

**KNOWLEDGE DESIGN IN RULE-BASED SYSTEM
ARCHITECTURE FOR C TUTORIAL**

Perpustakaan SKTM

**V THAVACHELVI A/P VELAUTHAM
WGC020017**

**DISSERTATION SUBMITTED IN FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SOFTWARE ENGINEERING**

**FACULTY OF COMPUTER SCIENCE AND
INFORMATION TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR**

FEBRUARY 2005

Abstract

CTutorial4u is a standalone system developed in Visual Basic 6.0 which is able to provide guidance to the students who are pursuing C programming subject in their higher learning institutes. It is designed based on rule-based system architecture in order to provide a flexible knowledge base foundation within C programming. The knowledge base structure is essential to the successful execution of rules underlying the basic modules or processes involved in the system.

CTutorial4u's functionalities divided into two categories that are 'User Module' and 'Admin Module'. User Module has 'Lecture notes', 'Tutorial session' and 'Quiz session'. The Admin Module has 'User management' and 'Question management'. Lecture notes contains lecture notes in power point format, tutorial session contains knowledge based tutorials to teach main topics in C programming interactively and the quiz session provide interactive quiz to the students.

The knowledge design in rule based system architecture actually run based on some standard rules set in the working memory. The rules will remain the same for different runs. CTutorial4u is not only an exercise to the students but actually improves basic understanding in C programming and as a tool to test their knowledge in that field. At the same time, it will judge student's ability and efficiency of answering different set of quizzes with various concepts in programming language. The system is intended to be used by Faculty of Science Computer and Information Technology (FCSIT), University Malaya students where it will be installed FCSIT's laboratories.

Acknowledgements

Firstly, I would like to thank the Almighty GOD for the strength and courage He has given me throughout the completion of this project. This project is dedicated to my beloved parents and siblings who inspired and encouraged me to carry out the project successfully.

I would like take this opportunity to express my utmost gratitude to my project supervisor Puan Nazean Jomhari whose guidance, encouragement and advice helped to bring the best out of me. She is a very dedicated lecturer and always willing to share her opinions to make the project a worthy one. Timely feedback, new ideas and suggestions from her made the system to be more interactive. Not forgetting my internal examiner Pn. Norjihan where all her comments were the inspiration to reproduce a good final documentation. Thank you very much Pn. Norjihan.

At the same time, I would like to thank all the students from all the local universities and private colleges who courteous and sportive enough to help me throughout the requirement collection session where they rendered their valuable time to answer the questionnaire and evaluate the system without any hesitation. Special thanks to all the faculty members of FCSIT, University Malaya who helped in the procedures and planning for the dissertation submission and tutors who helped in the testing phase. Last but not least, special thanks go to my bosses at my workplace who were very understanding throughout the completion of the project.

Table of Contents

Abstractii

Acknowledgementsiii

Table of Contents.....iv

List of Figuresix

List of Tables.....xii

Chapter 1 Introduction 1

1.1 Project Definition 1

1.2 Project Motivation 2

1.3 Project Objectives..... 3

1.4 Project Scope 4

1.5 Project Limitations 4

1.6 Target Audience 5

1.7 Project Expected outcome 5

1.8 Project Development Methodology..... 6

1.9 Project Schedule 6

Chapter 2 Literature Review..... 8

2.1 Introduction to Literature Review 8

2.2 Introduction to Artificial Intelligence..... 10

2.2.1 Expert System or Knowledge System..... 11

2.2.2 The structure of an expert system 11

2.2.3 What is Rule-Based System? 13

2.2.3.1 Rule-Based System Architecture 14

2.2.3.2 Inference Mechanisms or Techniques 15

2.3 Introduction to C Programming..... 17

2.3.1 Basic Structure and Data Types 18

2.3.2 Topics in C Programming 20

2.3.2.1 Arithmetic 21

2.3.2.1.1 Application of rules to teach arithmetic 21

2.3.2.2 Control Structures 24

2.3.2.2.1 Application of rules to teach control structures	27
2.3.2.3 Functions.....	30
2.3.2.3.1 Application of rules to teach function	31
2.3.2.4 Arrays and Sorts	34
2.3.2.4.1 Application of rules to teach arrays.....	36
2.3.2.5 Pointers	42
2.3.2.5.1 Application of rules to teach pointers	44
2.3.2.6 File Processing.....	52
2.3.2.6.1 Application of rules to teach file processing.....	52
2.3.3 Benefits of Designing in Rule-Based System Architecture	56
2.3.4 Analysis and Synthesis.....	56
2.4 Surveys on existing system	57
2.4.1 MYCIN	58
2.4.2 The Rule-Based Expert System Using an Interactive QA Sequence.....	59
2.4.3 INTELLITUTOR II	64
2.4.4 The ANDES Physics Tutoring System	67
2.4.5 Softsys.....	69
2.4.6 Carnegie Learning Cognitive Tutor Integrated Math I, II, III.....	70
2.5 CTutorial4u Vs Existing Systems	71
2.6 Software and Technologies	73
2.6.1 Visual Basic	74
2.6.2 Microsoft Access.....	75
Chapter 3 System Analysis	76
3.1 System Development Methodology	76
3.1.1 Waterfall Model	77
3.1.2 Linear Model of Experts System Development.....	78
3.1.3 Evolutionary Development Model.....	82
3.1.4 Analysis and Synthesis.....	83
3.2 Requirement Analysis	87
3.2.1 How to elucidate user requirement	87
3.2.1.1 Discussion with the project supervisor	88
3.2.1.2 Survey about C among students using Questionnaires.....	88
3.2.1.3 Surf through the Internet and research on reading materials	91

3.2.2 Functional requirements.....	91
3.2.2.1 Administration session.....	92
3.2.2.2 Lecture notes session	93
3.2.2.3 Tutorial session.....	93
3.2.2.4 Quiz session.....	97
3.2.3 Non-Functional Requirements	98
3.3 Development Environments	100
3.3.1 Hardware Requirements.....	100
3.3.2 Software Requirements	100
6.2.2 Module Testing	139
Chapter 4 System Design	102
4.1 Overview of Ctutorial4u Architecture.....	102
4.2 Process design	103
4.2.1 System structure chart.....	103
4.2.2 Flow chart diagram	104
4.3 UML Diagram	106
4.3.1 Class diagram.....	107
4.3.2 CTutorial4u Use Case Diagram	110
4.3.3 Sequence Diagram	111
4.4 Suggested Database Design.....	116
4.5 Interface Design.....	118
4.5.1 Sample User Interfaces	121
7.2.4 Validation of Input Data	130
Chapter 5 System Implementation	126
5.1 Development Environment.....	126
5.1.1 Hardware Configurations	126
5.1.2 Software Configurations	126
5.2 Project Development	127
5.2.1 Data Preparation.....	127
5.2.1.1 Still Images and animated graphics	128
5.2.1.2 Database preparation	128
5.2.1.3 Input form design.....	128
5.2.1.4 User interface design	129
5.2.2 Coding.....	129

5.2.2.1 Database connection	133
5.2.2.2 Authenticate member	133
5.2.2.3 Process with Database	134
5.2.3 System Integration	137
Chapter 6 System Testing	138
6.1 Introduction to System Testing	138
6.2 Testing Phase	138
6.2.1 Unit Testing.....	138
6.2.2 Module Testing	139
6.2.3 Overall system Testing.....	139
6.2.4 Test Case	140
6.2.4.1 User Module Test Case.....	142
6.2.4.2 Admin Module Test Case	145
6.3 Compliance of the system to its scope and requirements.....	145
Chapter 7 System Evaluation	147
7.1 Problems Encountered and Solutions.....	147
7.2 System Strengths	148
7.2.1 User Friendliness.....	148
7.2.2 Password Protected System – Security Features.....	149
7.2.3 Reliable System with Effective Error Handling.....	149
7.2.4 Validation on Input Data.....	150
7.2.5 Interactivity	150
7.2.6 Simplicity & Consistency	150
7.3 System Limitations.....	152
7.3.1 Poor Accessibility Feature	152
7.3.2 Lack of information quality	152
7.3.3 Quiz session only consists of multiple choices questions.....	152
7.4 Future Enhancements	153
7.4.1 Web- based system.....	153
7.4.2 Improve Information Quality	153
7.4.3 Quiz session to be more interactive	153
7.5 Comparison between before and after using CTutorial4u.....	153

Chapter 8 Conclusion.....	155
References	156
Bibliography.....	158
Appendix A: Questionnaire Form	159
Appendix B: Evaluation Form	161
Appendix C: User Manual.....	162
Figure 2.3: Rule Based System Architecture	14
Figure 2.4: Forward and backward chaining	17
Figure 2.5: Flowchart to teach Arithmetic (multiplication)	22
Figure 2.6: Flowchart to teach Control Structure (Selection Structure).....	28
Figure 2.7: Flowchart to teach Functions.....	33
Figure 2.8: Flowchart to teach Arrays.....	37
Figure 2.9: Flowchart to teach Pointers.....	45
Figure 2.10: Flowchart to teach File processing (Create a Textual File)	54
Figure 2.11: Main menu of rule-based expert system	60
Figure 2.12: Expert menu's input window	60
Figure 2.13: Solving problem window for user menu	61
Figure 2.14: Map Viewer of Rule-Based Expert System	62
Figure 2.15: Overview of the API	67
Figure 2.16: Andes screen.....	68
Figure 2.17: A dialogue box for drawing a vector	69
Figure 3.1: Waterfall Model	78
Figure 3.2: Linear Model Diagram For Expert System Development	79
Figure 3.3: Phases of Knowledge Engineering.....	82
Figure 3.4: Iterative Development Model (Exploratory Prototyping)	83
Figure 3.5: Line graph shows the features expected from CTutorial4u	92
Figure 3.6: Pie Chart shows votes on ways to design the tutorial	93
Figure 3.7: Bar chart shows students' choice of the most difficult C topic	94
Figure 3.8: Doughnut chart shows the responses on how to design the quizzes	95
Figure 3.9: Pie chart on time to finish a quiz consists of 10 simple questions.....	98
Figure 4.1: Architecture of CTutorial4u	102
Figure 4.2: Structure chart of CTutorial4u	104
Figure 4.3: Flow chart of CTutorial4u	105
Figure 4.4: Class Diagram of CTutorial4u.....	109

List of Figures

Figure 1.1: Gantt chart of Ctutorial4u	7
Figure 2.1: Human Expert Problem Solving	12
Figure 2.2: Expert System Problem Solving	13
Figure 2.3: Rule Based System Architecture	14
Figure 2.4: Forward and backward chaining	17
Figure 2.5: Flowchart to teach Arithmetic (multiplication)	22
Figure 2.6: Flowchart to teach Control Structure (Selection Structure)	28
Figure 2.7: Flowchart to teach Functions	32
Figure 2.8: Flowchart to teach Arrays	37
Figure 2.9: Flowchart to teach Pointers	45
Figure 2.10: Flowchart to teach File processing (Create Sequential File)	54
Figure 2.11: Main menu of rule-based expert system	60
Figure 2.12: Expert menu's input window	60
Figure 2.13: Solving problem window for user menu	61
Figure 2.14: Map Viewer of Rule-Based Expert System	62
Figure 2.15: Overview of the APLUS II	67
Figure 2.16: Andes screen	68
Figure 2.17: A dialogue box for drawing a vector	69
Figure 3.1: Waterfall Model	78
Figure 3.2: Linear Model Diagram For Expert System Development	79
Figure 3.3: Phases in Knowledge Engineering	82
Figure 3.4: Evolutionary Development Model (Exploratory Prototyping)	83
Figure 3.5: Line graph shows the features expected from CTutorial4u	92
Figure 3.6: Pie Chart shows votes on ways to design the tutorial	93
Figure 3.7: Bar chart shows students' choice of the most difficult C topic	94
Figure 3.8: Doughnut chart shows the responses on how to design the quizzes	98
Figure 3.9: Pie chart on time to finish a quiz consists of 10 simple questions	98
Figure 4.1: Architecture of CTutorial4u	102
Figure 4.2: Structure chart of CTutorial4u	104
Figure 4.3: Flow chart of CTutorial4u	105
Figure 4.4: Class Diagram of CTutorial4u	109

Figure 4.5: Use Case Diagram of CTutorial4u..... 111

Figure 4.6: Log In Sequence Diagram 113

Figure 4.7: Overall Tutorial Session Sequence Diagram 114

Figure 4.8: Overall Quiz Session Sequence Diagram 115

Figure 4.9: Sample of error message in the Tutorial Session..... 120

Figure 4.10: Login page 121

Figure 4.11: Main menu page..... 122

Figure 4.12: Lecture notes page 122

Figure 4.13: Tutorial session page (Arithmetic)..... 123

Figure 4.14: Tutorial session page (Arithmetic)..... 123

Figure 4.15: Quiz session page..... 124

Figure 4.16: Quiz to Guess Choice page 124

Figure 4.17: Administration session page (Add user) 125

Figure 5.1: Abstract of ‘Arithmetic’ Algorithm in the Tutorial session..... 130

Figure 5.2: Abstract of ‘Control structure’ Algorithm in the Tutorial session..... 131

Figure 5.3: Abstract of ‘Function’ Algorithm in the Tutorial session..... 132

Figure 5.4: Module page..... 133

Figure 5.5: Abstract of ‘Add New Question’ in Administration session 135

Figure 5.6: Abstract of Rndmz function in the Quiz session 136

Figure 6.1: Bottom up Testing..... 140

Figure 7.1: CTutorial4u’s contribution to improve C knowledge..... 148

Figure 7.2: User friendliness of Ctutorial4u 149

Figure 7.3: CTutorial4u’s error messages 150

Figure 7.4: Simplicity of CTutorial4u 151

Figure 7.5: Time taken to familiarize with the system 151

Figure C.1: Main functions of CTutorial4u..... 162

Figure C.2: CTutorial4u setup wizard 163

Figure C.3: Logon page 165

Figure C.4: Menu page 165

Figure C.5: Menu page highlighting the menus 166

Figure C.6: Contents page 167

Figure C.7: Technical Support page 168

Figure C.8: Lecture notes (Introduction) page 168

Figure C.9: Arithmetic tutorial page 171

Figure C.10: Arithmetic.exe	171
Figure C.11: Control Structure tutorial page.....	172
Figure C.12: CtrlStructure.exe	173
Figure C.13: Function tutorial page.....	174
Figure C.14: Function.exe	174
Figure C.15: Array tutorial page	176
Figure C.16: Array.exe	176
Figure C.17: Pointer tutorial page	177
Figure C.18: Pointer.exe.....	178
Figure C.19: File processing tutorial page	179
Figure C.20: Quiz session page.....	180
Figure C.21: Quiz to Choose answer page	181
Figure C.22: Score Card page	182
Figure C.23: Questions Attempted page	183
Figure C.24: Administration page	184
Figure C.25: Input box for Admin ID	185
Figure C.26: Question page.....	186
Figure C.27: User page.....	188
Table 4.6: Quiz Table	117
Table 6.1: Logon page test case	142
Table 6.2: Menu page test case	142
Table 6.3: Lecture Note page test case	143
Table 6.4: Tutorial session page test case	143
Table 6.5: Quiz session page test case	144
Table 6.6: Administration page test case	145
Table 7.1: Problems Encountered and Solutions.....	147
Table 7.2: Comparison between before and after using Tutorial4u	154
Table C.1: Arithmetic Tab.....	170
Table C.2: Control Structure Tab	172
Table C.3: Function Tab	173
Table C.4: Array Tab	175
Table C.5: Pointer Tab	177
Table C.6: File processing Tab	178

List of Tables

Table 2.1: The main structure of expert system	12
Table 2.2: IF-THEN rules for Arithmetic.....	23
Table 2.3: IF-THEN rules for Control Structure	29
Table 2.4: Examples of math library functions	30
Table 2.5: IF-THEN rules for Function.....	33
Table 2.6: IF-THEN rules for Array.....	38
Table 2.7: IF-THEN rules for Pointer	46
Table 2.8: IF-THEN rules for File Processing	55
Table 2.9: Example of expert systems.....	57
Table 2.10: Rules for Rule-Based Expert System	63
Table 3.1: Software requirements for Ctutorial4u.....	101
Table 4.1: Class Notation Table	108
Table 4.2: Sequence Diagram Notation Table	112
Table 4.3: User Table	117
Table 4.4: Admin Table.....	117
Table 4.5: Tutorial Table.....	117
Table 4.6: Quiz Table	117
Table 6.1: Logon page test case	142
Table 6.2: Menu page test case.....	142
Table 6.3: Lecture Notes page test case	143
Table 6.4: Tutorial Session page test case.....	143
Table 6.5: Quiz Session page test case	144
Table 6.6: Administration page test case.....	145
Table 7.1: Problems Encountered and Solutions.....	147
Table 7.2: Comparison between before and after using Ctutorial4u	154
Table C.1: Arithmetic Tab.....	170
Table C.2: Control Structure Tab	172
Table C.3: Function Tab.....	173
Table C.4: Array Tab	175
Table C.5: Pointer Tab	177
Table C.6: File processing Tab.....	178

Chapter 1 Introduction

1.1 Project Definition

CTutorial4u is a knowledge-based system, which incorporated with rule-based system architecture as the backbone to develop tutorial for C programming. One of the capabilities that distinguish the CTutorial4u as a medium for learning is the ability it provides to individual users to participate in the system as well as receive results after going through the quizzes.

The concept of knowledge management also plays a valuable role in the context of intelligent learning. All of the same elements of knowledge management exist in this CTutorial4u such as tools for capture, central database for storage and tools for help. However, the objective is to develop a resource for learners seeking to improve their understanding of C Programming. The deployment of such a system not only provides a valuable information resource, but it also promotes a sense of community amongst learners by providing them the opportunity to test their understanding in the quiz session provided.

This system offers the users the opportunity to familiarize with C Programming by providing lecture notes session. This session allows the users to refer to the notes anytime they want. It consists of ten most important chapters of C programming. The administrator (lecturers and tutors) can upload and change the notes. Then the tutorial part is for the user to visualize what they have learned. The user is required to choose the answers for each selection appeared for the domain selected. Once all the answers gathered and entered, the system will show the source code for the domain based on the selected. Only six important domains will be taught in the tutorial session. Next, the

quiz session consist of ten simple questions for each domain for the students to test their understanding of the lectures and tutorials they gone through earlier.

