(lb_clear,6,change_cues),
   send_control_msg(lb_clear,7,change_cues),
   send_control_msg(lb_insert_string($$,0),5,change_cues), % insert null strg
   send_control_msg(lb_insert_string($$,0),6,change_cues),
   send_control_msg(lb_insert_string($$,0),7,change_cues)
  !],
   fail.

get_msg_dc(next_ctrl(2,15),change_cues) :-     % hi_cue name - backward
```

```
  send_dialog_msg(get_msg_dc,next_ctrl(2,14),change_cues). % reroute - Exit

get_msg_dc(next_ctrl(2,3),change_cues) :-      % hi_cue name - forward
  send_control_msg(ef_set_text(Ehi_cue,SS),2,change_cues), % get hi_cue name
  send_control_msg(update,2,change_cues),      % update control
  string_term(Ehi_cue,Hi_cue),                 % convert to term
  retract_hi_cue(Hi_cue),              % delete all hi_cue rules
  send_dialog_msg(get_msg_dc,next_ctrl(2,14),change_cues). % reroute - Exit

get_msg_dc(next_ctrl(14,12),change_cues) :-    % Exit - backward
  send_dialog_msg(get_msg_dc,next_ctrl(14,2),change_cues). % reroute - hi_cue

get_msg_dc(next_ctrl(14,15),change_cues) :-    % Exit - forward
  send_dialog_msg(get_msg_dc,next_ctrl(14,2),change_cues). % reroute - hi_cue

get_msg_dc(command(_,exit),change_cues) :-     % Exit pushbutton
  !,
  exit_dbox(change_cues).              % exit dialog box

get_msg_dc(Msg,Key) :-                         % default dialog box functions
  def_dialog_fn(Msg,Key).

retract_hi_cue(Hi_cue) :-              % delete all hi_cue rules
  retract(cue(Hi_cue,_)),              % delete hi_cue description
  retract(cue(Hi_cue,_,_)),                    % delete hi_cue/cue rule
  fail.                                        % backtrack for more rules

retract_hi_cue(Hi_cue).                        % guarantee success


%     ***************************************************************

%  menu selection - Check Cue Definitions

%     ***************************************************************

check_cue :-
  abolish(new_cue/0),                          % delete reminder flag
  dialog_run(definition,get_msg_def).          % activate dialog box

get_msg_def(command(nobutton,ok),definition) :-      % Enter key pressed
  which_control(Old),                          % get current control #
  New is Old + 1,                      % increment
  send_dialog_msg(get_msg_def,next_tabstop(Old,New),definition). % reroute

get_msg_def(char(0,80),definition) :-                % down arrow
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),4,definition),     % get cue index
  Newndx is Ndx + 1,
  send_control_msg(lb_set_index(_,Newndx),5,definition), % reset index
  send_control_msg(update,5,definition)              % update control
 !],
  fail.

get_msg_def(char(0,72),definition) :-                % up arrow
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),4,definition), % get index
  ifthenelse((Ndx - 1) < 1,                    % if (index - 1) < 1
            Newndx is 1,               % set index to 1
            Newndx is Ndx - 1),        % else decrement
  send_control_msg(lb_set_index(_,Newndx),5,definition), % reset index
  send_control_msg(update,5,definition)        % update control
 !],
  fail.

get_msg_def(char(0,81),definition) :-                % page down
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),4,definition), % get index
  send_control_msg(lb_get_count(Count),4,definition),      % get # in box
  ifthenelse((Ndx + 12) > Count,       % if (index + 12) past end
            Newndx is Count,                   % set to end
            Newndx is Ndx + 12),               % else set to next page
  send_control_msg(lb_set_index(_,Newndx),5,definition), % reset index
  send_control_msg(update,5,definition)        % update control
 !],
  fail.

get_msg_def(char(0,73),definition) :-               % page up
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),4,definition), % get index
  ifthenelse((Ndx - 12) < 1,                   % if (index - 12) < 1
             Newndx is 1,              % set index to 1
             Newndx is Ndx - 12),              % else set to prev page
  send_control_msg(lb_set_index(_,Newndx),5,definition), % reset index
  send_control_msg(update,5,definition)        % update control
 !],
  fail.

get_msg_def(init_dialog,definition) :-         % initialize dialog box
  [!
   send_control_msg(lb_clear,4,definition),    % clear list boxes
   send_control_msg(lb_clear,5,definition),
   send_control_msg(lb_insert_string($$,0),4,definition), % insert null strg
   send_control_msg(lb_insert_string($$,0),5,definition),
   add_cue_def_d,                   % add undefined cues in diagnoses
   add_cue_def_c,                   % add undefined cues in hi_cues
   send_control_msg(update,4,definition),      % update controls
   send_control_msg(update,5,definition)
  !],
   fail.

get_msg_def(command(_,exit),definition) :-     % Exit
  adjust_input,                                % add cues to input record
  !,
  exit_dbox(definition).              % exit dialog box


get_msg_def(Msg,Key) :-                        % default dialog box functions
  def_dialog_fn(Msg,Key).

add_cue_def_d :-           % add undefined cues used in diag to list box
  diagnose(_,Cue,Wt),                 % get cue used in building diags
  ifthen(                    % not in input or hi_cue
       (not(defined_in_input(Cue)),not(cue(Cue,_,_))),
       (                     % add cue to list box
        send_control_msg(lb_add_string(Cue),4,definition),
        ifthenelse(cue(Cue,Descrip),   % display description
          send_control_msg(lb_add_string(Descrip),5,definition),
          send_control_msg(lb_add_string(no_description),5,definition))
       )),
  fail.