1.2 Project Motivation

The purpose of the project is to assist and guide students and lecturers to excel in the C programming course. A tutorial is NOT a further chance for staff members to teach students. The tutorial is NOT "owned" by the tutor. All participants should own it. A tutorial should facilitate student learning and should take into account the abilities, needs and interests of each individual involved in a way that lectures cannot.

Most of the students from various public institutes and private colleges finds it hard to get an A grade or a good results in programming subjects. Reason for increase in failure percentage in C programming is because the students are least exposed to programming languages in school levels and at the same time it is normally first year course in every universities, so the students finds it difficult to familiarize and master in this course.

The knowledge design in rule based system architecture actually run based on some standard rules set in the working memory. The rules will remain the same for different runs. CTutorial4u not only serve as an exercise to a student but actually to improve basic understanding in C programming and as a tool to test their knowledge and judge ability and efficiency of answering different set of quizzes with various concepts in programming language. Other than that, actually it is easier and effective to convey the easy learning system to the students in the laboratory where installing this system in every PC will actually allow all the students to utilize it.

1.3 Project Objectives

CTutorial4u's main objective is to provide flexible knowledge base foundation in C programming topics such as arithmetic, control structure, function, array, pointer and file processing. Using this system, users who are not good at both programming and problem domain can get solution to a specific problem easily from the constructed rules through the interactive question and answer sequence. The said solution is in the form of C source code and incorporated with the result. The knowledge base structure is essential to the successful execution of rules underlying the processes involved in the tutorial and quiz sessions. The knowledge base designed in such a way to promote scalability, extension and modifiability to the basic design of the system.

The system consists of lecture notes to provide simple understanding of topics in C language, tutorial session to help the students to construct source code based on choice of selection and quiz session to test the user's proficiency, understanding and knowledge gained from lecture notes and tutorial session. The quiz presented with multiple-choice questions and the learner had to respond correctly. Some interesting ideas have emerged to mechanize teaching using various teaching methods. The sub objectives of the knowledge designed rule-based C Tutorial was seen as:

- save time and cost by moving towards paperless
- provide interactive learning system (user select choices of answers, system execute and manipulate the source code stored in the database based on the input),
- exploit the "law of exercise" by providing quiz,
- Provide timely and immediate feedback for tutorial and quiz.

This tutoring system is capable of interacting with the user by accepting the user input and amends the code saved in the working memory. Process the user input and fix the input into the basic program in the working memory and display the recent program code after grabbing some user input from the selection criteria such as variable type, variable value and value of the variable. The best of the tutorial session is to display the output of the program after it has been executed.

1.4 Project Scope

The tutorial includes arithmetic, control structure, function, array, pointer and file processing. Lecture notes will cover the main ten topics in C programming and presented to users in power point format whereas quiz session only has questions from six main topics covered in tutorial session. Each time a person logon to the system, it randomly chooses different set of questions based on the topic selected. The quiz set the finish time and student will be given a mark immediately after the time out.

Tutorial and quiz session is designed to evaluate the student's learning and provide constructive feedback where necessary. It is an aid to learning and not an elimination process. Vague and confusing questions will not be asked.

- Tutorial question
- Possible answer choices
- Correct answer + (feedback)

1.5 Project Limitations

CTutorial4u provides a lot of benefits but on contrary it has a few limitations as well. This system can not be accessed by users from anywhere at anytime. This system is

limited to those who have the CD. Other than that, the system is intended to be installed in all laboratories in FCSIT at University Malaya, so only FCSIT students will be exposed to the system. At the same time, the CTutorial4u does not provide any subjective questions which actually a major setback of a tutorial system. Besides, C compiler is not embedded to test whether the written codes by the users are correct nor has bugs in there.

1.8 Project Development Methodology

1.6 Target Audience

CTutorial4u is specifically designed to help and guide student from universities and private colleges to train and practice programming skills and test their ability through the quiz session available. On the other hand, tutorial session will be reference to their programming logic. At the same time, users of the systems will be lecturers who would like to refer to the system for some kind of quiz questions with answers based on the chapters available in the course, which will make their work easier. Targeted audience is beginners and intermediate users of C programming tools.

1.7 Project Expected outcome

The system is expected to be user friendly. The tutorials will be designed in a manner to cover all the topics in C programming. Standard C programming topics will be analysed based on various reference books available in the market to produce the tutorial session.

In quiz session, each time the person logon to the system, it randomly choose different quiz question. The quiz set the finish time and student will be given a mark immediately after the time out. It is attached with [Good] message if they get more than

50% and [Try again] message if they get less than 50%. Apart from that, the quiz session has embedded interface metaphor in human computer interaction concepts. If the answer for the question is correct then, there will be correct sign [✓] on the question coloured green and wrong sign [X] on the wrong answer coloured red. Lecture notes as a supplementary feature which to be in power point format.

1.8 Project Development Methodology

Linear Model of Knowledge engineering suits the best for developing Ctutorial4u since it's a rule-based system because it is better to adopt a system development strictly designed for expert system and it will be further described in detail in chapter 3.

1.9 Project Schedule

The project is a 2-semester project. For the first semester, standard questionnaires will be prepared and delivered to local universities and some private colleges. At the same time, other similar system or system which serves the same purpose will be analyzed to get new ideas and to improve the existing setbacks. Collected data will be analyzed and based on the analysis the new system will be planned and a simple screen designs will be scratched. Prototype of the system will be developed similar to the paper scratches.

On the second semester, based on the comments from supervisor and some of the colleagues, the prototype was redesigned. The real system design took place during second semester. The system was tested and evaluated by a number of students and tutors of FCSIT, UM. The documentation is submitted to the internal and external to examine the documentation. The formal presentation on the system was held on 9th October 2004 to all the faculty members of FCSIT, UM. The gantt chart of the project schedule is as in the Figure 1.1 in the format of MS Project.

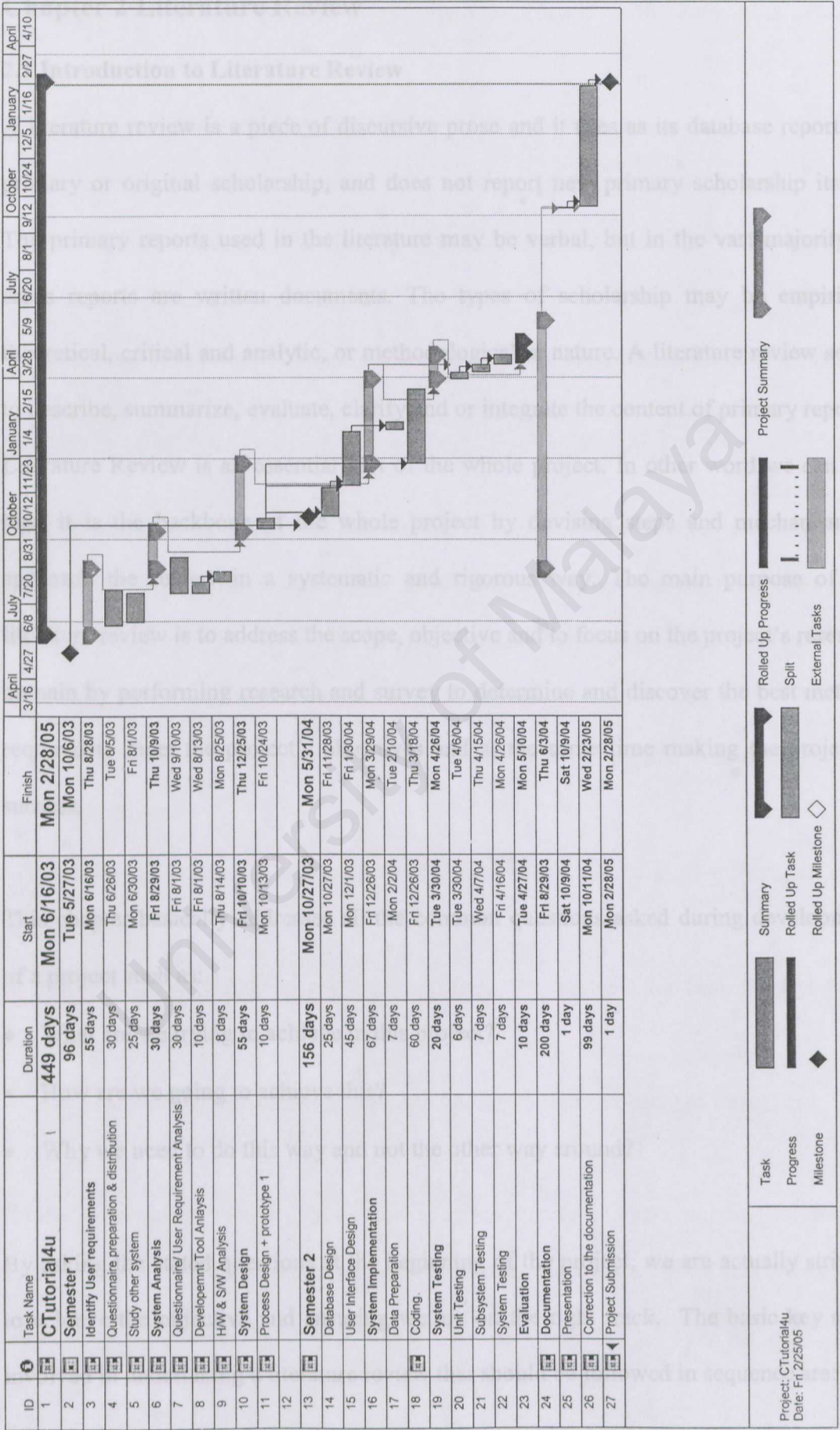


Figure 1.1: Gantt chart of Ctutorial4u

Chapter 2 Literature Review the literature review in the form of a rationale

2.1 Introduction to Literature Review

A literature review is a piece of discursive prose and it uses as its database reports of primary or original scholarship, and does not report new primary scholarship itself.

The primary reports used in the literature may be verbal, but in the vast majority of cases reports are written documents. The types of scholarship may be empirical, theoretical, critical and analytic, or methodological in nature. A literature review seeks to describe, summarize, evaluate, clarify and or integrate the content of primary reports.

Literature Review is an essential part of the whole project. In other word we can say that, it is the backbone of the whole project by devising steps and mechanism to approach the review in a systematic and rigorous way. The main purpose of the literature review is to address the scope, objective and to focus on the project's research domain by performing research and survey to determine and discover the best method required to meet the project's objectives and at the same time making the project a success.

• Surveys of existing system and comparison with the new system intended to be

This chapter basically addresses all the common questions asked during development of a project such as:

- What are we trying to achieve in this project?
- How are we going to achieve this?
- Why we need to do this way and not the other way around?

Source of research varies such as books, questionnaire analysis, existing knowledge

By asking the stated questions at the beginning of the project, we are actually striving to achieve the objective and ensuring we are on the right track. The basic key steps involved in undertaking a literature review that should be followed in sequence are:

- Clarification of the purpose of the literature review in the form of a rationale statement
- Planning the review through drawing up a blueprint document
- Conducting a comprehensive literature search, according to the blueprint
- Selection and focused reviewing of individual items, according to the blueprint, creating a set of individual reviews
- Integrated or 'synthesis' reviewing according to the blueprint, to produce the review document.

Specifically to this project, this chapter allows to gather the essential and imperative information needed for the development of the tutorial system for C in rule based system architecture. The review of the project actually divided into five main parts:

- General overview of the rule based expert system
- Research on C programming topics, which can be applied rule-based system architecture
- Surveys of existing system and comparison with the new system intended to be developed
- Questionnaire design to get feedback from the students on how it need to be designed
- Overview of the technologies available to ease the development of the system

Source of research varies such as books, questionnaire analysis, existing knowledge based systems, tutorial systems and online articles. Later sections are undoubtedly the outputs of the five key steps mentioned above.

2.2 Introduction to Artificial Intelligence

Artificial Intelligence is a branch of science which deals with helping machines to find solutions to complex problem in a more human-like fashion. There are five branches of AI such as robotics, vision, natural language understanding, sound recognition and knowledge system [2].

Robotics is the study of machines to perform many human-like tasks such as mechanical manipulation and how to make them function with some "intelligence" and autonomy. Robotics is proving to be very valuable to the manufacturing industry. There are robotic automobile painters and assembly line workers.

Vision systems are those that successfully interpret two- and three-dimensional pictures from a two-dimensional image obtained through manmade sensors. This involves processing the image, classifying it and then interpreting the scene.

Natural language is human language. Natural language understanding is the ability to communicate with a computer by conventional language text, such as English and Russian instead of a highly structured language such as standard query language (SQL).

Sound recognition is the field that the goal is to enable machines to listen and understand their auditory environment in which processing and reasoning about acoustic sensors such as alarms, spoken words, or automobile engines takes place.

Sound recognition systems take an audible sound and make it readable.

Knowledge systems or expert system are software systems that have structured knowledge about a field of expertise. They are able to solve some problems within their

domain by using knowledge derived from experts in the field. This approach emphasizes data interpretation.

Knowledge Base	Maintain the expert domain knowledge in a module. Knowledge obtained from the expert is coded here using one of the representations.
----------------	--

2.2.1 Expert System or Knowledge System

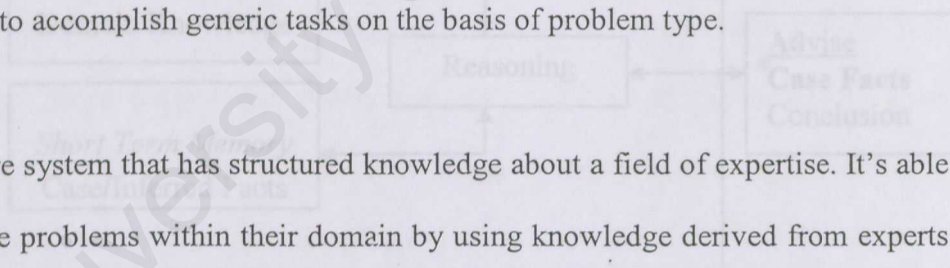
Working memory	Contain the facts about a problem that are discovered during the problem solving process.
Inference Engine	Processor to compare and match the facts contained in the knowledge base to conclude the problem.

The area of knowledge systems has blossomed over the past decade from merely an academic interest into a useful technology. Expert system is one of the most popular and feasible facets of Artificial Intelligence. The most fitting definition of expert

system itself is ‘a computer program that represents and reasons with knowledge of some specialists subject with a view to solve problems or to give advice’ [6].One of the

ways to categorize the application of expert system is a problem-solving paradigm. Experts perform a generic set of tasks when solving certain types of problem such as

diagnosis or planning. Regardless of the application area, given the type of problem, the expert collects and reasons with information in similar ways. Expert systems likewise are designed to accomplish generic tasks on the basis of problem type.



It’s a software system that has structured knowledge about a field of expertise. It’s able to solve some problems within their domain by using knowledge derived from experts in the field. Development of the methods for knowledge representation followed the knowledge acquisition phase. With a suitable amount of knowledge gathered, the structure and representation method for the knowledge system can be described.

There are two traits of an expert that are attempted to model an

2.2.2 The structure of an expert system

The Table 2.1 describes the main structure of an expert system.

The knowledge base contains highly specialized knowledge on the problem area as provided by the expert. It includes problem facts, rules, concepts and relationship. The inference engine is the knowledge processor, which works as the

Table 2.1: The main structure of expert system

Structure	Description
Knowledge Base	Maintain the expert domain knowledge in a module. Knowledge obtained from the expert is coded here using one of the several knowledge representations.
Working memory	Contain the facts about a problem that are discovered during consultation or inferred by the system.
Inference Engine	Processor to compare and match the facts contained in the working memory with the domain knowledge contained in the knowledge base to conclude the problem.
Explanation Facility	Provides an explanation to the user about why it is asking a question and how it reached some conclusion.

The Figure 2.1 depicts human being’s problem solving method. Based on the domain knowledge stored in the long term memory and inferred facts obtained from the short term memory, human could reason a problem and make a conclusion.

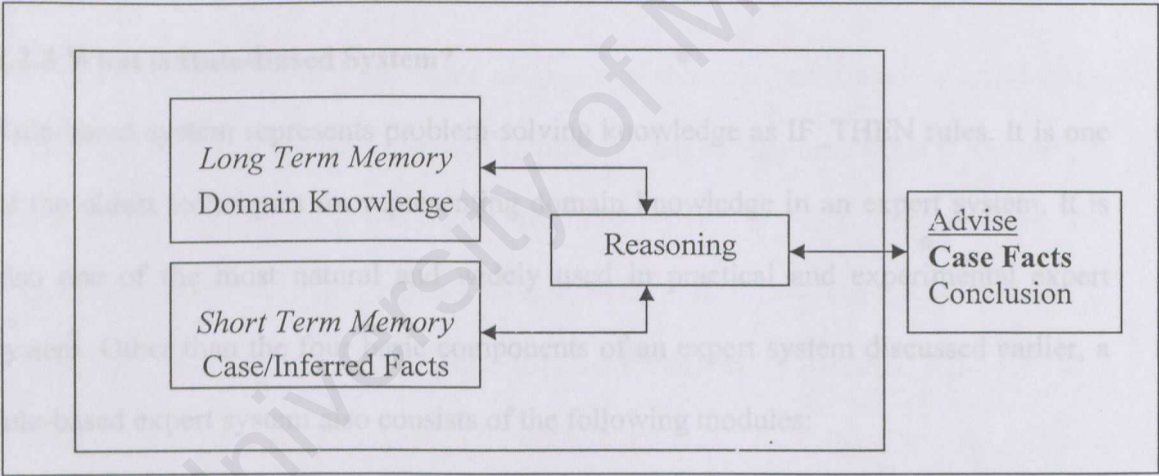


Figure 2.1: Human Expert Problem Solving

Expert system’s goal is to enable many people to benefit the knowledge of one person which is the expert. There are two traits of an expert that are attempted to model an expert system which are the expert’s knowledge and reasoning. The system must have two principles modules such as a knowledge base and an inference engine to accomplish this. The knowledge base contains highly specialized knowledge on the problem area as provided by the expert. It includes problem facts, rules, concepts and relationship. The inference engine is the knowledge processor, which works as the

reasoning in the human expert problem solving. The knowledge gained through consultation about a problem is stored in the working memory.

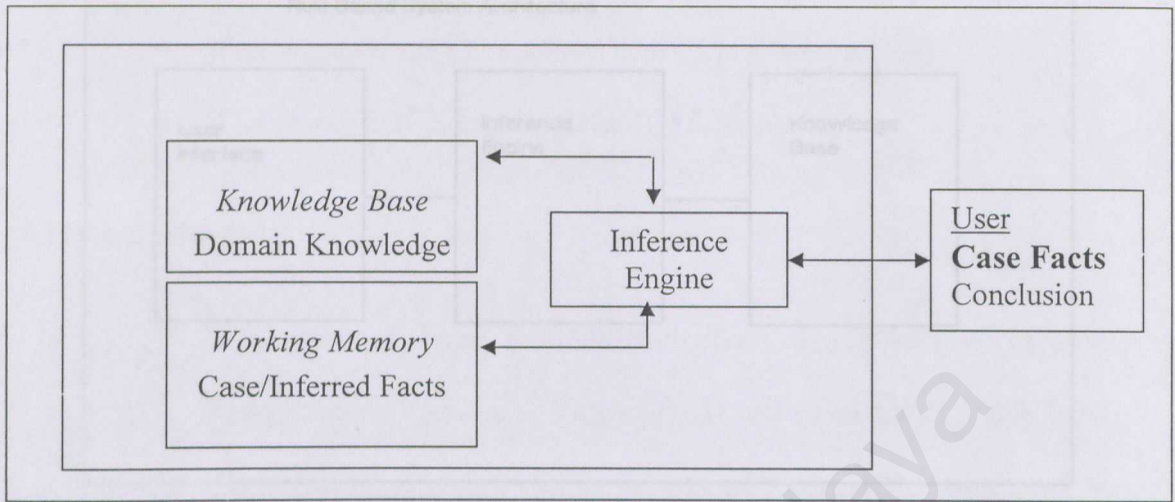


Figure 2.2: Expert System Problem Solving

2.2.3 What is Rule-Based System?

Rule-based system represents problem-solving knowledge as IF_THEN rules. It is one of the oldest techniques for representing domain knowledge in an expert system. It is also one of the most natural and widely used in practical and experimental expert system. Other than the four basic components of an expert system discussed earlier, a rule-based expert system also consists of the following modules:

- **User Interface:** It is the vehicle through which a user views and interacts with the system.
- **External Program:** External program are programs such as spreadsheets and algorithms that work in support for the system.

2.2.3.1 Rule-Based System Architecture

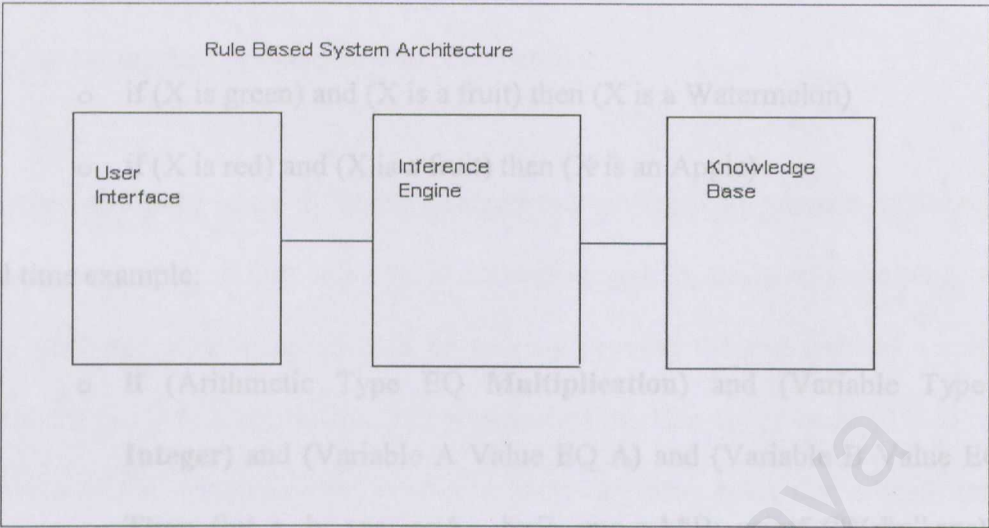


Figure 2.3: Rule Based System Architecture

The above figure shows three major elements of the rule based system architecture, which are the user interface, inference engine and knowledge base. This is the runtime architecture. The user starts the system and interacts with it via the user interface. The engine is the part of the program that actually does stuff. It says, run the system, that is, look at working memory, see what rules fire, and apply them. Actually, same inference engine applies for each rule base. The knowledge base is the rules and the working memory. The rules will remain the same for different runs. Working memory (WM) changes for each run and during the run. Basically working memory is the database, the medium to store the information derived from the system when or during the system run.

The knowledge base consists of rules and working memory (WM). Rules are if and then statements. On the **If** side there will be conditional expressions as (X is green). At the same time, we can have variables in here, in this case X is a variable and on the

Then side, usually have assignments that are set or modify working memory items.

Below are some rules:

- if (X is green) and (X is a fruit) then (X is a Watermelon)
- if (X is red) and (X is a fruit) then (X is an Apple)

A real time example:

- **if** (Arithmetic Type EQ **Multiplication**) and (Variable Type EQ **Integer**) and (Variable A Value EQ **A**) and (Variable B Value EQ **B**)
Then (int a, b, ans; a=A, b=B, ans =A*B; printf ("%d\n",ans) and
Result = A*B;)

2.2.3.2 Inference Mechanisms or Techniques

Expert systems model the reasoning process of humans using technique called inference, which derives new information from known information. In rule-based systems, knowledge is represented as facts about the world and rules to manipulate the facts. At any one time more than one rule may be applied to solve a problem and when each rule is applied other rules may applicable to those rules. Therefore, a rule-based system needs a control structure to decide which rule should be applied first or next and which rules are put together. The two basic inference techniques used in an expert system are forward and backward-chaining. Both techniques are compared to determine the best inference strategy to be adopted into the CTutorial4u project.

Forward-chaining is an inference strategy where conclusions are drawn by first looking at the facts or data on the problem. This style of reasoning is also known as data-driven search. In a rule-based system, forward chaining begins by asserting certain

facts, seeing what rules can fire based on these assertions, picking a rule to fire, then cycles and checks the rules again looking for new matches. This process is continued until a goal is reached or no additional rules can fire.

Backward-chaining is an inference strategy that attempts to prove a hypothesis by gathering supporting information. In a rule-based system, backward chaining begins with a goal and tries to prove it to be true by proving the premises of a rule that contains the goal as its conclusion. The premises of this rule are considered 'sub goals', which the system tries to prove, is true by pursuing other rules that contain the sub goals as conclusions. Eventually, this backward chaining sequence reaches premises that are not supported by other rules and the user is then asked to verify the truth of the premise statement. This type of inference strategy is also called goal-driven search.

In forward chaining, the inferences would be made in the order indicated by the numbers on left flow. Forward chaining uses the depth of the tree structure of condition-action relation. If F was true and a solution before it is inferred that E is true, it may not need to infer that E is true. This forward chaining is used to infer new facts from existing facts. It is also possible to use the same set of rules in reverse to determine what needs to be true for a premise to be true. This method is called backward chaining. Backward chaining is commonly used in rule-based expert systems to enable a hypothesis to be tested and this problem-solving strategy is often referred to as generate and test.

Given an assertion, it is expected that both the assertion and the conclusion are valid throughout the session.

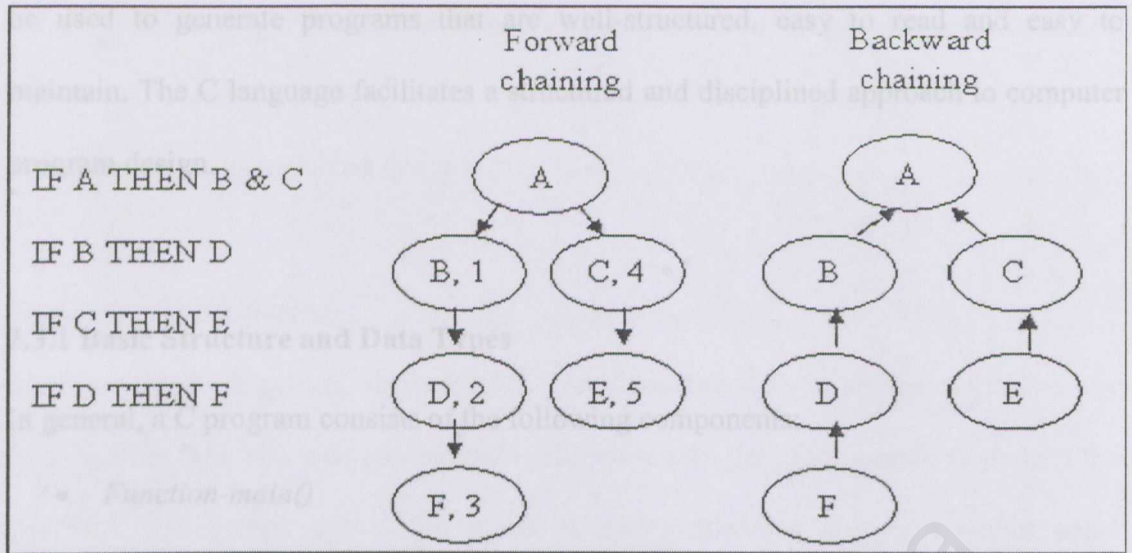


Figure 2.4: Forward and backward chaining

Monotonic reasoning is when a system retains its facts as unchanged assertions. It is more appropriate to be used for a diagnostics and prescription tasks. Non-monotonic reasoning is a method of reasoning that allows for changes in a given fact. Non-monotonic reasoning is an important feature of expert system applied to planning or design tasks.

2.3 Introduction to C Programming

The C programming was developed by Dennis Ritchie at Bell Laboratories in the early 1970s as a system implementation language. From then till now it has evolved into a general-purpose language that combines the convenience of high level languages with the power of assembly language. Currently, standard C compilers are available for many microcomputers, mini computer systems and mainframes. C is becoming increasingly more popular in a variety of computer applications. The language has many powerful features and it is possible to develop portable programs in it and at the same time it is to master and when coupled with good program design techniques it can

be used to generate programs that are well-structured, easy to read and easy to maintain. The C language facilitates a structured and disciplined approach to computer program design.

2.3.1 Basic Structure and Data Types

In general, a C program consists of the following components:

- *Function-main()*
- *Program comments*
- *Preprocessor directives*
- *Data Type declarations*
- *Variables*

Function –main()

Each C program must have one main function. The function main in a program marks the entry point of a program. The opening and closing parentheses () indicate that the identifier is a function. The opening and closing braces {} define the body of the function. The format is

```
main (){  
    statements  
}
```

Program Comments

Comments are text statements that document and describe the program. Comments are optional, non-executable statements that are placed in the program to explain what the

program does and how the code works. (The computer does not process non-executable statements.) A comment begins with `/*` and ends with `*/`. The example is as below:

```
/* A comment may be coded like this */
```

Preprocessor Directives

A preprocessor directive, also called a compiler directive is an instruction to the preprocessor. The `#include` preprocessing directive tells the preprocessor to replace the directive with a copy of the file specified by the filename argument within angle brackets `<>`. The preprocessing directive instructs the preprocessor to modify the source code program. The example is:

```
#include <stdio.h>
```

According to the example above a copy of the standard input/output header file replaces the directive in the source code. The `stdio.h` header file enables the program to perform basic input and output operations.

Data Types

Data types stipulates (in what format) the data is stored in memory. In C language, built-in data types are classified as fundamental data types and derived data types. Fundamental data types correspond to the most common, fundamental units of a computer and the most common, fundamental ways of using such data. Although C allows other types, these are the only ones commonly being used.

- **int** –to declare numeric program variables of integer types and restricted to whole number such as 5, 16 and 78. Integer variables hold data in the range –32768 to 32767. If it is beyond or less than the above value then the number is declared using long data type.

2.3.1 • **float**- to declare real or floating point numbers have decimal points such as 2.6571, 74.9 and 567.89.

• **char**- to declare character variables such as single letter, numeric digit, punctuation mark or special symbol. If more than one character, then we define that as string.

2.3.2.1 Application of rules to teach arithmetic

Variables

A variable is a data item that may assume different values. Example of how to declare variable is shown below:

```
int grade;
```

```
int final;
```

The variable grade and final are declared as integers (int). A global variable is declared outside of main and is available to the whole program. A local variable on the other hand is declared inside a specific program function and is not available to any other function.

2.3.2 Topics in C Programming

C programming covers a variety of topics such as arithmetic, control structures, counters, functions, arrays, pointers and file processing. After a thorough analysis of all the topics in C, the above mentioned topics are the topics that will be discussed in detailed in this chapter.

2.3.2.1 Arithmetic

Arithmetic in C is regarding numeric data types and arithmetic calculations. Arithmetic expressions are performed using arithmetic operators. They are (+) for addition, (-) for subtraction, (*) for multiplication, (/) for division and (%) for remainder or modulus.

2.3.2.1.1 Application of rules to teach arithmetic

Based on the literature review, topic arithmetic can be easily taught to the student in by applying rule based system architecture. Based on the question and answer from the system the solution or code for the arithmetic selected is shown or given for the user's view. Figure 2.5 shows the arithmetic for multiplication type which selects the proper code based on the algorithm in the next page. Table 2.2 defines the rules for all the arithmetic types and the algorithm is the same for all arithmetic type.

Q: Problem	Algorithm
A: Arithmetic	If Problem is Arithmetic
Q: Types of Arithmetic	Then many types of Arithmetic
A: Multiplication	If type is Multiplication
Q: Variable type?	Then many types of variable type
A: Integer	If variable type is integer
Q: Values of variable A and B	Then value of the variables
A: A is A and B is B	If Value A is A and B is B
	Then choose Code for
	Arithmetic to multiple integer A
	& integer B

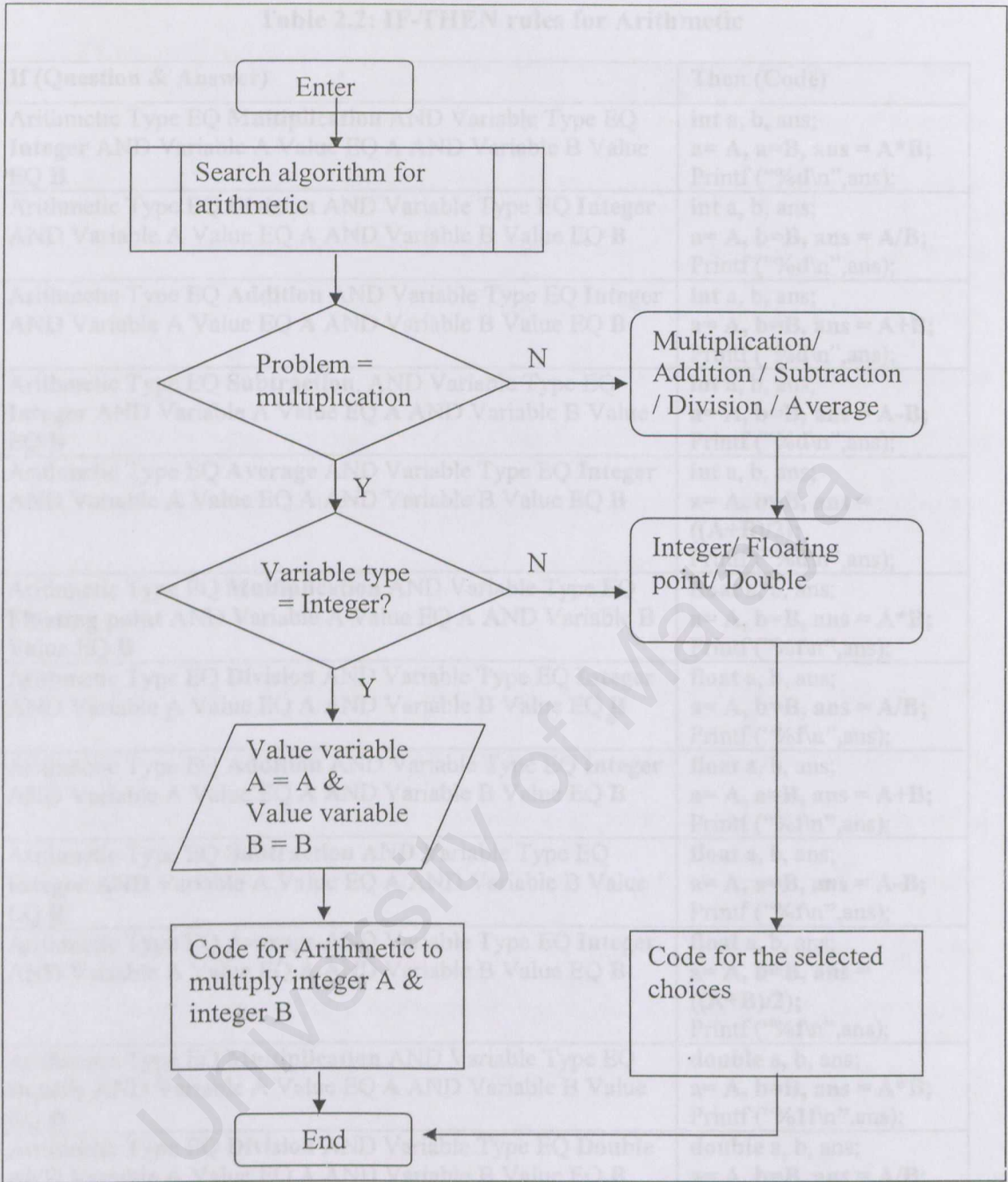


Figure 2.5: Flowchart to teach Arithmetic (multiplication)

Table 2.2: IF-THEN rules for Arithmetic

If (Question & Answer)	Then (Code)
Arithmetic Type EQ Multiplication AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	int a, b, ans; a = A , a = B , ans = A*B ; Printf ("%d\n",ans);
Arithmetic Type EQ Division AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	int a, b, ans; a = A , b = B , ans = A/B ; Printf ("%d\n",ans);
Arithmetic Type EQ Addition AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	int a, b, ans; a = A , b = B , ans = A+B ; Printf ("%d\n",ans);
Arithmetic Type EQ Subtraction , AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	int a, b, ans; a = A , b = B , ans = A-B ; Printf ("%d\n",ans);
Arithmetic Type EQ Average AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	int a, b, ans; a = A , b = B , ans = ((A+B)/2) ; Printf ("%d\n",ans);
Arithmetic Type EQ Multiplication AND Variable Type EQ Floating point AND Variable A Value EQ A AND Variable B Value EQ B	float a, b, ans; a = A , b = B , ans = A*B ; Printf ("%f\n",ans);
Arithmetic Type EQ Division AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	float a, b, ans; a = A , b = B , ans = A/B ; Printf ("%f\n",ans);
Arithmetic Type EQ Addition AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	float a, b, ans; a = A , a = B , ans = A+B ; Printf ("%f\n",ans);
Arithmetic Type EQ Subtraction AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	float a, b, ans; a = A , a = B , ans = A-B ; Printf ("%f\n",ans);
Arithmetic Type EQ Average AND Variable Type EQ Integer AND Variable A Value EQ A AND Variable B Value EQ B	float a, b, ans; a = A , b = B , ans = ((A+B)/2) ; Printf ("%f\n",ans);
Arithmetic Type EQ Multiplication AND Variable Type EQ Double AND Variable A Value EQ A AND Variable B Value EQ B	double a, b, ans; a = A , b = B , ans = A*B ; Printf ("%1f\n",ans);
Arithmetic Type EQ Division AND Variable Type EQ Double AND Variable A Value EQ A AND Variable B Value EQ B	double a, b, ans; a = A , b = B , ans = A/B ; Printf ("%1f\n",ans);
Arithmetic Type EQ Addition AND Variable Type EQ Double AND Variable A Value EQ A AND Variable B Value EQ B	double a, b, ans; a = A , b = B , ans = A+B ; Printf ("%1f\n",ans);
Arithmetic Type EQ Subtraction AND Variable Type EQ Double AND Variable A Value EQ A AND Variable B Value EQ B	double a, b, ans; a = A , b = B , ans = A-B ; Printf ("%1f\n",ans);
Arithmetic Type EQ Average AND Variable Type EQ Double AND Variable A Value EQ A AND Variable B Value EQ B	double a, b, ans; a = A , b = B , ans = ((A+B)/2) ; Printf ("%1f\n",ans);

2.3.2.2 Control Structures

C has seven control structures all together, namely sequence, three types of selection structure and three types of repetition. The three types of selection structures are **if**, **if/else** and **switch** structure. The **if** structure is called a single-selection structure because it selects or ignores a single action where it either performs or selects an action if a condition is true or skips the action if the condition is false. On contrary, the **if/else** selection structure performs an action if a condition is true and performs a different action if the condition is false. The **switch** selection structure performs one of many different actions depending on the value of the expression.