add_cue_def_d.                        % guarantee success

add_cue_def_c :-           % add undefined cues used in hi_cue to list
  cue(_,Cue,Wt),             % get cue used in building hi_cue
  ifthen(                    % not in input or hi_cue
       (not(defined_in_input(Cue)),not(cue(Cue,_,_))),
       (                     % add cue to list box
        send_control_msg(lb_add_string(Cue),4,definition),
        ifthenelse(cue(Cue,Descrip),   % display description
          send_control_msg(lb_add_string(Descrip),5,definition),
          send_control_msg(lb_add_string(no_description),5,definition))
       )),
  fail.

add_cue_def_c.                        % guarantee success

defined_in_input(Cue) :-         % determine if cue defined in input
  clause(assert_input(Pos),Body),       % find if cue defined in input rec
  arg(1,Body,Term),              % Term is cue(X)
  arg(1,Term,Input_cue),         % Input_cue is X
  Input_cue == Cue.              % is Cue defined in input rec?
```

```
adjust_input :-                             % add checked cues to input record
  send_control_msg(lb_get_choices(_,Bcue),4,definition), % get choices
  string_term(Bcue,Cue),           % convert to term
  last_input_position(Pos),              % get last input position used
  Nextpos is Pos + 1,                    % add 1 to last position
  retract(last_input_position(Pos)),     % delete old last position
  assertz(last_input_position(Nextpos)), % record new last position
  assertz((assert_input(Nextpos) :- assertz(cue(Cue)))), % define cue-input
  fail.

adjust_input.                              % guarantee success


%      ***************************************************************

%  menu selection - Select Client (also generates diagnoses/hi_cues)

%      ***************************************************************

select_client :-
  abolish(diag_act/2),                   % delete any old actual diags
  abolish(diag_pos/2),                   % delete any old possible diags
  abolish(diag_pot/2),                   % delete any old potential diags
  abolish(cue/1),                 % delete any old cues
  abolish(cue_gen/1),                    % delete any old cues generated
  abolish(cnumber/1),                    % delete any old client numbers
  abolish(cname/1),               % delete andy old client names
  dialog_run(cselect,get_msg_sel),       % activate dialog box (cselect)
  cnumber(Cnumber),               % get selected client number
  concat($'$,Cnumber,A),          % convert to term (manually)
  concat(A,$'$,B),
  string_term(B,Cnum),                   % convert to string
  current_window(_,cnumber),
  cls,
  display(Cnum),                  % display client number string
  cname(Cname),                          % get selected client name
  concat($'$,Cname,C),                   % convert to term (manually)
  concat(C,$'$,D),
  string_term(D,Cnam),                   % convert to string
  current_window(_,cname),
  cls,
  display(Cnam),                  % display client name string
  current_window(_,foreground),
  produce_diag.                          % produce diagnoses


get_msg_sel(command(nobutton,ok),cselect) :-   % Enter key pressed
  which_control(Old),                          % get current control #
  New is Old + 1,                       % increment
  send_dialog_msg(get_msg_sel,next_tabstop(Old,New),cselect). % reroute

get_msg_sel(char(0,80),cselect) :-             % down arrow
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),2,cselect), % get index
  Newndx is Ndx + 1,                           % increment index
  send_control_msg(lb_set_index(_,Newndx),3,cselect), % reset index
  send_control_msg(update,3,cselect)           % update control
 !],
  fail.

get_msg_sel(char(0,72),cselect) :-             % up arrow
 [!
  send_control_msg(lb_set_index(Ndx,Ndx),2,cselect), % get index
  ifthenelse((Ndx - 1) < 1,                    % if (index - 1 ) < 1
             Newndx is 1,               % set index to 1
             Newndx is Ndx - 1),        % else decrement
  send_control_msg(lb_set_index(_,Newndx),3,cselect), % reset index
  send_control_msg(update,3,cselect)           % update control
 !],
  fail.

get_msg_sel(char(0,81),cselect) :-             % page down
```

```
  [!
   send_control_msg(lb_set_index(Ndx,Ndx),2,cselect), % get index
   send_control_msg(lb_get_count(Count),2,cselect), % get # in box
   ifthenelse((Ndx + 12) > Count,        % if (index + 12) past end
              Newndx is Count,                 % set index to end
              Newndx is Ndx + 12),             % else set index to next page
   send_control_msg(lb_set_index(_,Newndx),3,cselect), % reset index
   send_control_msg(update,3,cselect)          % update control
  !],
   fail.

get_msg_sel(char(0,73),cselect) :-             % page up
  [!
   send_control_msg(lb_set_index(Ndx,Ndx),2,cselect), % get index
   ifthenelse((Ndx - 12) < 1,                  % if (index - 12) < 1
              Newndx is 1,             % set index to 1
              Newndx is Ndx - 12),             % else set to prev page
   send_control_msg(lb_set_index(_,Newndx),3,cselect), % reset index
   send_control_msg(update,3,cselect)          % update control
  !],
   fail.

get_msg_sel(init_dialog,cselect) :-            % initialize dialog box
   [!
    send_control_msg(lb_clear,2,cselect),      % clear list boxes
    send_control_msg(lb_clear,3,cselect),
    add_clients                        % read client db/add to list boxes
   !],
    fail.