At the same time, C also provides three types of repetition structure such as **while**, **do/while** and **for**. The **while** repetition structure allow us to specify that an action is to be repeated while some condition remains true whereas the **do/while** repetition structure tests the loop continuation condition after the loop body is performed therefore the loop body will be executed at least once. The **for** loop is to set up a counter-controlled loop. The **for** statement repeats the statement in the loop a given number of times where the statement body executes as long as the condition test is true.

If Structure

Purpose: To set up one-way conditional branch.

```
if (credits<45){  
    printf ("Welcome freshman");  
}
```

If/Else Structure

Purpose: To set up a two-way conditional branch

```
if (credits < 45) {  
    printf ("Welcome freshman");  
}  
else {  
    printf ("Welcome upperclassman");  
}
```

Switch Structure

Purpose: To set up a multi-path conditional path. The switch structure allows the program to select one option from a given set of options.

```
int choice;  
.....  
printf ( " Enter your choice...\n");  
printf ("Movie menu: 1-Action, 2-Comedy, 3-Drama \n");  
Switch (choice) {  
    case 1:  
        printf ("Action movie fan\n");  
        break;  
    case 2:  
        printf ("Comedy movie fan\n");  
        break;  
    default:  
        printf ("Invalid choice\n");  
        break;}
```

While Structure

Purpose: To print numbers from 1 to 10.

```
main ()
{
    int counter = 1;

    while (counter <= 10) {
        printf ("%d\n", counter);
        ++counter; }

    return 0;
}
```

Do While Structure

Purpose: To print the numbers from 1 to 10.

```
main() {
    int counter = 1;

    do {
        printf ("%d ", counter);

    }

    while (++counter <= 10);

    return 0;
}
```


For Structure

Purpose: To print the numbers from 1 to 10.

```
main()
{
    int counter;

    for (counter = 1; counter <= 10; counter++)
        printf ("%d\n", countr);

    return 0;}
```

2.3.2.2.1 Application of rules to teach control structures

Based on the analysis, control structures can be taught to the student in an easier way if we apply rule based system architecture. Based on the question and answer from the system the control structure can be designed in rule based system architecture. The correct codes are shown to the users based on the algorithm below. Figure 2.6 shows the control structure for only selection structure and Table 2.3 contains all the if-then rules to be used for control structures.

Q: Problems?	Algorithm
A: Control Structure	If Problem is Control Structure
Q: Type of Control Structure	Then many types of Control Structure
A: Selection structure	If type is Selection structure
Q: How many action?	Then number of action to be performed
A: 2	If action is 2
	Then take algorithm for if/else

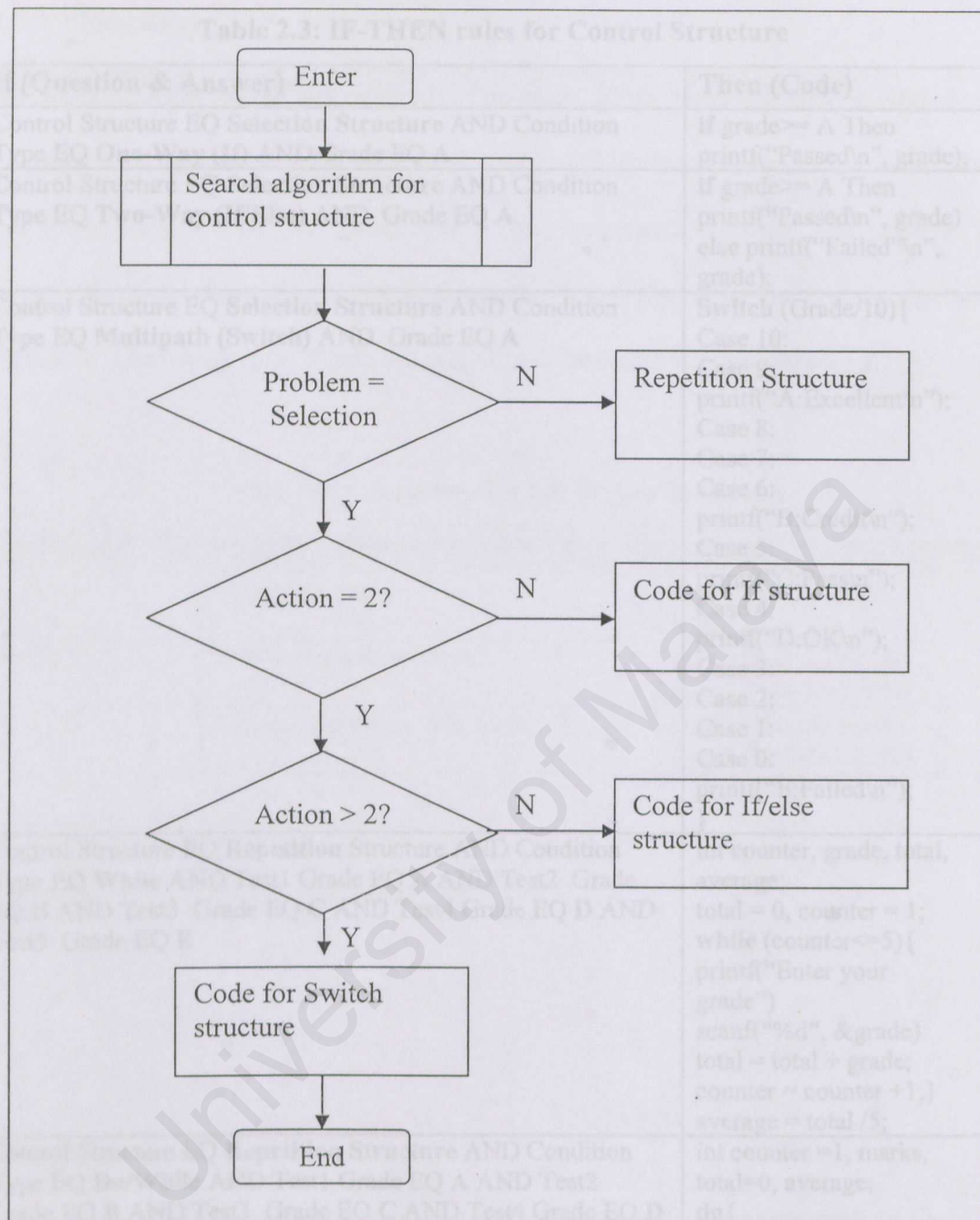


Figure 2.6: Flowchart to teach Control Structure (Selection Structure)

Table 2.3: IF-THEN rules for Control Structure

If (Question & Answer)	Then (Code)
Control Structure EQ Selection Structure AND Condition Type EQ One-Way (If) AND Grade EQ A	if grade >= A Then printf("Passed\n", grade);
Control Structure EQ Selection Structure AND Condition Type EQ Two-Way (If/Else) AND Grade EQ A	if grade >= A Then printf("Passed\n", grade) else printf("Failed\n", grade);
Control Structure EQ Selection Structure AND Condition Type EQ Multipath (Switch) AND Grade EQ A	Switch (Grade/10){ Case 10: Case 9: printf("A:Excellent\n"); Case 8: Case 7: Case 6: printf("B:Credit\n"); Case 5: printf("C:Pass\n"); Case 4: printf("D:OK\n"); Case 3: Case 2: Case 1: Case 0: printf("E:Failed\n"); }
Control Structure EQ Repetition Structure AND Condition Type EQ While AND Test1 Grade EQ A AND Test2 Grade EQ B AND Test3 Grade EQ C AND Test4 Grade EQ D AND Test5 Grade EQ E	int counter, grade, total, average; total = 0, counter = 1; while (counter <= 5){ printf("Enter your grade") scanf("%d", &grade) total = total + grade; counter = counter + 1;} average = total / 5;
Control Structure EQ Repetition Structure AND Condition Type EQ Do/While AND Test1 Grade EQ A AND Test2 Grade EQ B AND Test3 Grade EQ C AND Test4 Grade EQ D AND Test5 Grade EQ E	int counter = 1, marks, total = 0, average; do { printf("Enter marks") scanf("%d", &marks) total = total + marks; counter = counter + 1;} average = total / 5; while (++counter <= 5)
Control Structure EQ Repetition Structure AND Condition Type EQ While AND Test1 Grade EQ A AND Test2 Grade EQ B AND Test3 Grade EQ C AND Test4 Grade EQ D AND Test5 Grade EQ E	int marks, test, average, total = 0; for (test = 1, test <= 5, test++) printf("Enter marks") scanf("%d", &marks) total = total + marks; average = total / 5;

2.3.2.3 Functions

Modules in C are called functions [3]. Functions allow programmer to modularize a program. All variables declared in functions definitions are local variables. Most functions have a list of parameters. The parameters provide the means for communicating information between functions. A function's parameters are also local variables. The table below shows the predefined functions in math library [3].

Table 2.4: Examples of math library functions

Function	Description	Example
sqrt(x)	Square root of x	sqrt(900.0) is 30.0 sqrt(9.0) is 3.0
exp(x)	Exponential function e^x	exp(1.0) is 2.718282 exp(2.0) is 7.389056
Log(x)	Natural logarithm of x (base e)	Log(2.718282) is 1.0 Log(7.389056) is 2.0

The format of a function definition is:

return-value-type function name (parameter list)

```
{
    declarations
    statements
}
```

When user defines a function, it is called programmer-define function such as function maximum to determine and return the largest of three integers. The example of such function is as below:

```
int maximum (int x, int y, int z)
{
    int max = x;
```

```
if (y > max)
```

```
    max = y;
```

```
if (z > max)
```

```
    max = z;
```

```
return max;
```

```
}
```

2.3.2.3.1 Application of rules to teach function

Although there are many functions in C programming but only function maximum and minimum will provided in this system. This system is only to assist the users to know how the function works and useful to get the answer using numeric calculations.

Q: Problems?

A: Function

Q: Type of Function

A: Maximum

Q: Value variable A, B & C?

A: A= A, B= B & C= C

Algorithm

If Problem is Function

Then many types of Function

If type is Maximum

Then value for variable A and B

If B is B and value variable C is C

Then solution is code for maximum to

choose the maximum value between A, B

and C.

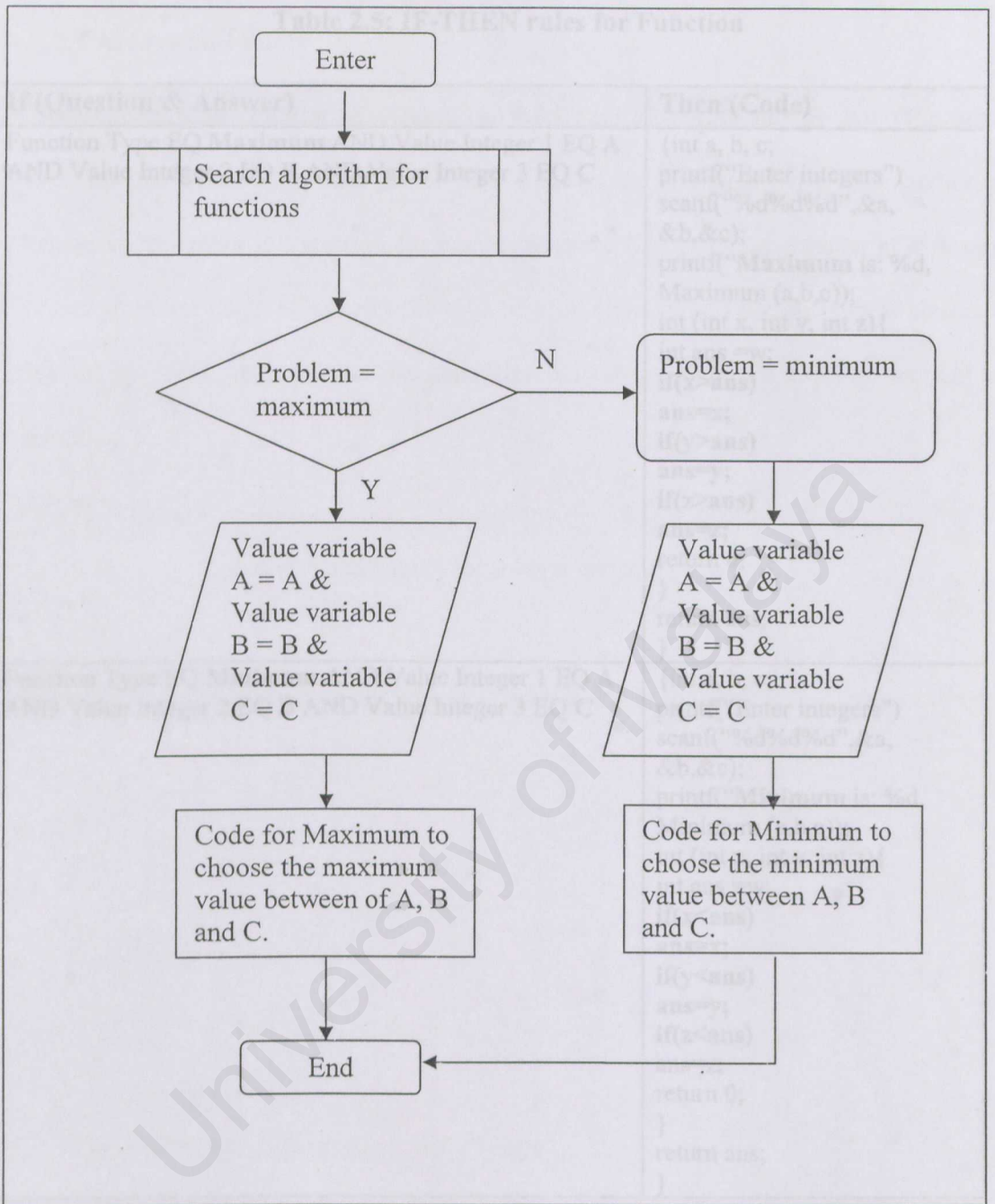


Figure 2.7: Flowchart to teach Functions

Table 2.5: IF-THEN rules for Function

If (Question & Answer)	Then (Code)
Function Type EQ Maximum AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C	<pre> {int a, b, c; printf("Enter integers") scanf("%d%d%d",&a, &b,&c); printf("Maximum is: %d, Maximum (a,b,c)); int (int x, int y, int z){ int ans =w; if(x>ans) ans=x; if(y>ans) ans=y; if(z>ans) ans=z; return 0; } return ans; } </pre>
Function Type EQ Minimum AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C	<pre> {int a, b, c; printf("Enter integers") scanf("%d%d%d",&a, &b,&c); printf("Minimum is: %d, Minimum (a,b,c)); int (int x, int y, int z){ int ans =w; if(x<ans) ans=x; if(y<ans) ans=y; if(z<ans) ans=z; return 0; } return ans; } </pre>

2.3.2.4 Arrays and Sorts

Array is a group of memory locations related by the fact that they all have the same name and the same type. The name of the array and the position of the particular element in the array is specified for easy reference. Arrays occupy space in memory.

The programmer needs to specify the type of each element and the number of elements required by each array so that the computer may reserve the appropriate amount of memory.

Example:

```
int c [10];    /* to reserve 10 elements for integer array c*/
```

Purpose: To initialize 10 element integer array n to zeros, and prints the array in tabular format.

```
main()
{
    int n[10], I;
    for (i = 0; i <= 9; i++)
        n[i] = 0;
    printf ("%s%13s\n", "Element", "Value");
    for (i=0; i <= 9; i++)
        printf ("%7d%13d\n", I, n[i]);
    return 0;
}
```

Sortings

Sorting is the process of arranging data in a given order. The content of an array may be used to arrange the elements in ascending or descending order. There are three types of sortings namely bubble sort, selection sort and recursive sort.

Example

Purpose: To sort an array of five elements using bubble sort. A list of five elements is arranged in ascending order.

Before sort: 90, 20, 80, 60, 10

After sort: 10, 20, 60, 80, 90

```
for (i = 0; i < 5; i++) {  
    if (num[i] > num[i+1]) {  
        temp = num[i];  
        num[i] = num[i+1];  
        num[i+1] = temp;  
    }  
}
```


2.3.2.4.1 Application of rules to teach arrays

Array can be easily taught to students using rule based system architecture based on the algorithm below.

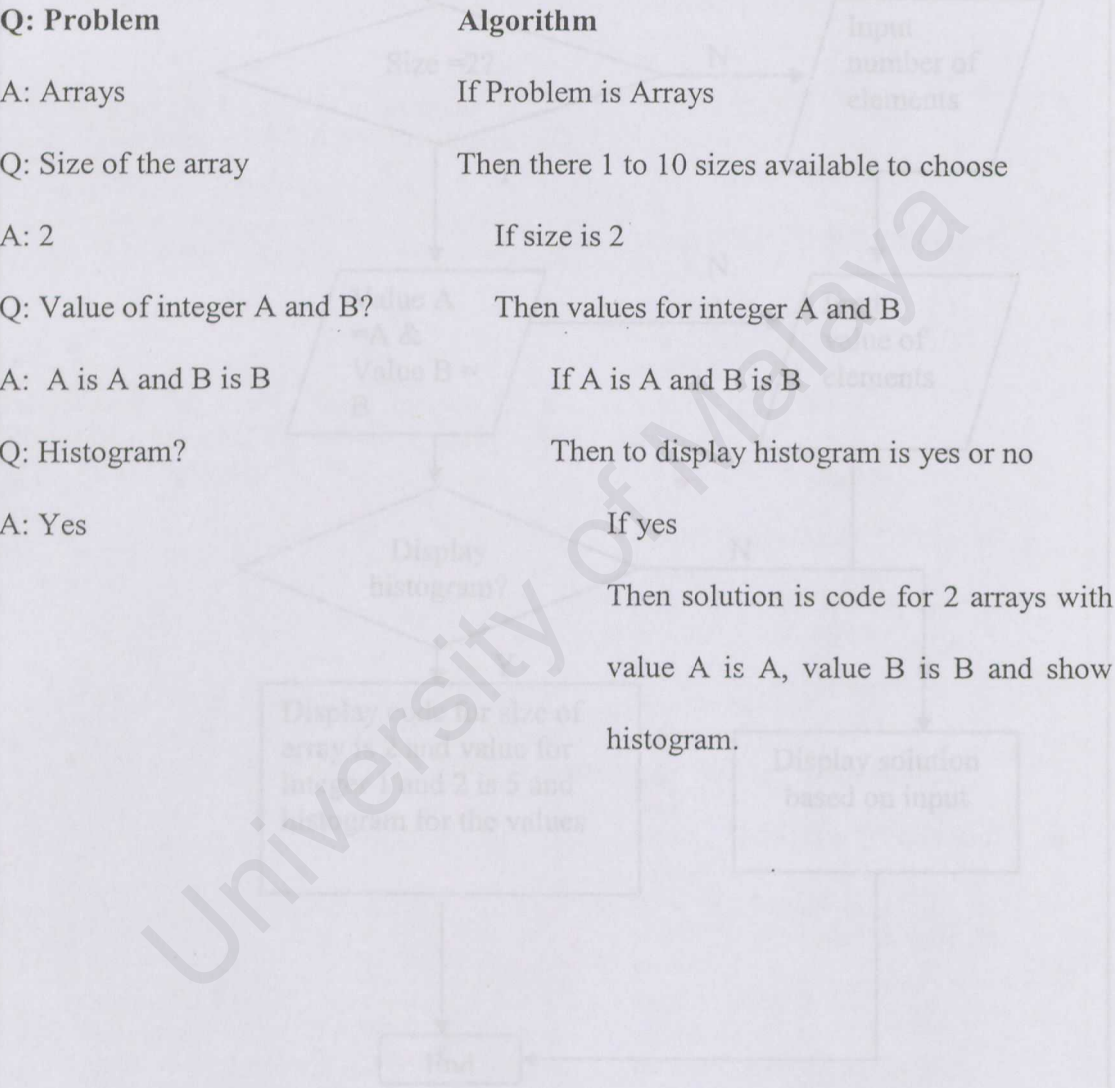


Figure 2.8: Flowchart to teach Arrays

Table 2.6: IF-THEN rules for Array

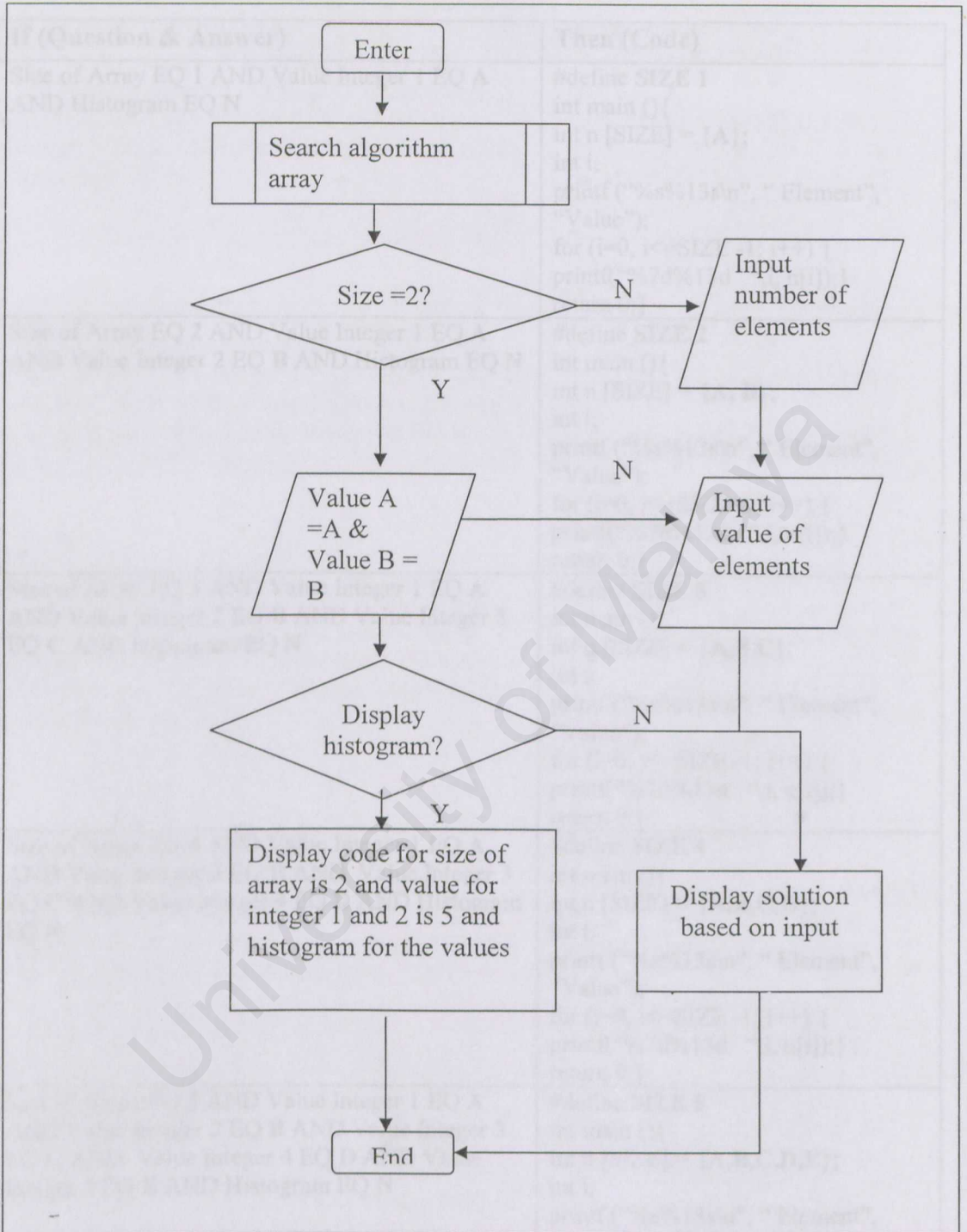


Figure 2.8: Flowchart to teach Arrays

Table 2.6: IF-THEN rules for Array

If (Question & Answer)	Then (Code)
Size of Array EQ 1 AND Value Integer 1 EQ A AND Histogram EQ N	<pre>#define SIZE 1 int main () { int n [SIZE] = {A}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 2 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Histogram EQ N	<pre>#define SIZE 2 int main () { int n [SIZE] = {A, B}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 3 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Histogram EQ N	<pre>#define SIZE 3 int main () { int n [SIZE] = {A,B,C}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 4 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Histogram EQ N	<pre>#define SIZE 4 int main () { int n [SIZE] = {A,B,C,D}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 5 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Histogram EQ N	<pre>#define SIZE 5 int main () { int n [SIZE] = {A,B,C,D,E}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>

Table 2.6, continued

If (Question & Answer)	Then (Code)
Size of Array EQ 6 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Histogram EQ N	<pre>#define SIZE 6 int main () { int n [SIZE] = {A,B,C,D,E,F}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 7 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Histogram EQ N	<pre>#define SIZE 7 int main () { int n [SIZE] = {A,B,C,D,E,F,G}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 8 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Value Integer 8 EQ H AND Histogram EQ N	<pre>#define SIZE 8 int main () { int n [SIZE] = {A,B,C,D,E,F,G,H}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 9 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Value Integer 8 EQ H AND Value Integer 9 EQ I AND Histogram EQ N	<pre>#define SIZE 9 int main () { int n [SIZE] = {A,B,C,D,E,F,G,H,I}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>
Size of Array EQ 10 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Value Integer 8 EQ H AND Value Integer 9 EQ I AND Value Integer 10 EQ J AND Histogram EQ N	<pre>#define SIZE 10 int main () { int n [SIZE] = {A,B,C,D,E,F,G,H,I,J}; int i; printf ("%s%13s\n", " Element", "Value"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); } return 0;}</pre>

Table 2.6, continued

If (Question & Answer)	Then (Code)
Size of Array EQ 1 AND Value Integer 1 EQ A AND Histogram EQ Y	<pre> #define SIZE 1 int main () { int n [SIZE] = {A}; int i; printf ("%s%13s\n", "Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') } return 0; } </pre>
Size of Array EQ 2 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Histogram EQ Y	<pre> #define SIZE 2 int main () { int n [SIZE] = {A,B}; int i; printf ("%s%13s\n", "Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') } return 0; } </pre>
Size of Array EQ 3 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Histogram EQ Y	<pre> #define SIZE 3 int main () { int n [SIZE] = {A,B,C}; int i; printf ("%s%13s\n", "Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') } return 0; } </pre>
Size of Array EQ 4 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Histogram EQ Y	<pre> #define SIZE 4 int main () { int n [SIZE] = {A,B,C,D}; int i; printf ("%s%13s\n", "Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') } return 0; } </pre>

Table 2.6, continued

If (Question & Answer)	Then (Code)
Size of Array EQ 5 AND Value Integer 1 EQ A Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Histogram EQ Y	<pre>#define SIZE 5 int main () { int n [SIZE] = {A,B,C,D,E}; int i; printf ("%s%13s\n", " Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') printf ("%c\n", '*') } return 0; }</pre>
Size of Array EQ 6 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Histogram EQ Y	<pre>#define SIZE 6 int main () { int n [SIZE] = {A,B,C,D,E,F}; int i; printf ("%s%13s\n", " Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') printf ("%c\n", '*') } return 0; }</pre>
Size of Array EQ 7 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Histogram EQ Y	<pre>#define SIZE 7 int main () { int n [SIZE] = {A,B,C,D,E,F,G}; int i; printf ("%s%13s\n", " Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') printf ("%c\n", '*') } return 0; }</pre>
Size of Array EQ 8 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Value Integer 8 EQ H AND Histogram EQ Y	<pre>#define SIZE 8 int main () { int n [SIZE] = {A,B,C,D,E,F,G,H}; int i; printf ("%s%13s\n", " Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", '*') printf ("%c\n", '*') } return 0; }</pre>

Table 2.6, continued

If (Question & Answer)	Then (Code)
Size of Array EQ 9 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Value Integer 8 EQ H AND Value Integer 9 EQ I AND Histogram EQ Y	<pre> #define SIZE 9 int main () { int n [SIZE] = {A,B,C,D,E,F,G,H,I}; int i; printf ("%s%13s\n", " Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", n[j]); printf ("%c\n", '*') } return 0; } </pre>
Size of Array EQ 10 AND Value Integer 1 EQ A AND Value Integer 2 EQ B AND Value Integer 3 EQ C AND Value Integer 4 EQ D AND Value Integer 5 EQ E AND Value Integer 6 EQ F AND Value Integer 7 EQ G AND Value Integer 8 EQ H AND Value Integer 9 EQ I AND Value Integer 10 EQ J AND Histogram EQ Y	<pre> #define SIZE 10 int main () { int n [SIZE] = {A,B,C,D,E,F,G,H,I,J}; int i; printf ("%s%13s\n", " Element", "Value", "Histogram"); for (i=0, i<=SIZE -1; i++) { printf ("%7d%13d ", i, n[i]); for (j=1, j<=n[i], j++) printf ("%c", n[j]); printf ("%c\n", '*') } return 0; } </pre>

2.3.2.5 Pointers

Pointers are variables that contain memory addresses as their values. Normally a variable directly contains a specific value whereas a pointer contains an address of a variable that contains a specific value. Pointers must be declared before they can be used such as in the example below:

Example:

```
int *countPtr, count;
```

The pointer operators are & and *. The & or address operator is a unary operator that returns the address of its operand. The example of algorithm is as given below:

```
int y = 5;
```

```
int *yPtr;
```

yPtr = &y; /* assigns the address of the variable y to pointer variable yPtr */

The * operator or commonly referred as the indirection operator or dereferencing operator, returns the value of the object to which its operand points.

Example:

```
printf ("%d", *yPtr); /* points the value of variable y */
```

```
main ()
```

```
{  
  
    int a;  
  
    int *aPtr;  
  
    a=7;  
  
    aPtr = &a;  
  
    printf ("The address of a is %p\n"  
"The value of aPtr is %p\n\n", &a, aPtr);  
  
    printf ("The value of a is %d\n"  
"The value of *aPtr is %d\n\n", a, *aPtr);  
  
    printf ("Proving that * and & are complements of"  
"each other.\n&*aPtr = %p\n*&aPtr = %p\n",  
    &*aPtr, *&aPtr);  
  
    return 0;  
}
```

2.3.2.5.1 Application of rules to teach pointers

Q: Problem	Algorithm
A: Pointers	If Problem is Pointers
Q: Types of pointer solutions	Then many types of pointer solutions
A: Call by Reference	If solution type is call by reference
Q: Variable value?	Then value of variable
A: A	If variable value is 5
	Then take algorithm for Pointer and call by reference for value variable is 5