get_msg_sel(command(_,exit),cselect) :-        % Exit (Continue)
   send_control_msg(lb_set_index(Ndx,Ndx),2,cselect), % get index of client
   send_control_msg(lb_get_text(Ndx,Cnumber),2,cselect), % get client number
   send_control_msg(lb_get_text(Ndx,Cname),3,cselect),       % get client name
   assertz(cnumber(Cnumber)),                  % record client number
   assertz(cname(Cname)),              % record client name
   last_input_position(Pos),                   % get length of cue string
   open(Cdb,$cdb$,r),                          % open client data base
   Reclen is Pos + 34,                 % add length of client number,name,CR
   Offset is 0,
   seek(Cdb,Offset,eof,Eof),                   % get EOF
   get_client_rec(Cdb,Reclen,Offset,Eof,Cnumber), % process client record
   close(Cdb),                                 % close file
   !,
   exit_dbox(cselect).                         % exit dialog box

get_msg_sel(Msg,Key) :-                        % default dialog box functions
   def_dialog_fn(Msg,Key).

add_clients :-                                 % add clients to list boxes
   last_input_position(Pos),                   % get length of cue string
   open(Cdb,$cdb$,r),                          % open client data base
   Reclen is Pos + 34,                 % add length of client number,name,CR
   Offset is 0,
   seek(Cdb,Offset,eof,Eof),                   % get EOF
   add_next_client(Cdb,Reclen,Offset,Eof),     % add this client to boxes
   close(Cdb).                                 % close file

add_next_client(Cdb,Reclen,Offset,Eof) :-      % add 1 client to list boxes
   seek(Cdb,Offset,bof,_),             % seek beginning of client
   read_string(Cdb,Reclen,Clientrec),          % read Client database record
   substring(Clientrec,0,11,Cnumber),          % client number
   substring(Clientrec,11,20,Cname),           % client name
   send_control_msg(lb_add_string(Cnumber),2,cselect),       % add number to list
   send_control_msg(lb_add_string(Cname),3,cselect),   % add name to list
   Newoffset is Offset + Reclen,               % offset for next client
   ifthen(Newoffset < Eof,  % if not(EOF), recurse for another client
          add_next_client(Cdb,Reclen,Newoffset,Eof)).

get_client_rec(Cdb,Reclen,Offset,Eof,Cnum) :-  % get client record
   seek(Cdb,Offset,bof,_),             % seek beginning of client
   read_string(Cdb,Reclen,Clientrec),          % read client database record
  substring(Clientrec,0,11,Cnumber),          % client number
  ifthenelse(Cnum = Cnumber,                  % is this right client?
            (Pos is Reclen - 33,              % get start of cue indicators
             substring(Clientrec,31,Pos,Crec),      % slice indicators from recrd
             process_cdb(Crec)),              % process client cues
            (Newoffset is Offset + Reclen,    % else get position of next
             ifthen(Newoffset < Eof,   % if not(EOF), recurse for another
                    get_client_rec(Cdb,Reclen,Newoffset,Eof,Cnum)))).

process_cdb(Crec) :-                          % process client cue string
  string_search($1$,Crec,Cuepos),             % get position of indicator
  assert_input(Cuepos),                       % record cue
  fail.                                       % backtrack for more

process_cdb(Crec).                     % guarantee success

produce_diag :-
  develop_cues,                               % develop higher level cues
  develop_cues,                               % try again for deeper levels
  produce_act_diag,                    % produce actual/possible diag
  produce_pot_diag.                    % produce potential diags

produce_act_diag :-                    % produce actual/possible diag
  cue(Cue),                            % get client cue
  string_term(Cue_str,Cue),                   % convert to string
  nth_char(0,Cue_str,99),              % first char = c?
  diagnose(Diag,Cue,_),                       % find diag to match cue
  not(diag_act(Diag,_)),        % proceed if this diag not
  not(diag_pos(Diag,_)),        %   already generated, else backtrack
  findall(Wt,(get_cue_wt_act_d(Diag,Wt)),Wtlist),     % list wts for
                                % all cues present in that diag
  sum_wts(Totwt,Wtlist),               % total wts
  ifthenelse(Totwt >= 100,                    % if threshold is met
     assertz(diag_act(Diag,Totwt)),           % flag diag actual
     assertz(diag_pos(Diag,Totwt))),          % else flag diag possible
  fail.

produce_act_diag.                      % guarantee success

produce_pot_diag :-                    % produce potential diagnoses
  cue(Cue),                            % get client cue
  string_term(Cue_str,Cue),                   % convert to string
  nth_char(0,Cue_str,114),                    % first char = r?
  diagnose(Diag,Cue,_),                       % find diag to match cue
  not(diag_act(Diag,_)),        % proceed if act diag not genned
  not(diag_pot(Diag,_)),        % proceed if pot diag not already gen
  findall(Wt,(get_cue_wt_pot_d(Diag,Wt)),Wtlist),     % list wts for
                                % all cues present in that diag
  sum_wts(Totwt,Wtlist),               % total wts
  ifthen(Totwt >= 100,                        % if threshold is met
        assertz(diag_pot(Diag,Totwt))),       % flag diag potential
  ifthen(diag_pos(Diag,_),                    % delete pos diag if present
         retract(diag_pos(Diag,_))),
  fail.                                       % backtrack for more

produce_pot_diag.                      % guarantee success

sum_wts(X,[]) :-           % sum cue weights, stop when list is empty
  X is 0.                              % assign 0 to X
sum_wts(X,[H|T]) :-
  sum_wts(X1,T),                       % recurse for tail of list
  X is X1 + H.                                % X = H + sum(T)

get_cue_wt_act_d(Diag,Wt) :-           % get a diag to match cue present
  cue(Cue),                            % get Cue
  string_term(Cue_str,Cue),                   % convert to string
  nth_char(0,Cue_str,99),              % first char = c?
  diagnose(Diag,Cue,Wt).               % get Diag to match