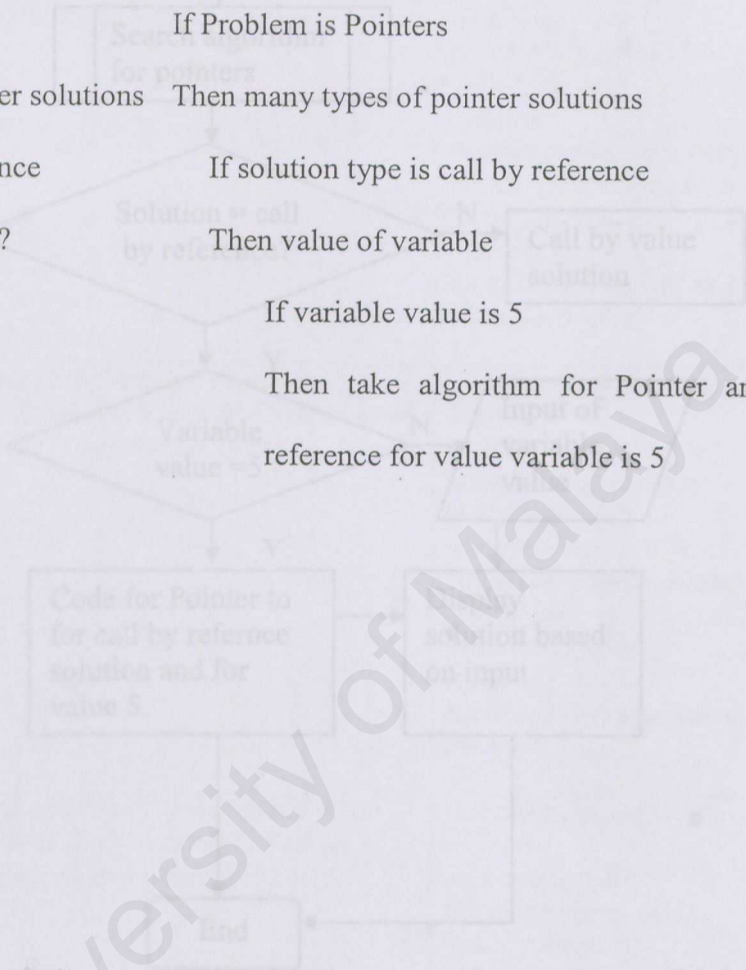


Figure 2.9: Flowchart to teach Pointers

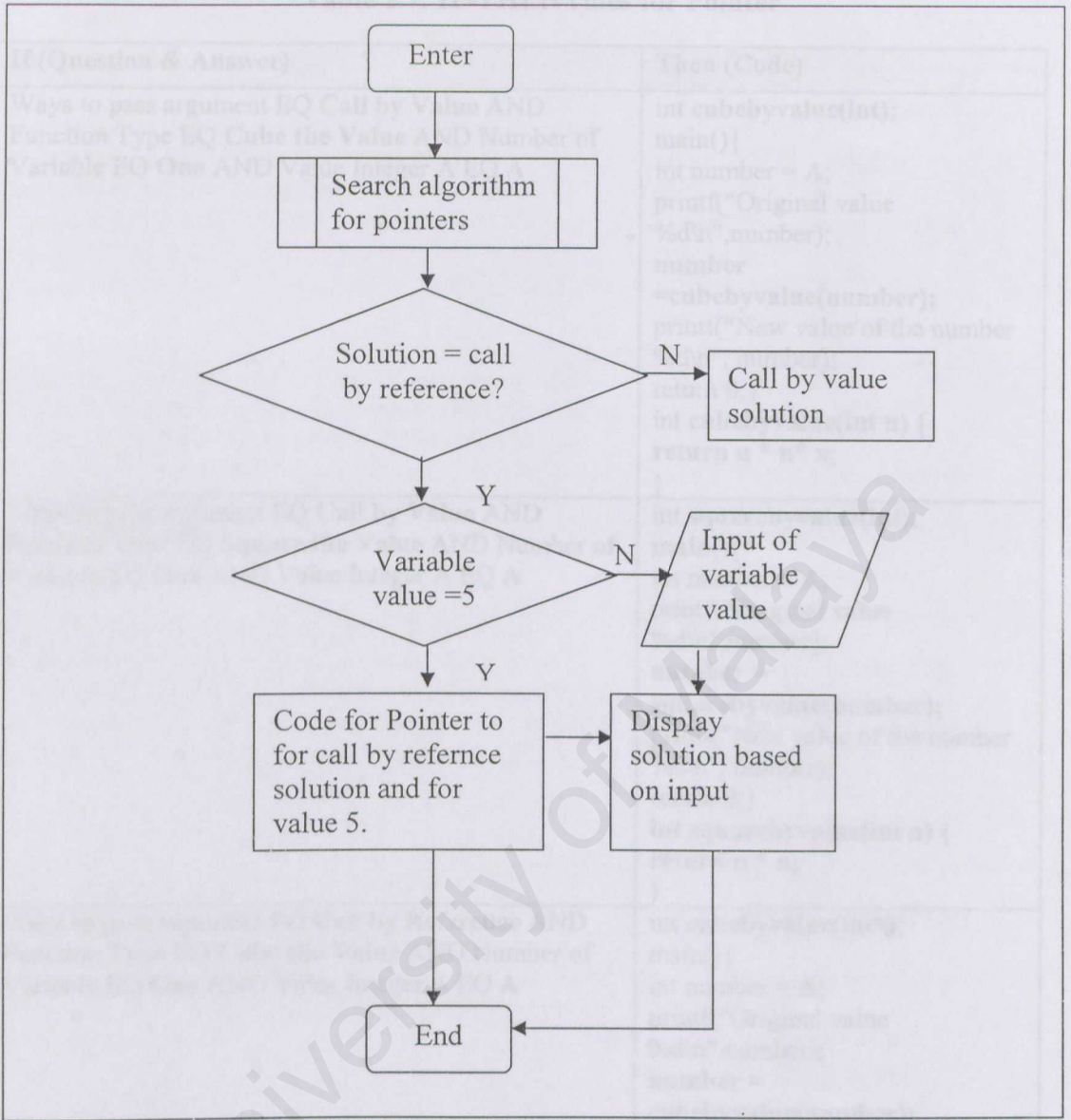


Figure 2.9: Flowchart to teach Pointers

Table 2.7: IF-THEN rules for Pointer

If (Question & Answer)	Then (Code)
Ways to pass argument EQ Call by Value AND Function Type EQ Cube the Value AND Number of Variable EQ One AND Value Integer A EQ A	<pre> int cubebyvalue(int); main() { int number = A; printf("Original value %d\n", number); number =cubebyvalue(number); printf("New value of the number %d\n", number); return 0;} int cubebyvalue(int n) { return n * n * n; }</pre>
Ways to pass argument EQ Call by Value AND Function Type EQ Square the Value AND Number of Variable EQ One AND Value Integer A EQ A	<pre> int squarebyvalue(int); main() { int number = A; printf("Original value %d\n", number); number = squarebyvalue(number); printf("New value of the number %d\n", number); return 0;} int squarebyvalue(int n) { return n * n; }</pre>
Ways to pass argument EQ Call by Reference AND Function Type EQ Cube the Value AND Number of Variable EQ One AND Value Integer A EQ A	<pre> int cubebyvalue(int*); main() { int number = A; printf("Original value %d\n", number); number = cubebyvalue(number); printf("New value of the number %d\n", number); return 0;} int cubebyvalue(int *nPtr) { return *nPtr * *nPtr * *nPtr; }</pre>
Ways to pass argument EQ Call by Reference AND Function Type EQ Square the Value AND Number of Variable EQ One AND Value Integer A EQ A	<pre> int squarebyvalue(int*); main() { int number = A; printf("Original value %d\n", number); number = squarebyvalue(number); printf("New value of the number %d\n", number); return 0;} int squarebyvalue(int *nPtr) { return *nPtr * *nPtr; }</pre>

Table 2.7, continued

If (Question & Answer)	Then (Code)
Ways to pass argument EQ Call by Reference AND Function Type EQ Cube the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 2 AND Value Integer A EQ A , Value Integer B EQ B AND Sort EQ N	<pre>#define SIZE 2 {int a [SIZE] = {A,B}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference AND Function Type EQ Square the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 3 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Sort EQ N	<pre>#define SIZE 3 {int a [SIZE] = {A,B,C}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Square the Value , Number of Variable EQ More Than One , Number of Variable EQ 4 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Sort EQ N	<pre>#define SIZE 4 {int a [SIZE] = {A,B,C,D}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Square the Value , Number of Variable EQ More Than One , Number of Variable EQ 5 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Value Integer E EQ E , Sort EQ N	<pre>#define SIZE 5 {int a [SIZE] = {A,B,C,D,E}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Square the Value , Number of Variable EQ More Than One , Number of Variable EQ 6 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Value Integer E EQ E , Value Integer F EQ F , Sort EQ N	<pre>#define SIZE 6 {int a [SIZE] = {A,B,C,D,E,F}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference AND Function Type EQ Square the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 7 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Value Integer D EQ D AND Value Integer E EQ E AND Value Integer F EQ F AND Value Integer G EQ G AND Sort EQ N	<pre>#define SIZE 7 {int a [SIZE] = {A,B,C,D,E,F,G}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>

Table 2.7, continued

If (Question & Answer)	Then (Code)
Ways to pass argument EQ Call by Reference AND Function Type EQ Square the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 8 , Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Value Integer D EQ D AND Value Integer E EQ E AND Value Integer F EQ F AND Value Integer G EQ G AND Value Integer H EQ H AND Sort EQ N	<pre>#define SIZE 8 {int a [SIZE] = {A,B,C,D,E,F,G,H}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Square the Value , Number of Variable EQ More Than One , Number of Variable EQ 9 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Value Integer E EQ E , Value Integer F EQ F , Value Integer G EQ G , Value Integer H EQ H , Value Integer I EQ I , Sort EQ N	<pre>#define SIZE 9 {int a [SIZE] = {A,B,C,D,E,F,G,H,I}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference AND Function Type EQ Square the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 10 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Value Integer D EQ D AND Value Integer E EQ E AND Value Integer F EQ F AND Value Integer G EQ G AND Value Integer H EQ H AND Value Integer I EQ I AND Value Integer J EQ J AND Sort EQ N	<pre>#define SIZE 10 {int a [SIZE] = {A,B,C,D,E,F,G,H,I,J}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;}</pre>
Ways to pass argument EQ Call by Reference AND Function Type EQ Cube the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 2 AND Value Integer A EQ A AND Value Integer B EQ B AND Sort EQ Y	<pre>#define SIZE 2 void bubbleSort(int*, const int) {int a [SIZE] = {A,B}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>

Table 2.7, continued

If (Question & Answer)	Then (Code)
Ways to pass argument EQ Call by Reference AND Function Type EQ Square the Value AND Number of Variable EQ More Than One AND Number of Variable EQ 3 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Sort EQ Y	<pre>#define SIZE 3 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Cube the Value , Number of Variable EQ More Than One , Number of Variable EQ 4 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Sort EQ Y	<pre>#define SIZE 4 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C, D}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Square the Value , Number of Variable EQ More Than One , Number of Variable EQ 5 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Value Integer E EQ E , Sort EQ Y	<pre>#define SIZE 5 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C,D,E}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>

Table 2.7, continued

If (Question & Answer)	Then (Code)
<p>Ways to pass argument EQ Call by Reference AND Function Type EQ Cube the Value, AND Number of Variable EQ More Than One AND Number of Variable EQ 6 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Value Integer D EQ D AND Value Integer E EQ E AND Value Integer F EQ F AND Sort EQ Y</p>	<pre>#define SIZE 6 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C, D,E,F}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>
<p>Ways to pass argument EQ Call by Reference AND Function Type EQ Cube the Value, AND Number of Variable EQ More Than One AND Number of Variable EQ 7 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C AND Value Integer D EQ D AND Value Integer E EQ E AND Value Integer F EQ F AND Value Integer G EQ G AND Sort EQ Y</p>	<pre>#define SIZE 7 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C,D,E,F,G}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>

Table 2.7, continued

If (Question & Answer)	Then (Code)
Ways to pass argument EQ Call by Reference , Function Type EQ Cube the Value , Number of Variable EQ More Than One , Number of Variable EQ 8 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Value Integer E EQ E , Value Integer F EQ F , Value Integer G EQ G , Value Integer H EQ H , Sort EQ Y	<pre> #define SIZE 8 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C, D,E,F,G, H}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;} </pre>
Ways to pass argument EQ Call by Reference , Function Type EQ Cube the Value , Number of Variable EQ More Than One , Number of Variable EQ 9 , Value Integer A EQ A , Value Integer B EQ B , Value Integer C EQ C , Value Integer D EQ D , Value Integer E EQ E , Value Integer F EQ F , Value Integer G EQ G , Value Integer H EQ H , Value Integer I EQ I , Sort EQ Y	<pre> #define SIZE 9 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C,D,E,F,G,H,I}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;} </pre>

Table 2.7, continued

If (Question & Answer)	Then (Code)
Ways to pass argument EQ Call by Reference AND Function Type EQ Cube the Value AND Number of Variable EQ More Than One , AND Number of Variable EQ 10 AND Value Integer A EQ A AND Value Integer B EQ B AND Value Integer C EQ C , Value Integer D EQ D AND Value Integer E EQ E AND Value Integer F EQ F AND Value Integer G EQ G AND Value Integer H EQ H , AND Value Integer I EQ I AND Value Integer J EQ J AND Sort EQ Y	<pre>#define SIZE 10 void bubbleSort(int*, const int) {int a [SIZE] = {A,B,C,D,E,F,G,H,I,J}; int i; printf("Data item in original order\n"); for(i=0, i<SIZE, i++) printf("%4d\n", a[i]); return 0;} void bubbleSort (int *array, const int size) void swap(int*, int*) int pass,j; for (pass =0; pass < size-1; pass++) if (array[j]> array[j+1]) swap (&array[j], &array[j+1]);} void swap (int *element1Ptr, int *element2Ptr) int hold = *element1Ptr; *element1Ptr= *element2Ptr; *element1Ptr = hold;}</pre>

2.3.2.6 File Processing

Storage of data in variables and arrays is temporary; all such data is lost when a program terminates. Files are used for programmer’s retention of large amounts of data. Computers store files on secondary storage devices, especially disk storage devices. Files can be created, updated and processed. This part will explain further on how to create a sequential file.

2.3.2.6.1 Application of rules to teach file processing

File processing can be taught easily using rule-based system architecture based on the algorithm below. Figure 2.10 explains how the system give the output of code based on the user selection and Table 2.8 depicts the if-then rules for file processing.

Q: Problem

Algorithm

A: File Processing

If Problem is File Processing

Q: Types of processes ?

Then many types of processes

A: Create A Sequential File

If process is create a sequential file

Q: Variable type to be written?

There are many variable types

A: Integer

If variable type is integer

Q: Variable name?

Then variable name

A: A

If variable name is A

Q: File name?

Then file name

A: test

If file name is test

Then code for create
sequential file and integer

A and file name is test.

Display Solution
based on input

Figure 1.38: Flowchart for File processing (Create Sequential File)

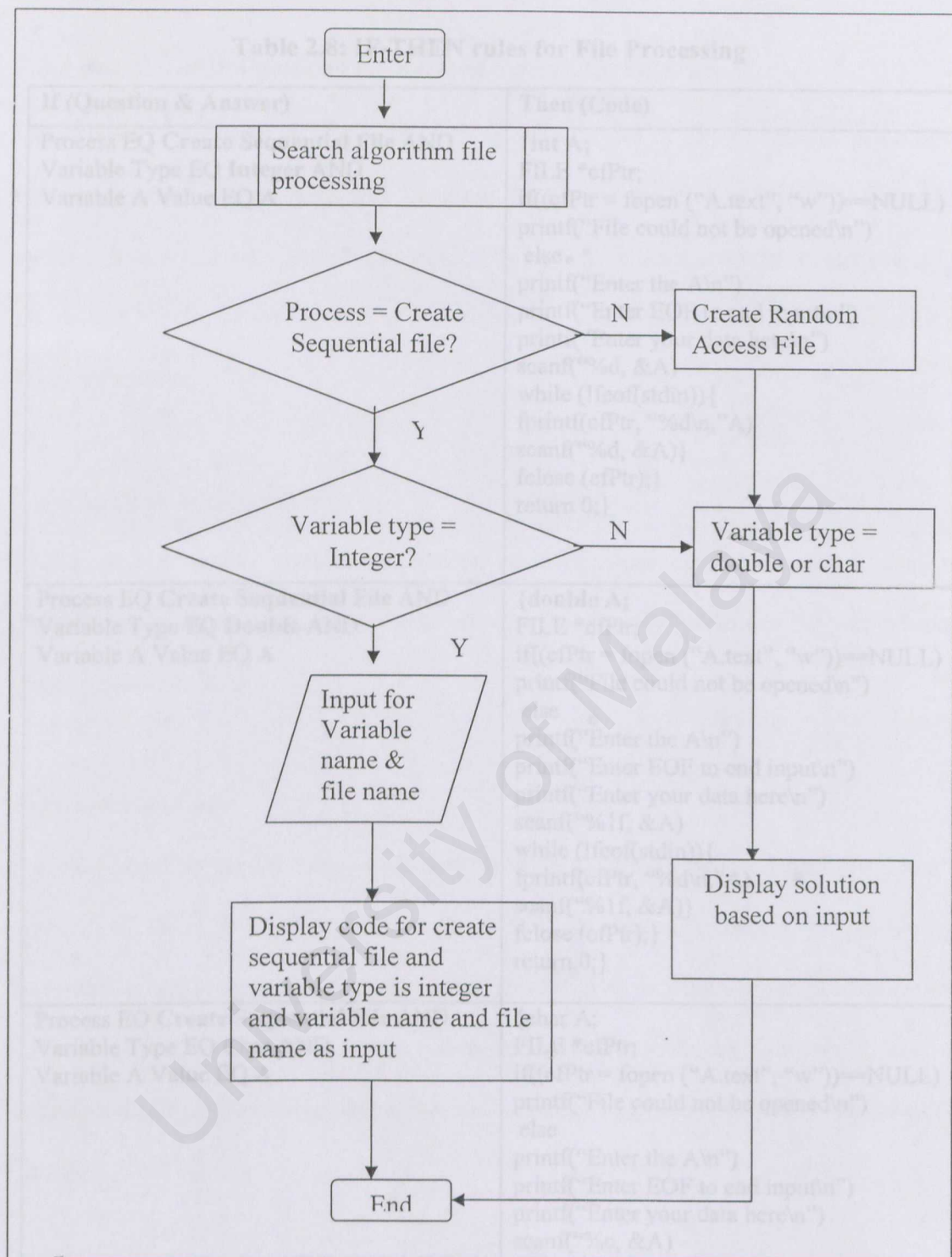


Figure 2.10: Flowchart to teach File processing (Create Sequential File)

Table 2.8: IF-THEN rules for File Processing

If (Question & Answer)	Then (Code)
Process EQ Create Sequential File AND Variable Type EQ Integer AND Variable A Value EQ A	<pre> {int A; FILE *cfPtr; if((cfPtr = fopen ("A.text", "w"))==NULL) printf("File could not be opened\n") else printf("Enter the A\n") printf("Enter EOF to end input\n") printf("Enter your data here\n") scanf("%d, &A) while (!feof(stdin)){ fprintf(cfPtr, "%d\n",A) scanf("%d, &A)} fclose (cfPtr);} return 0;}</pre>
Process EQ Create Sequential File AND Variable Type EQ Double AND Variable A Value EQ A	<pre> {double A; FILE *cfPtr; if((cfPtr = fopen ("A.text", "w"))==NULL) printf("File could not be opened\n") else printf("Enter the A\n") printf("Enter EOF to end input\n") printf("Enter your data here\n") scanf("%lf, &A) while (!feof(stdin)){ fprintf(cfPtr, "%d\n",A) scanf("%lf, &A)} fclose (cfPtr);} return 0;}</pre>
Process EQ Create Sequential File AND Variable Type EQ Char AND Variable A Value EQ A	<pre> {char A; FILE *cfPtr; if((cfPtr = fopen ("A.text", "w"))==NULL) printf("File could not be opened\n") else printf("Enter the A\n") printf("Enter EOF to end input\n") printf("Enter your data here\n") scanf("%c, &A) while (!feof(stdin)){ fprintf(cfPtr, "%d\n",5) scanf("%c, &A)} fclose (cfPtr);} return 0;}</pre>

2.3.3 Benefits of Designing in Rule-Based System Architecture

Ctutorial4u is designed in rule-based system architecture in order to develop the best tutoring system where there will be input from user to retrieve solution from the system. It is a two-way learning process. The system is designed with respect to user interaction where the system acts as a guide for the students to excel in C programming for novice users.

2.3.4 Analysis and Synthesis

Based on the expert system explanation and its components, rule based system architecture is chosen to develop Ctutorial4u as it is closely resembles the way human experts solve problem.

Forward chaining

Forward chaining can provide a considerable amount of information from only a small amount of data. It works well when a problem naturally begins by gathering information and seeing what can be inferred from it. This is what needed in Ctutorial4u where it is more feasible to begin a session with data given by the user to derive conclusion rather than beginning a session by trying to prove a present conclusion is valid.

Monotonic Reasoning

Monotonic reasoning is the best reasoning technique for Ctutorial4u since the facts remain unchanged in the database for the problems. The rules are set in the inference engine to for look appropriate solution.