get_cue_wt_pot_d(Diag,Wt) :-           % get a diag to match cue present
  cue(Cue),                            % get Cue
  string_term(Cue_str,Cue),                   % convert to string
  nth_char(0,Cue_str,114),                    % first char = r?
  diagnose(Diag,Cue,Wt).               % get Diag to match

get_cue_wt_c(Hi_cue,Wt) :-                    % get a Hi_cue to match cue
  cue(Cue),                            % get Cue
  cue(Hi_cue,Cue,Wt).                         % get Hi_cue to match

develop_cues :-                               % develop hi_cues
  cue(Cue),                            % get raw client cue
  cue(Hi_cue,Cue,_),                          % find Hi_cue to match cue
                    % create a higher level cue if possible
  not(cue(Hi_cue)),              % proceed if this Hi_cue not already
                                       % present, else backtrack for
                                       % another Hi_cue
  findall(Wt,(get_cue_wt_c(Hi_cue,Wt)),Wtlist), % list wts for
                                 % all cues present for that Hi-cue
  sum_wts(Totwt,Wtlist),               % total wts
  ifthen(Totwt >= 100,assert_cue(Hi_cue)),     %  record new cue in database
                                       % if threshold of 100 met
  fail.                                       % backtrack for another cue

develop_cues.                                 % guarantee success

assert_cue(Hi_cue) :-                         % record hi_cue
  assertz(cue(Hi_cue)),                       % record new cue
  assertz(cue_gen(Hi_cue)).                   % flag as a generated cue


%      **************************************************************

%  menu selection - Display Actual Diagnoses

%      **************************************************************

display_act :-
  ctr_set(0,1),                        % set counter for screen display
  ifthen(not(diag_act(Diag,_)),               % check for none generated
        (nl,nl,display('no actual diagnoses generated'),fail)),
  diag_act(Diag,Totwt),                       % get actual diag generated
  nl,
  nl,
  ctr_inc(0,Linenbr),                         % increment counter
  ifthen(Linenbr > 5,                         % pause after 5 displayed
        (nl,
         display(more),
         keyb(_,_),
         nl,
         ctr_set(0,1))),
  ifthenelse(diag_descr(Diag,Descr_str),      % if description present
        (concat($'$,Descr_str,A),             % display it
         concat(A,$'$,B),
         string_term(B,Descr),
         display(Descr)),
         display(Diag)),
  display(' is confirmed with a weight of'),
  tab(2),
  display(Totwt),                      % display weight
  fail.                                       % backtrack for another diag

display_act :-                                % end processing
  nl,
  nl,
  display('press any key to return to menu'),
  keyb(_,_).


%      **************************************************************

%  menu selection - Display Possible Diagnoses

%      **************************************************************
```

```
display_pos :-
  ctr_set(0,1),
  ifthen(not(diag_pos(Diag,_)),                 % check for none generated
        (nl,nl,display('no possible diagnoses generated'),fail)),
  diag_pos(Diag,Totwt),                         % get possible diag generated

  nl,
  nl,
  ctr_inc(0,Linenbr),
  ifthen(Linenbr > 5,                           % pause after 5 displayed
        (nl,
         display(more),
         keyb(_,_),
         nl,
         ctr_set(0,1))),
  ifthenelse(diag_descr(Diag,Descr_str),        % if description present
        (concat($'$,Descr_str,A),               % display it
         concat(A,$'$,B),
         string_term(B,Descr),
         display(Descr)),
         display(Diag)),
  display(' is possible, generated with a weight of'),
  tab(2),
  display(Totwt),                        % display weight
  fail.                                         % backtrack for another diag

display_pos :-                                  % end processing
  nl,
  nl,
  display('press any key to return to menu'),
  keyb(_,_).


%     ************************************************************

%  menu selection - Display Potential Diagnoses

%     ************************************************************

display_pot :-
  ctr_set(0,1),
  ifthen(not(diag_pot(Diag,_)),                 % check for none generated
        (nl,nl,display('no potential diagnoses generated'),fail)),
  diag_pot(Diag,Totwt),                         % get potential diag generated

  nl,
  nl,
  ctr_inc(0,Linenbr),
  ifthen(Linenbr > 5,                           % pause after 5 displayed
        (nl,
         display(more),
         keyb(_,_),
         nl,
         ctr_set(0,1))),
  ifthenelse(diag_descr(Diag,Descr_str),        % if description present
        (concat($'$,Descr_str,A),               % display it
         concat(A,$'$,B),
         string_term(B,Descr),
         display(Descr)),
         display(Diag)),
  display(' is potential, generated with a weight of'),
  display(Totwt),                        % display weight

  fail.                                         % backtrack for another diag

display_pot :-                                  % end processing
  nl,
  nl,
  display('press any key to return to menu'),
  keyb(_,_).
```

```
%     *************************************************************

%  menu selection - Explain Actual Diagnoses

%     *************************************************************

explain_act_diag :-
  ctr_set(0,1),
  ifthen(not(diag_act(Diag,_)),               % check for none generated
        (nl,nl,display('no actual diagnoses generated'),fail)),
  diag_act(Diag,_),                     % get diagnosis to explain
  ctr_inc(0,Disp_ctr),
  ifthen(Disp_ctr > 1,                        % pause after each diag
         (nl,
         nl,
         display(more),
         keyb(_,_))),
  nl,
  nl,
  ifthenelse(diag_descr(Diag,Ddescr_str),     % if description present
        (concat($'$,Ddescr_str,A),            % display it
         concat(A,$'$,B),
         string_term(B,Ddescr),
         display(Ddescr)),
         display(Diag)),
  display(' confirmed due to the presence of the following cues:'),
  diagnose(Diag,Cue,Wt),                % get possible cue with Diag
  cue(Cue),                             % is this cue present?
  string_term(Cue_str,Cue),
  nth_char(0,Cue_str,99),               % first char = c?
  nl,
  tab(2),
  ifthenelse(cue(Cue,Cdescr_str),             % if description present
        (concat($'$,Cdescr_str,C),            % display it
         concat(C,$'$,D),
         string_term(D,Cdescr),
         display(Cdescr)),
         display(Cue)),
  fail.                                       % backtrack for more

explain_act_diag :-                     % end processing
  nl,
  nl,
  display('press any key to return to menu'),
  keyb(_,_).


%     *************************************************************

%  menu selection - Explain Possible Diagnoses

%     *************************************************************

explain_pos_diag :-
  ctr_set(0,1),
  ifthen(not(diag_pos(Diag,_)),               % check for none generated
        (nl,nl,display('no possible diagnoses generated'),fail)),
  diag_pos(Diag,_),                     % get diagnosis to explain
  ctr_inc(0,Disp_ctr),
  ifthen(Disp_ctr > 1,                        % pause after each diag
        (nl,
         nl,
         display(more),
         keyb(_,_))),
  nl,
  nl,
  ifthenelse(diag_descr(Diag,Ddescr_str),     % if description present
        (concat($'$,Ddescr_str,A),            % display it
         concat(A,$'$,B),
         string_term(B,Ddescr),
         display(Ddescr)),
         display(Diag)),
   display(' possible due to the presence of the following cues:'),
   diagnose(Diag,Cue,Wt),                   % get possible cue with Diag
   cue(Cue),                                % is this cue present?
   string_term(Cue_str,Cue),
   nth_char(0,Cue_str,99),                  % first char = c?
   nl,
   tab(2),
   ifthenelse(cue(Cue,Cdescr_str),                % if descrip present
         (concat($'$,Cdescr_str,C),               % display it
          concat(C,$'$,D),
          string_term(D,Cdescr),
          display(Cdescr)),
          display(Cue)),
   fail.                                          % backtrack for more

explain_pos_diag :-                       % end processing
   nl,
   nl,
   display('press any key to return to menu'),
   keyb(_,_).


%      ***************************************************************

%  menu selection - Explain Potential Diagnoses

%      ***************************************************************

explain_pot_diag :-
   ctr_set(0,1),
   ifthen(not(diag_pot(Diag,_)),                  % check for none generated
         (nl,nl,display('no potential diagnoses generated'),fail)),
   diag_pot(Diag,_),                      % get diagnosis to explain
   ctr_inc(0,Disp_ctr),
   ifthen(Disp_ctr > 1,                           % pause after each diagnosis
         (nl,
          nl,
          display(more),
          keyb(_,_))),
   nl,
   nl,
   ifthenelse(diag_descr(Diag,Ddescr_str),        % if description present
         (concat($'$,Ddescr_str,A),               % display it
          concat(A,$'$,B),
          string_term(B,Ddescr),
          display(Ddescr)),
          display(Diag)),
   display(' is potential due to the presence of the following cues:'),
   diagnose(Diag,Cue,Wt),                 % get possible cue with Diag
   cue(Cue),                              % is this cue present?
   string_term(Cue_str,Cue),
   nth_char(0,Cue_str,114),                       % first char = c?
   nl,
   tab(2),
   ifthenelse(cue(Cue,Cdescr_str),                % if description present
         (concat($'$,Cdescr_str,C),               % display it
          concat(C,$'$,D),
          string_term(D,Cdescr),
          display(Cdescr)),
          display(Cue)),
   fail.                                          % backtrack for more

explain_pot_diag :-                       % end processing
   nl,
   nl,
   display('press any key to return to menu'),
   keyb(_,_).


%      ***************************************************************

%  menu selection - Explain Generated Cues
%     ****************************************************************

explain_cue :-
  ctr_set(0,1),
  ifthen(not(cue_gen(Hi_cue)),                      % check for none generated
         (nl,nl,display('no cues generated'),fail)),
  cue_gen(Hi_cue),                           % get hi_cue to explain
  ctr_inc(0,Disp_ctr),
  ifthen(Disp_ctr > 1,                              % pause after each
        (nl,
         nl,
         display(more),
         keyb(_,_))),
  nl,
  nl,
  ifthenelse(cue(Hi_cue,Hdescr_str),                % if description present
        (concat($'$,Hdescr_str,A),                  % display it
         concat(A,$'$,B),
         string_term(B,Hdescr),
         display(Hdescr)),
         display(Hi_cue)),
  tab(2),
  display('derived from:'),
  cue(Hi_cue,Cue,_),                                % get possible cue with Hi_cue
  cue(Cue),                                  % is this cue present
  nl,
  tab(2),
  ifthenelse(cue(Cue,Cdescr_str),                   % if description present
        (concat($'$,Cdescr_str,C),                  % display it
         concat(C,$'$,D),
         string_term(D,Cdescr),
         display(Cdescr)),
         display(Cue)),
  fail.                                             % backtrack for more

explain_cue :-                                      % end processing
  nl,
  nl,
  display('press any key to return to menu'),
  keyb(_,_).