2.4 Surveys on existing system

Thousands of expert systems have been constructed in the last few years. Expert systems have been applied in many areas, such as business, chemistry, education, finance, law, mathematics, medicine, mining and space technology. They are used for control, design, diagnosis, prediction, planning, simulation etc. Thousands of systems have been developed and are in use throughout the world. Expert systems have moved from the research labs to the general market place and industry. This has resulted from the better understanding of the technology and the production of tools for building such systems. The Table 2.9 shows some of the early expert system constructed.

Table 2.9: Example of expert systems

Expert system	Description
DENDRAL	A system developed at Stanford to interpret mass spectrograms.
Drilling Advisor	A system developed by Elf to help determine why a drill sticks.
MYCIN	A medical diagnosis system.
XCON	A computer configuration system developed by DEC for VAX computers.
LENDING ADVISOR	Used for evaluating the risks on possible loans.
PROUST	For finding semantic bugs in novice Pascal programmers' code.

We need to analyze some similar system to get some ideas, knowledge and guidance. So, there are few system being analyzed and compared to produce a system which meets the requirements of users.

2.4.1 MYCIN

Mycin was an expert system developed at Stanford in the 1970s using Lisp. It is a program for advising physicians on treating bacterial infections of the blood and meningitis. MYCIN conducts a question and answer dialog. After asking basic facts about the patient such as name, sex and age, MYCIN asks about suspected bacterial organisms, suspected sites of infection, the presence of specific symptoms such as fever or headache which is relevant to diagnosis. It then recommends a certain course of antibiotics. MYCIN's dialogs are in English so it avoids having to understand freely written English by controlling the dialog. It outputs sentences, but the user types only single words or standard phrases. Its major innovations over many previous expert systems were that it uses measures of uncertainty (not probabilities) for its diagnoses and the fact that it is prepared to explain its reasoning to the physician, so he can decide whether to accept it.

MYCIN extended the notion that the knowledge base should be separate from the inference engine, and its rule-based inference engine was built on a backward-chaining or goal-directed control strategy. Since it was designed as a consultant for physicians, MYCIN was given the ability to explain both its line of reasoning and its knowledge. Mycin represented its knowledge as a set of IF-THEN rules with certainty factors. The following is an English version of one of Mycin's rules:

IF the infection is primary-bacteremia

AND the site of the culture is one of the sterile sites

AND the suspected portal of entry is the gastrointestinal tract

THEN there is suggestive evidence (0.7) that infection is bacteroid.

The 0.7 is roughly the certainty that the conclusion will be true given the evidence. If the evidence is uncertain, the bits of certainties of evidence will be combined with the certainty of the rule to give derive a conclusion. The action part of the rule could just be a conclusion about the problem being solved, or it could be an arbitrary lisp expression. This allowed great flexibility.

Figure 2.11: Main menu of rule-based expert system

MyCin use the basic backward chaining reasoning strategy that we described above. However, MyCin used various heuristics to control the search for a solution or proof of some hypothesis. These helped to make the reasoning efficient and to prevent the user being asked too many unnecessary questions.

Though it's an expert system for medical field and not for education or learning, but MYCIN act as a premier or role-model to rely on for all future expert systems development. The rules IF- THEN of MYCIN undoubtedly are the guidance for CTutorial4u.

2.4.2 The Rule-Based Expert System Using an Interactive QA Sequence

This rule-based system consists largely of a main window system (knowledge acquisition module, rule-based inference engine, and user interface for input/output), a simple GIS mapping system, and a database [2]. The knowledge acquisition module, inference engine, and user interface were built using Visual Basic 6.0. The GIS mapping system was also built using Visual Basic 6.0 based on MapObjects Version 2. MapObjects has many GIS facilities and can be extended and integrated easily with other systems using a conventional language such as Visual C++ and Visual Basic. The database for the knowledge base (prologue, questions and answers, and rules) is designed in Microsoft Access 2002.



Figure 2.11: Main menu of rule-based expert system

Expert main menu: Consists of three sub menus: new problem, open problem. The expert part is the knowledge acquisition module. The expert part can be used to construct new knowledge (prologue, question, answer, and rule) or to update the existing knowledge by experts.

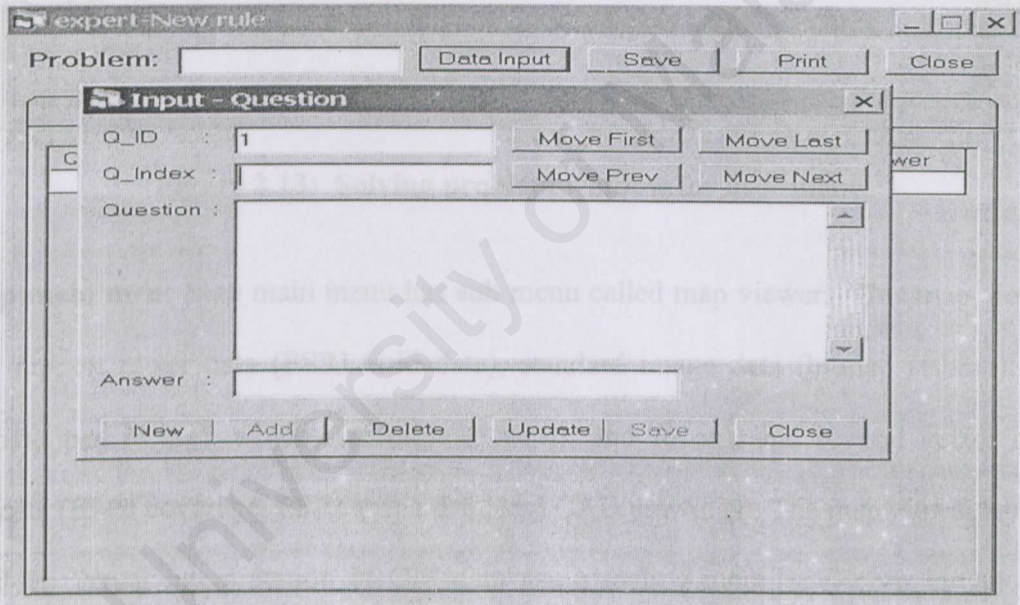


Figure 2.12: Expert menu's input window

User main menu: The rule-based inference engine was implemented in the user part because the inference rule and the search strategy are needed in the user part in order to solve a selected problem. This inference engine is built based on deductive reasoning using forward chaining. The inference engine will generate questions automatically when the user selects a problem. The solution will also be generated from the previous questions and answers using the existing rules by the rule-based inference engine.

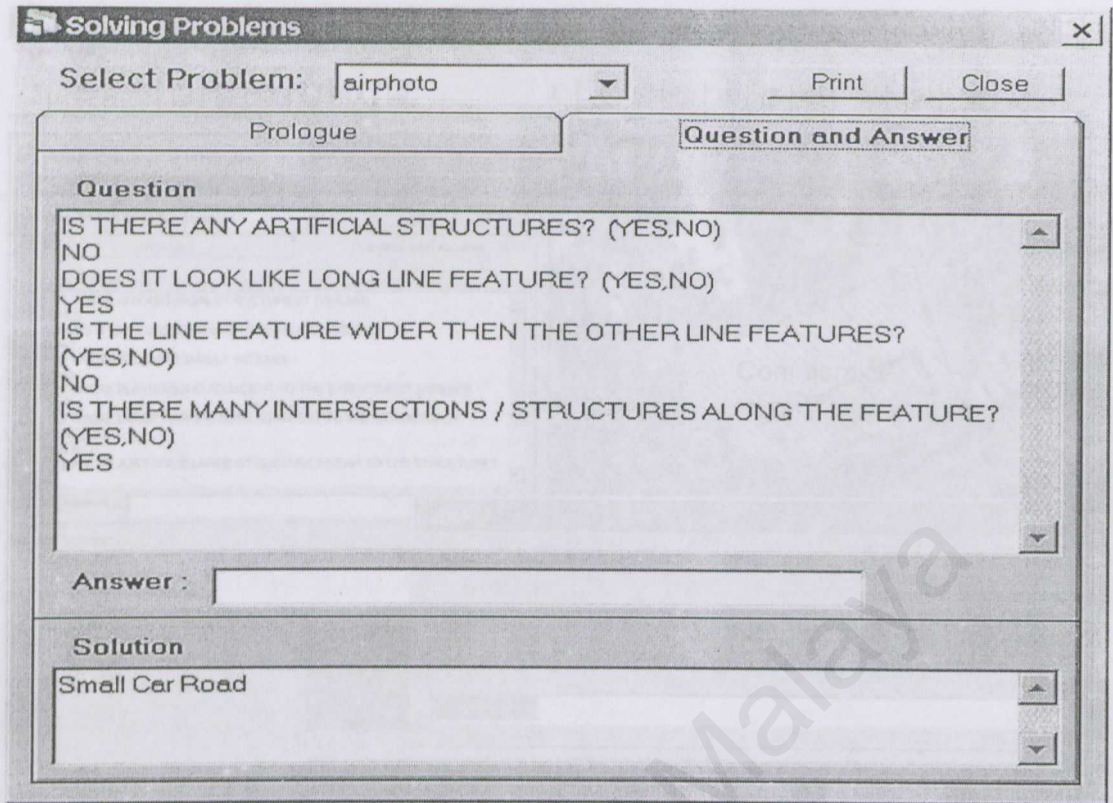


Figure 2.13: Solving problem window for user menu

Map main men: Map main menu has sub menu called map viewer. This map viewer can support raster data (ESRI Grid data), standard image data (bitmap (*.bmp), gif (*.gif), jpeg (*.jpg), window metafile (*.wmf), and so on), and several vector data formats (ESRI coverage, ESRI shape data, and CAD drawings). The map viewer can be used for visual interpretation of GIS data using scaling modules (Zoom In/Out and Pan). The map viewer also can be used for the thematic mapping. The thematic mapping process has four steps. First, the user selects features on the existing image or map. Second, the user draws polygons for the boundary of the selected features. Third, the user interprets the selected features using the user part in the expert system. Last, the user labels the polygon for selected features using labeling module when the expert system generates the solution.

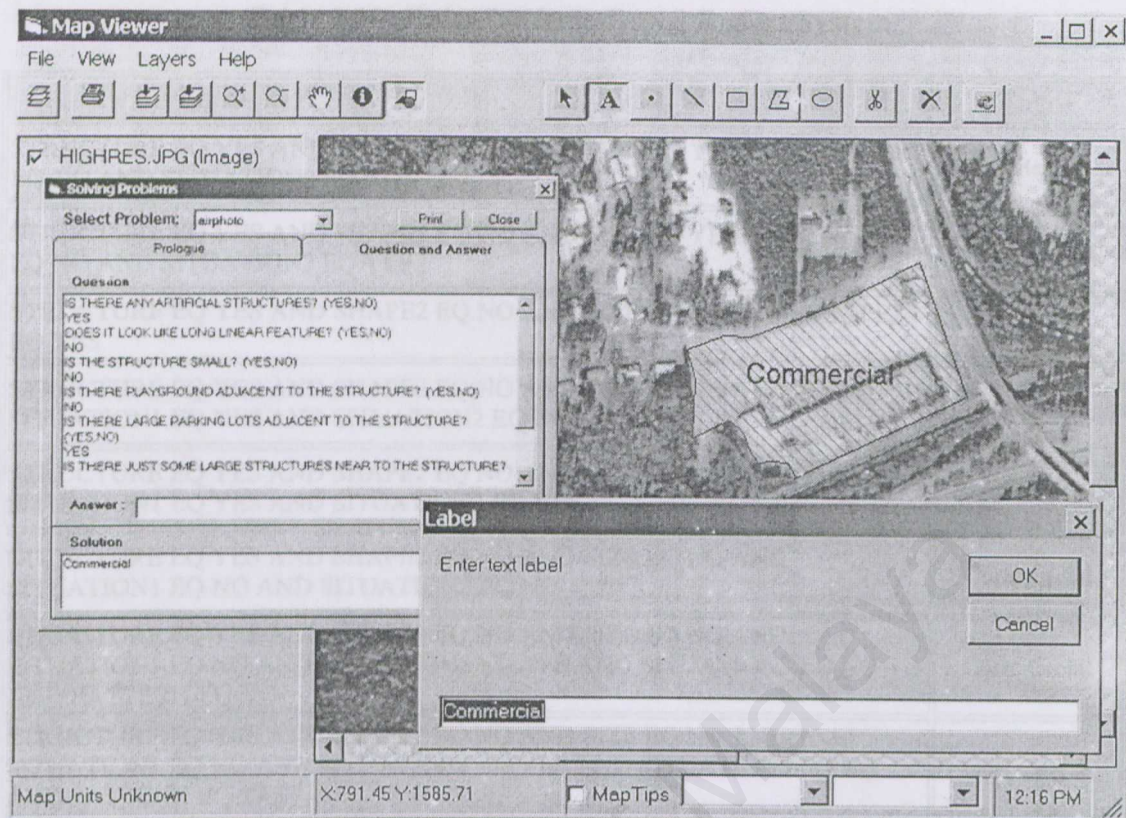


Figure 2.14: Map Viewer of Rule-Based Expert System

STRUCTURE EQ YES AND SHAPE EQ NO AND SIZE EQ NO AND SITUATION EQ YES	Residential (Apartment)
STRUCTURE EQ YES AND SHAPE EQ NO AND SIZE EQ NO AND SITUATION EQ NO	Industrial
STRUCTURE EQ YES AND SHAPE EQ YES AND SITUATION EQ YES	Highway
STRUCTURE EQ YES AND SHAPE EQ YES AND SITUATION EQ NO	Railroad
STRUCTURE EQ YES AND SHAPE EQ YES AND SITUATION EQ YES	Small Car Road
STRUCTURE EQ YES AND SHAPE EQ YES AND SITUATION EQ YES	Golf Course
STRUCTURE EQ YES AND SHAPE EQ YES AND SITUATION EQ YES	Lake
STRUCTURE EQ YES AND SHAPE EQ YES AND SITUATION EQ YES	Swamp

The Rule-Based Expert System discussed earlier actually is quite similar to CTutorial4u. The If-Then rules in Table 2.10 actually are guideline to develop the rules for CTutorial4u. This system uses interactive question and answer session of YES and NO answers to derive a solution. There are too many ways to derive solution.

Table 2.10: Rules for Rule-Based Expert System

If	Then
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ YES AND SHAPE1 EQ NO AND SITUATION2 EQ NO	Residential
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ YES AND SHAPE1 EQ NO AND SITUATION2 EQ YES	Commercial
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ YES AND SHAPE1 EQ YES	Industrial
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ YES AND SITUATION2 EQ NO	School
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ YES AND SITUATION2 EQ YES	Service (park)
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ NO AND SITUATION2 EQ NO	Commercial
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ NO AND SITUATION2 EQ YES AND SITUATION3 EQ NO AND PATTERN EQ NO	Commercial
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ NO AND SITUATION2 EQ YES AND SITUATION3 EQ NO AND PATTERN EQ YES	Golf Course
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ NO AND SITUATION2 EQ YES AND SITUATION3 EQ YES AND SITUATION4 EQ NO	Residential (Apartment)
STRUCTURE EQ YES AND SHAPE2 EQ NO AND SIZE EQ NO AND SITUATION1 EQ NO AND SITUATION2 EQ YES AND SITUATION3 EQ YES AND SITUATION4 EQ YES	Industrial
STRUCTURE EQ YES AND SHAPE2 EQ YES AND SHAPE3 EQ YES	Highway
STRUCTURE EQ YES AND SHAPE2 EQ YES AND SHAPE3 EQ NO AND SITUATION5 EQ NO	Railroad
STRUCTURE EQ YES AND SHAPE2 EQ YES AND SHAPE3 EQ NO AND SITUATION5 EQ YES	Small Car Road
STRUCTURE EQ NO AND PATTERN EQ YES	Golf Course
STRUCTURE EQ NO AND PATTERN EQ NO AND SHAPE4 EQ YES	Lake
STRUCTURE EQ NO AND PATTERN EQ NO AND SHAPE4 EQ NO AND TONE EQ WHOLEY	Evergreen

The Rule-Based Expert System discussed earlier actually is quite similar to CTutorial4u. The If-Then rules in Table 2.10 actually are guideline to develop the rules for CTutorial4u. This system uses interactive question and answer session of YES and NO answers to derive a solution. There are so many ways to derive solution.

CTutorial4u can't use question and answer with YES and NO answer but it is designed to use question and answer with selection criteria. The answers are set in the rules so the users can only select the answers specified.

2.4.3 INTELLITUTOR II

An Intelligent programming environment for learning programming is an interactive application of knowledge-based Artificial Intelligence [11]. INTELLITUTOR II consists of GUIDE, ALPUS II and a C environment as an integrated environment, and is implemented as a server-client system on the Internet. The server system consists of a Web server, an ALPUS II server and a C server [10]. The service includes reading C documents, editing C programs with guide and help functions, detecting logical bugs by means of a knowledge-based program understander ALPUS II, compiling the program and executing it with the C environment. ALPUS is a knowledge-based program understander by means of four kinds of programming knowledge on program semantics, which are knowledge on algorithms, knowledge on programming techniques, knowledge on variables and knowledge on bugs. The knowledge on a programming language is used as well as a base level understanding.

Knowledge Modeling in ALPUS II

In ALPUS II the algorithm-oriented programming knowledge plays a key role in understanding a buggy program. This knowledge is represented in a hierarchical data structure called HPG.

Program Understanding in ALPUS II

Program understanding in APLUS II is done by four major steps such as [9]:

Step 1: Generalization of program statements is done first, mainly by introducing the language independent representation of program statements in AL (abstract language). AL was designed to represent source statements in both Pascal and C, so that knowledge-based program comprehension can be done by means of the common knowledge base in which every piece of the knowledge is represented in the AL formalism. Student's program is translated to AL statements first in this step. For example, an assignment statement is represented as “(<- Left-variable Expression)”, and conditional loops are represented as “(?Loop Expression Statement)” for a so-called pre-check loop for a While-Do statement in Pascal and a While statement in.

Step 2: Normalization of program segments is done next by five steps to decrease a variety of program code. At first, a source program in AL is converted to a Lisp-like representation with line number. Next, order of relational operators is normalized so that the evaluation can be easier. Simplification of a program structure is done next. For example, meaningless procedures are detected and inserted into calling program by modification of code and variables. Deletion of redundant type and constant declarations is done next. Deletion of the RECORD-type declarations is done lastly.