%     ****************************************************************

%  menu selection - Print Client Information

%     ****************************************************************

print_client :-
  create(Pclnt,pclnt),                              % create print file
  stdout(pclnt,p_client),                    % redirect output and process
  close(Pclnt),                                     % close file
  shell('print pclnt < prnfile').                   % print file

p_client :-
  display(
  '                  University of North Florida                          '
         ),
  nl,
  display(
  '                        Diagnostician                             '
       ),
  nl,nl,
  display(  'CLIENT INFORMATION '),
  cnumber(Cnumber),                          % get client number
  cname(Cname),                                     % get client name
  concat($' for $,Cnumber,A1),                      % print them
  concat(A1,$  $,B1),
  concat(B1,Cname,C1),
  concat(C1,$'$,D1),
  string_term(D1,Cnam),
  display(Cnam),
  nl,
  nl,
  display('Cues Present'),
  nl,
  cue(Cue),                              % get cue to print
  nl,
  tab(2),
  ifthenelse(cue(Cue,Cdescr_str),               % if description is present
        (concat($'$,Cdescr_str,C),              % print it
         concat(C,$'$,D),
         string_term(D,Cdescr),
         display(Cdescr)),
         display(Cue)),
  cue_gen(Cue),                                 % proceed if this is a hi_cue
  nl,
  tab(2),
  display('derived from:'),
  Offset = 2,
  p_lo_cues(Cue,Offset),                 % print lower level cues
  fail.                                         % backtrack for more

p_client.                                % guarantee success


%     ***************************************************************

%  menu selection - Print Actual Diagnoses

%     ***************************************************************

print_act :-
  create(Pactd,pactd),                          % create print file
  stdout(pactd,p_act),                          % redirect output and process
  close(Pactd),                                 % close file
  shell('print pactd < prnfile').               % print file

p_act :-
  display(
  '                 University of North Florida                        '
        ),
  nl,
  display(
  '                       Diagnostician                        '
       ),
  nl,nl,
  display(  'GENERATED DIAGNOSES - ACTUAL    '),
  cnumber(Cnumber),                      % get client number
  cname(Cname),                                 % get client name
  concat($' for $,Cnumber,A1),                  % print them
  concat(A1,$  $,B1),
  concat(B1,Cname,C1),
  concat(C1,$'$,D1),
  string_term(D1,Cnam),
  display(Cnam),
  nl,
  ifthen(not(diag_act(Diag,_)),                 % check for none generated
        (nl,nl,display('no actual diagnoses generated'),fail)),
  diag_act(Diag,_),                      % get diagnosis to explain
  nl,
  nl,
  display(*********************************),
  nl,
  ifthenelse(diag_descr(Diag,Ddescr_str),       % if description present
        (concat($'$,Ddescr_str,A),              % print it
         concat(A,$'$,B),
         string_term(B,Ddescr),
         display(Ddescr)),
         display(Diag)),
  display(' confirmed due to the presence of the following cues:'),
  diagnose(Diag,Cue,Wt),                 % get possible cue with Diag
  cue(Cue),                              % is this cue present?
  string_term(Cue_str,Cue),
  nth_char(0,Cue_str,99),              % first char = c?
  nl,
  tab(2),
  ifthenelse(cue(Cue,Cdescr_str),             % if description present
        (concat($'$,Cdescr_str,C),            % print it
         concat(C,$'$,D),
         string_term(D,Cdescr),
         display(Cdescr)),
         display(Cue)),
  cue_gen(Cue),                               % proceed if this is a hi_cue
  nl,
  tab(2),
  display('derived from:'),
  Offset = 2,
  p_lo_cues(Cue,Offset),               % print lower level cues
  fail.                                       % backtrack for more

p_act.                                        % guarantee success


%     ***************************************************************

%  menu selection - Print Possible Diagnoses

%     ***************************************************************

print_pos :-
  create(Pposd,pposd),                        % create print file
  stdout(pposd,p_pos),                        % redirect output and process
  close(Pposd),                               % close print file
  shell('print pposd < prnfile').             % print file

p_pos :-
  display(
  '                 University of North Florida                        '
        ),
  nl,
  display(
  '                       Diagnostician                         '
       ),
  nl,nl,
  display('GENERATED DIAGNOSES - POSSIBLE   '),
  cnumber(Cnumber),                    % get client number
  cname(Cname),                               % get client name
  concat($' for $,Cnumber,A1),                % print them
  concat(A1,$  $,B1),
  concat(B1,Cname,C1),
  concat(C1,$'$,D1),
  string_term(D1,Cnam),
  display(Cnam),
  nl,
  ifthenelse(not(diag_pos(Diag,_)),           % if none generated
        (nl,
         nl,
         display('no possible diagnoses generated'),
         fail),
                                       % else
        (diag_pos(Diag,_),                    % get diagnosis to explain
         p_pos1(Diag),                        % print diagnosis
         p_other_cues_pos(Diag),       % print other cues to observe
         fail)).                       % backtrack for more

p_pos.                                        % guarantee success

p_pos1(Diag) :-                               % print possible diagnosis
  nl,
  nl,
  display(**********************************),
  nl,
  ifthenelse(diag_descr(Diag,Ddescr_str),     % if description present
        (concat($'$,Ddescr_str,A),            % print it
         concat(A,$'$,B),
         string_term(B,Ddescr),
         display(Ddescr)),
         display(Diag)),
  display(' possible due to the presence of the following cues:'),
  diagnose(Diag,Cue,Wt),               % get possible cue with Diag
  cue(Cue),                            % is this cue present?
  string_term(Cue_str,Cue),
  nth_char(0,Cue_str,99),              % first char = c?
  nl,
  tab(2),
  ifthenelse(cue(Cue,Cdescr_str),              % if description present
        (concat($'$,Cdescr_str,C),             % print it
         concat(C,$'$,D),
         string_term(D,Cdescr),
         display(Cdescr)),
         display(Cue)),
  cue_gen(Cue),                                % proceed if this is a hi_cue
  nl,
  tab(2),
  display('derived from:'),
  Offset = 2,
  p_lo_cues(Cue,Offset).               % print lower level cues

p_pos1(Diag).                                  % guarantee success


%     ****************************************************************

%  menu selection - Print Potential Diagnoses

%     ****************************************************************

print_pot :-
  create(Ppotd,ppotd),                         % create print file
  stdout(ppotd,p_pot),                         % redirect output and process
  close(Ppotd),                                % close print file
  shell('print ppotd < prnfile').              % print file

p_pot :-
  display(
  '                 University of North Florida                      '
        ),
  nl,
  display(
  '                        Diagnostician                             '
       ),
  nl,nl,
  display('GENERATED DIAGNOSES - POTENTIAL  '),
  cnumber(Cnumber),                    % get client number
  cname(Cname),                                % get client name
  concat($' for $,Cnumber,A1),                 % print them
  concat(A1,$  $,B1),
  concat(B1,Cname,C1),
  concat(C1,$'$,D1),
  string_term(D1,Cnam),
  display(Cnam),
  nl,
  ifthenelse(not(diag_pot(Diag,_)),            % if none generated
            (nl,
             nl,
             display('no potential diagnoses generated'),
             fail),
                                       % else
            (diag_pot(Diag,_),                 % get diagnosis to explain
             p_pot1(Diag),                     % print diagnosis
             p_other_cues_pot(Diag),           % print other cues to observe
             fail)).                           % backtrack for more

p_pot.                                         % guarantee success

p_pot1(Diag) :-                                % print potential diagnosis
  nl,
    nl,
    display(*********************************),
    nl,
    ifthenelse(diag_descr(Diag,Ddescr_str),      % if description present
          (concat($'$,Ddescr_str,A),             % print it
           concat(A,$'$,B),
           string_term(B,Ddescr),
           display(Ddescr)),
           display(Diag)),
    display(' potential due to the presence of the following cues:'),
    diagnose(Diag,Cue,Wt),              % get possible cue with Diag
    cue(Cue),                           % is this cue present?
    string_term(Cue_str,Cue),
    nth_char(0,Cue_str,114),                    % first char = r?
    nl,
    tab(2),
    ifthenelse(cue(Cue,Cdescr_str),             % if description present
          (concat($'$,Cdescr_str,C),            % print it
           concat(C,$'$,D),
           string_term(D,Cdescr),
           display(Cdescr)),
           display(Cue)),
    cue_gen(Cue),                               % proceed if this is a hi_cue
    nl,
    tab(2),
    display('derived from:'),
    Offset = 2,
    p_lo_cues(Cue,Offset).              % print lower level cues

p_pot1(Diag).                                   % guarantee success

p_lo_cues(Cue,Offset) :-                % print lower level cues
    cue_gen(Cue),                               % is this a generated cue?
    cue(Cue,Lo_cue,_),                          % develop lower levels
    cue(Lo_cue),                                % is this cue present?
    nl,
    Newoffset is Offset + 4,
    tab(Newoffset),
    ifthenelse(cue(Lo_cue,Cdescr_str),          % if description present
          (concat($'$,Cdescr_str,C),            % print it
           concat(C,$'$,D),
           string_term(D,Cdescr),
           display(Cdescr)),
           display(Lo_cue)),
    cue_gen(Lo_cue),                    % proceed if this is a hi_cue
    nl,
    tab(Newoffset),
    display('derived from:'),
    p_lo_cues(Lo_cue,Newoffset),                % recurse for lower level cue
    fail.                                       % backtrack for more

p_lo_cues(Cue,Offset).                          % guarantee success

p_other_cues_pos(Diag) :-  % print other cues to observe - possible diag
    diag_pos(Diag,_),                   % get possible diag
    nl,
    nl,
    display('**** Other defining characteristics to observe for '),
    ifthenelse(diag_descr(Diag,Ddescr_str),     % if description present
          (concat($'$,Ddescr_str,A),            % print it
           concat(A,$'$,B),
           string_term(B,Ddescr),
           display(Ddescr)),
           display(Diag)),
    nl,
    diagnose(Diag,Cue,_),                       % get cue in diag
    not(cue(Cue)),                  % ensure this cue not already present
    string_term(Cue_str,Cue),
    nth_char(0,Cue_str,99),             % first char = c?
    nl,
    tab(2),
    ifthenelse(cue(Cue,Cdescr_str),             % if description present
          (concat($'$,Cdescr_str,C),            % print it
           concat(C,$'$,D),
           string_term(D,Cdescr),
           display(Cdescr)),
           display(Cue)),
   fail.                                        % backtrack for more

 p_other_cues_pos(Diag) :-              % end processing
   nl,
   tab(2),
   display('**** End of other characteristics to observe').

 p_other_cues_pot(Diag) :-  % print other cues to observe - potential
   diag_pot(Diag,_),                    % get diag to print
   nl,
   nl,
   display('****  Defining characteristics to observe for '),
   ifthenelse(diag_descr(Diag,Ddescr_str),      % if description present
          (concat($'$,Ddescr_str,A),            % print it
           concat(A,$'$,B),
           string_term(B,Ddescr),
           display(Ddescr)),
           display(Diag)),
   nl,
   diagnose(Diag,Cue,_),                        % get cue in diag
   not(cue(Cue)),                 % ensure this cue not already present
   string_term(Cue_str,Cue),
   nth_char(0,Cue_str,99),              % first char = c?
   nl,
   tab(2),
   ifthenelse(cue(Cue,Cdescr_str),              % if description present
          (concat($'$,Cdescr_str,C),            % print it
           concat(C,$'$,D),
           string_term(D,Cdescr),
           display(Cdescr)),
           display(Cue)),
   fail.                                        % backtrack for more

 p_other_cues_pot(Diag) :-                      % end processing
   nl,
   tab(2),
   display('**** End of characteristics to observe').


%     ****************************************************************

%  menu selection - Print Diagnoses Definition

%     ****************************************************************

print_def :-
  cls,
  nl,
  nl,
  display('Enter diagnosis to print  '),        % get diag to print
  read_string(99,Ediag),
  string_term(Ediag,Diag),
  create(Pdef,pdef),                            % create print file
  stdout(pdef,p_def(Diag)),                     % redirect output and process
  close(Pdef),                                  % close print file
  shell('print pdef < prnfile').        % print file

p_def(Diag) :-
  display(
  '                  University of North Florida                        '
        ),
  nl,
  display(
  '                                     Diagnostician                          '

       ),
  nl,nl,
    display(
    'DIAGNOSIS DEFINITION                                          '
          ),
    nl,
    nl,
    display('Diagnosis  '),
    display(Diag),
    ifthen(diag_descr(Diag,Ddescr_str),
          (concat($' $,Ddescr_str,A),
           concat(A,$'$,B),
           string_term(B,Ddescr),
           display(Ddescr))),
    display(' is defined by the following cues:'),
    ifthen(not(diagnose(Diag,_,_)),              % check for none generated
          (nl,nl,display('diagnoses not defined'),fail)),
    p_diag_cues(Diag).                           % print defining cues

  p_def(Diag).                          % guarantee success

  p_diag_cues(Diag) :-                           % print defining cues
    diagnose(Diag,Cue,Wt),              % get Cue
    nl,
    nl,
    display(Cue),                                % print it
    ifthen(cue(Cue,Cdescr_str),
          (concat($' $,Cdescr_str,C),
           concat(C,$'$,D),
           string_term(D,Cdescr),
           display(Cdescr))),
    nl,
    display('whose weight is : '),
    display(Wt),
    display(' and is defined by: '),
    Offset = 0,
    ifthenelse(cue(Cue,_,_),                     % if hi_cue
               p_lo_diag(Cue,Offset),            % print lower level cues
               display(input)),                  % else cue comes from input
    fail.

  p_diag_cues(Diag,Cue,Wt).             % guarantee success

  p_lo_diag(Hi_cue,Offset) :-                    % print lower level cues
    cue(Hi_cue,Cue,Wt),                          % get cue to print
    nl,
    nl,
    Newoffset is Offset + 4,
    tab(Newoffset),
    display(Cue),
    ifthen(cue(Cue,Cdescr_str),
          (concat($' $,Cdescr_str,A),
           concat(A,$'$,B),
           string_term(B,Cdescr),
           display(Cdescr))),
    nl,
    tab(Newoffset),
    display('whose weight is : '),
    display(Wt),
    display(' and is defined by: '),
    ifthenelse(cue(Cue,_,_),                     % if this is a hi_cue
               p_lo_diag(Cue,Newoffset),  % recurse to print lower level cues
               display(input)).                  % else cue comes from input

  p_lo_diag(Cue,Offset).                         % guarantee success


  %     ***************************************************************

  %  menu selection - Print Input Format

  %     ***************************************************************
```

```
print_input :-
  create(Pinput,pinput),              % create print file
  stdout(pinput,p_input),             % redirect output and print
  close(Pinput),                      % close prnt file
  shell('print pinput < prnfile').          % print file

p_input :-
  display(
  '                University of North Florida                        '
        ),
  nl,
  display(
  '                              Diagnostician                                '

       ),
  nl,nl,
  display(
  'INPUT RECORD FORMAT                                             '
       ),
  nl,
  nl,
  clause(assert_input(Pos),Body),           % get body of input clause
  arg(1,Body,Term),                   % parse body
  arg(1,Term,Input_cue),              % get cue
  nl,
  nl,
  display('Position'),
  tab(2),
  display(Pos),                             % print position
  tab(2),
  display('is reserved for'),
  tab(2),
  display(Input_cue),                       % print cue
  ifthen(cue(Input_cue,Cdescr_str),         % if description present
        (concat($' $,Cdescr_str,A),         % print it
         concat(A,$'$,B),
         string_term(B,Cdescr),
         display(Cdescr))),
  fail.                                     % backtrack for more

p_input.                              % guarantee success

%     ***************************************************************

%  menu selection - Save Redefinitions

%     ***************************************************************

save_db :-                            % save nurse knowledge base
  file_list(nurse,
          [last_input_position/1,           % last position used
          assert_input/1,             % input format
           diagnose/3,                      % diagnosis rules
           diag_descr/2,              % diagnosis descriptions
           cue/3,                     % hi_cue rules
           cue/2]).                         % cue descriptions

%     ***************************************************************

%  menu selection - Quit - Return to DOS

%     ***************************************************************

quit_prolog :- halt.                        % halt prolog
```

VITA

Tom Edgar has a Bachelor of Industrial Engineering from Auburn University, 1976, and expects to receive a Master of Science in Computer and Information Sciences from the University of North Florida in May, 1991. Dr Jack E. Leitner of the University of North Florida is serving as Tom's thesis advisor. Tom is currently self-employed as a systems analyst/programmer and services companies on a contractual basis. He has held that position for eleven years. Prior to that, Tom served for three years as a systems analyst and methods analyst for Blue Cross and Blue Shield of Florida in providing paperless claims submission for doctor's offices and hospitals throughout Florida.