Step 3: Identification of key variables and their roles is done next based on the data obtained by Cognitive experiment as shown in the following procedure. At first rough segmentation of the source program is done by means of HPG-oriented algorithm knowledge. Then, likelihood scores are evaluated by means of the associated attributes attached to each key variable stored within the knowledge base, where the role and locations

within the algorithm are defined. The variable which has been assigned with the highest score is known as the identified variable and its role. The instance frame is generated from the associated class frame for each identified variable. It should be stressed that in this process bugs are not considered. This strategy works very well even if the program includes bugs and incompleteness.

Step 4: Identification of each process in detail is done next as the final step. The logical bugs and associated intentions are identified in this step by pattern matching techniques between the student's program segments and the template knowledge. The pattern matching is done against each node of the HPG graph from the root to the leaves one-by-one as follows:

- 1) One process is picked up from 5 the HPG nodes.
- 2) The template is generated from the attached knowledge and associated identified variables.
- 3) The pattern matching is applied to an associated segment of the target student's program to detect correctness or buggyness. The standard pattern is tried first. If the template matches then the system understands this segment as correct. If failed, then acceptable patterns are tried one by one. If one of the acceptable patterns has matched then the system supposes that although this segment would work well better coding should be used instead. If failed again, then buggy patterns are tried. If one of them matched then the system supposes this as a buggy segment. If failed, then the system supposes this segment as buggy one. However any advice can not be produced since information is not available within the knowledge base.

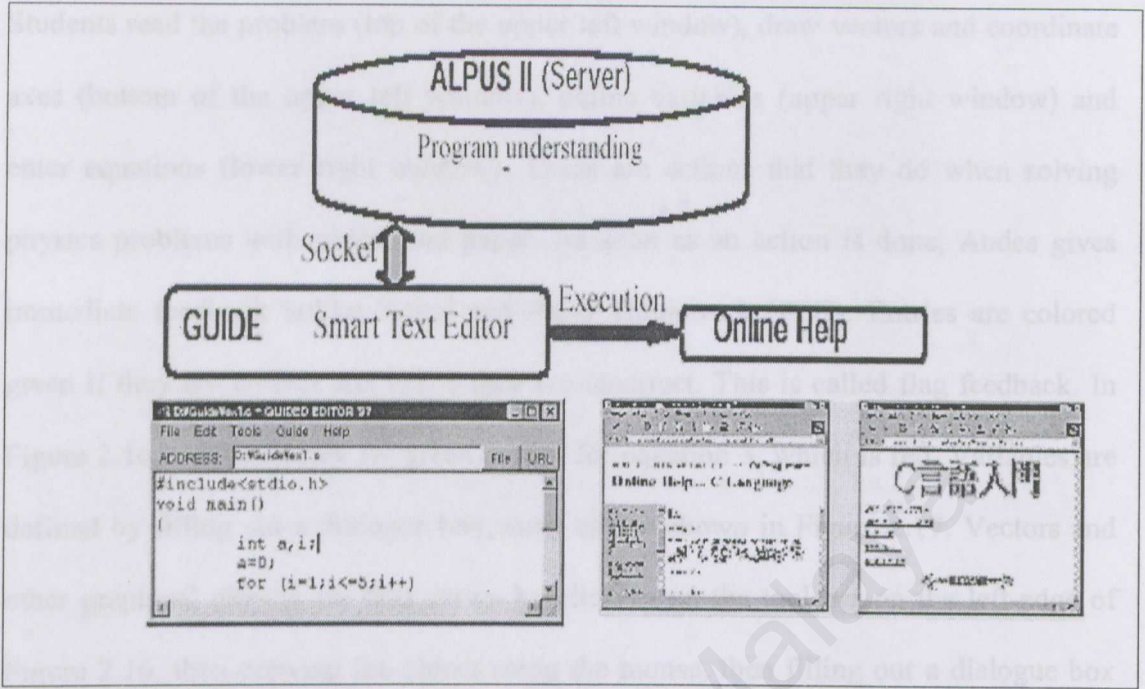


Figure 2.15: Overview of the APLUS II

2.4.4 The ANDES Physics Tutoring System

The Andes project originated with an Office of Naval Research management initiative to forge close relationships between ONR and the Navy's academic institutions. In particular, there was an interest in trying out artificially intelligent tutoring technology, a longstanding research area for ONR, at the Naval Academy. The Andes system is an intelligent tutoring system that has helped hundreds of students to improve their learning for physics in university. It replaces pencil and paper problem solving homework. Students continue to attend the same lectures, labs and recitations. Five years of experimentation at the United States Naval Academy indicates that it significantly improves student learning [7]. This report is a comprehensive description of Andes. It describes Andes' pedagogical principles and features, the system design and implementation, the evaluations of pedagogical effectiveness, and our plans for dissemination.

Students read the problem (top of the upper left window), draw vectors and coordinate axes (bottom of the upper left window), define variables (upper right window) and enter equations (lower right window). These are actions that they do when solving physics problems with pencil and paper. As soon as an action is done, Andes gives immediate feedback unlike Pencil and Paper Homework (PPH). Entries are colored green if they are correct and red if they are incorrect. This is called flag feedback. In Figure 2.16, all the entries are green except for equation 3, which is red. Variables are defined by filling out a dialogue box, such as one shown in Figure 2.17. Vectors and other graphical objects are first drawn by clicking on the tool bar on the left edge of Figure 2.16, then drawing the object using the mouse, then filling out a dialogue box like the one in Figure 2.17. Filling out these dialogue boxes forces students to precisely define the semantics of variables and vectors. PPH does not require this kind of precision, so students often just use variables in equations without defining them. If students include an undefined variable in an Andes equation, the equation turns red and a message box pops up indicating which variable(s) are undefined.

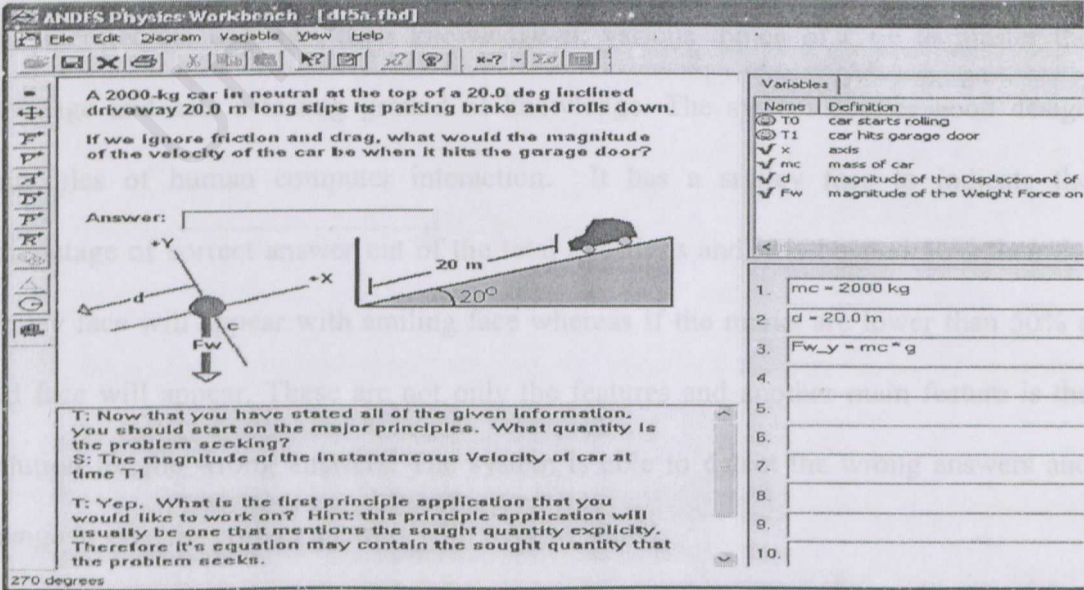


Figure 2.16: Andes screen

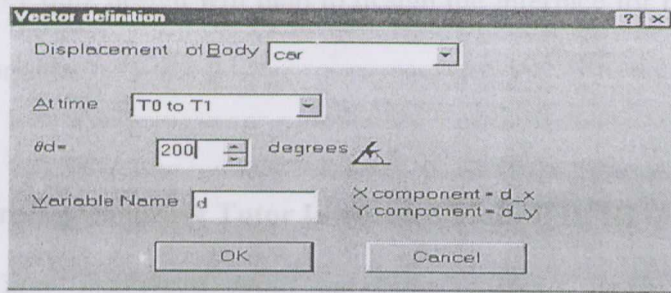


Figure 2.17: A dialogue box for drawing a vector

2.4.5 Softsys

Softsys provides software services as well as training courses. It is an online system. The C++ Quiz is used heavily from visitors worldwide which consists of short (12 questions) or detailed (55 questions).

The main feature of the system is the C++ quiz is which has two portion short quiz and detailed quiz. Basically here, the idea and design of the quiz is being evaluated to get better idea how the new system can be designed. The quiz is designed as a collection of question from various topics in C++ language and it is not based on topics which it is not a good learning approach provided to naïve users but it is a good practice for those who are in the intermediate stage. The intention of the system is to provide chance to the intermediate user test their knowledge in various topics of C++ to master the language and acts a testing ground of knowledge. The system applies good design principles of human computer interaction. It has a smiley face to indicate the percentage of correct answer out of the total questions and if is beyond 50% then the smiley face will appear with smiling face whereas if the marks are lower than 50% a sad face will appear. These are not only the features and another main feature is the solution for the wrong answers. The system is able to detect the wrong answers and manage to give the correct answers with reasons stated.

The principles of C++ quiz design will help to design the interface for new rule-based system for the Ctutorial4u.

2.4.6 Carnegie Learning Cognitive Tutor Integrated Math I, II, III

Using Cognitive Tutor Integrated Math curricula, students work with multiple representations of a linear function: tables, graphs, algebraic formulas and written text. By translating from one representation to another, students gain a solid understanding of how the representations interconnect [2]. Cognitive Tutor let the students learn in an environment that is similar to working one-on-one with an instructor. Each student benefits from an individualized course of instruction complete with immediate feedback, prescriptive mediation and assessment.

- **Problem Scenario:** The Problem Scenario helps students make connections between real-life problem situations and the mathematics needed to solve them. Embedded questions provide breadth and relevance to the problem-solving process.
- **Worksheet:** As students progress through the curriculum, they learn to generalize specific instances into mathematical formulas. Students complete the Worksheet (which functions like a spreadsheet) by recording answers to questions posed in the Problem Scenario.
- **Solver:** The Solver encourages students to solve equations within the context of the problem. Students learn techniques to solve problems and discover the value of mathematical skills beyond the classroom.
- **Graph:** Students represent mathematical functions graphically, set boundaries and intervals, label axes and plot points and lines.

Just-in-Time Help Messages: When students make errors, they receive immediate feedback. This gives students the opportunity to correct mistakes quickly. Teachers can spend more time with students who need additional help, confident that all students are engaged and on task.

Skills: The Cognitive Tutor dynamically assesses and tracks each student's progress and level of understanding on specific mathematical skills. As the Tutor guides them down an individualized learning path, students can access this information on demand, which encourages them to be accountable for their own learning progress.

Though it is not designed for C programming language but it has some basic good principles of designing a tutorial system for the students such the problem scenario, worksheets, solver, just-in time help messages and skills as mentioned above. The only drawback of the system is it is not designed in a knowledge based system where **if then rule** statement was not being used. The suggestion to this system is to use rule-system architecture to enhance it functionality, modifiability and scalability.

2.5 CTutorial4u Vs Existing Systems

The analysis of other system is very important to develop a new system although the existing system does not perform all the functionalities as the new system but actually the new system will adopt all the good strategies in the similar existing system and try to eliminate all the lacking in terms of design and functionalities.

Table 2.11: Comparison of existing system's and functionalities

System Functionalities	MYCIN	Rule- Based Expert System	INTELL ITUTO R	ANDES	Softsys	Cognitiv e Tutor
If-Then rules	Y	Y	N	N	N	N
Tutorial	N	N	Y	Y	Y	Y
Quiz	N	N	N	N	Y	Y
Help	Y	Y	Y	Y	Y	Y
Human computer interaction metaphor	Y	Y	Y	Y	Y	Y
Central database	Y	Y	Y	Y	Y	Y
User friendliness	N	Y	Y	Y	Y	Y
Consistency	Y	Y	Y	Y	Y	Y
Security	Y	Y	Y	Y	Y	Y
Immediate feedback	Y	Y	Y	Y	Y	Y
Program compiler/solver	Y	Y	Y	Y	Y	Y

The table above explains that all the systems functionalities were equally contributed to the system design of CTutorial4u. All these systems actually lacking at least one of the functionalities but the Ctutorial4u complying all the above functionalities and attempts to give the best to the users by adopting all good behaviors. The non rule-based expert systems such as Softsys and Cognitive Tutor didn't use If-Then rules such as INTELLITUTOR and ANDES (intelligent systems) whereas Softsys and Cognitive Tutor (normal online systems). Human computer metaphor is one of the interactivity features needed by expert system which complied by Rule-Based Expert System Using Interactive Question and Answer Sequence, ANDES, Softsys and Cognitive Tutor such as using the traffic light and smiley concepts in quiz session. All the systems uses central database and all has consistent interface throughout the whole system. These systems are user friendly where users don't find it difficult to use the systems the second time they login in except MYCIN needs the expert in that field who are in medical line to use because mostly uses medical terms. In terms of security, all the

system allows the administrators to amend the database or the design of the system where all are well protected with password.

After analyzing all the above stated system's functionalities, the Ctutorial4u includes just-in time help messages from Cognitive Tutor to indicate the percentage of correct answer out of the total questions and if is beyond 50% then the message excellent will appear with smiley whereas if the marks are lower than 50% then a try again message will appear with sad smiley. Ctutorial4u basically adopts this feature from Cognitive Tutor. Apart from that, CTutorial4u not only act as learning software but it is more like a real tutor where based on the user selection for every chapter, the system somehow solve the user directed problems using rules in the working memory. The memory contains all the possible solution for every topic in C.

On top of that, CTutorial4u also include quiz for every chapter to test the student's knowledge about a particular topic and store the respective student's marks in the database and quiz session applies the human computer metaphor adopted from Cognitive Tutor. All the systems compared above, provide immediate feedback to students and not like normal procedures in classroom or hospital where we got to wait till the lecturer mark our papers and doctors to diagnose the sickness. The only drawback of CTutorial4u is it does provide C compiler within the system itself for the student to test their programming skills or ability to write codes on their own where with it the system actually can correct their errors by debugging their codes.

2.6 Software and Technologies

Visual Basic 6.0 will be used to design the user interfaces and Microsoft Access 2000 to design the database of CTutorial4u.

2.6.1 Visual Basic

Visual Basic 6.0 is one the programming language used to develop most of the standalone software. It is based on graphical and event-driven user interface where an object can be built easily sing interface and the codes are easy to built as well. Actually event processor controls Visual Basic and nothing will happen until the event is being identified. When the event is identified, code relevant with the event procedure will be processed and done.

Advantages of Visual Basic

- Perfect set of objects
- Many icons and reusable components inclusive of graphical interface
- Good design of data structure for mathematic and string and graphic function operation
- Effectively connected to database especially (Open Database Connectivity) – server/client based for Microsoft SQL Server, SyBaseSQL and Oracle and Microsoft Access 2000 as well.
- Supports ActiveX
- Use package and deployment wizard to differentiate all the application easily
- Ability to connect to Internet

Added features in Visual Basic 6.0

- The Scripting Runtime Library
- The FileSystemObject and TextStream classes
- The Dictionary class
- New language features
- Dynamic control creation

- The CallByName feature

- The new Extender Validate event and CausesValidation property for controls

- Arrays can be returned from functions and assigned

- New component creation features

- Use UDTs as parameters or return types of public classes

- Persist class data in ActiveX components

- CreateObject improvement

- A class can act as a DataSource

- A class can act as a DataConsumer

- New HasDC property

- New members of the UserControl class

- Controls can be LightWeight (windowless)

- New FontChanged event of the StdFont class

2.6.2 Microsoft Access

Database Management System (DBMS) actually provide access to the users to reach the data and to change the data to relevant information needed by the users. There are many available DBMS but Microsoft Access 2000 is used to develop Ctutorial4u. Microsoft Access is the best way to connect intranet to retrieve data from database fast and effectively.

Chapter 3 System Analysis

3.1 System Development Methodology

Professional system developers and the customers they serve, share a common goal of building information systems that effectively support business process objectives. In order to ensure that cost-effective, quality systems are developed which address an organization's business goals, developers employ some kind of system development Process Model to direct the project's life cycle. An expert system emulates the behavior of a human expert within a specific domain of knowledge. There is a clear distinction between an expert system and conventional software. This clear distinction clearly manifests in the way problems are resolved in the two systems. The systematic approach will ensure availability of usable, stable, scalable, upgradeable and maintainable system. Typical activities performed in developing a system include the following:

- System conceptualization
- System requirements and benefits analysis
- Project adoption and project scoping
- System design
- Specification of software requirements
- Architectural design
- Detailed design
- Unit development
- Software integration & testing
- System integration & testing
- Installation at site
- Site testing and acceptance

- Training and documentation
- Implementation
- Maintenance

While nearly all system development efforts engage in some combination of the above tasks, they can be differentiated by the feedback and control methods employed during development and the timing of activities. Badly planned which does not meet the deadline and requirements would only bring frustrations among users. This chapter will discuss three different approaches to software development such as System Development Life Cycle's SDLC Waterfall Model, Linear Model of Expert System Development and Evolutionary Development's Prototype.

3.1.1 Waterfall Model

This is a classical model of system development. An alternative name for this model is the 'one-shot' approach [5]. As can be seen from the Figure 3.1, there is a sequence of activities working from top to bottom. The output for the each stage is the input for the next stage. This model allows developer to review back each phase base on the new idea of the system. The diagram shows some arrows pointing upwards and backwards. This indicates that the later stage might reveal the need for extra work at an earlier stage but this should definitely be the exception rather than rule. After all, the flow of waterfall should be downwards with the possibility of just little splashing back. The limited scope or iteration is in fact one of the strengths of this process model.

The waterfall model defines each development stage sequentially. Development process is more visible because of this cascade from one phase to another. In practice,

these phases overlap and feed information to each other. Therefore, the model doesn't depict a linear sequence of processes but iterations of activities.

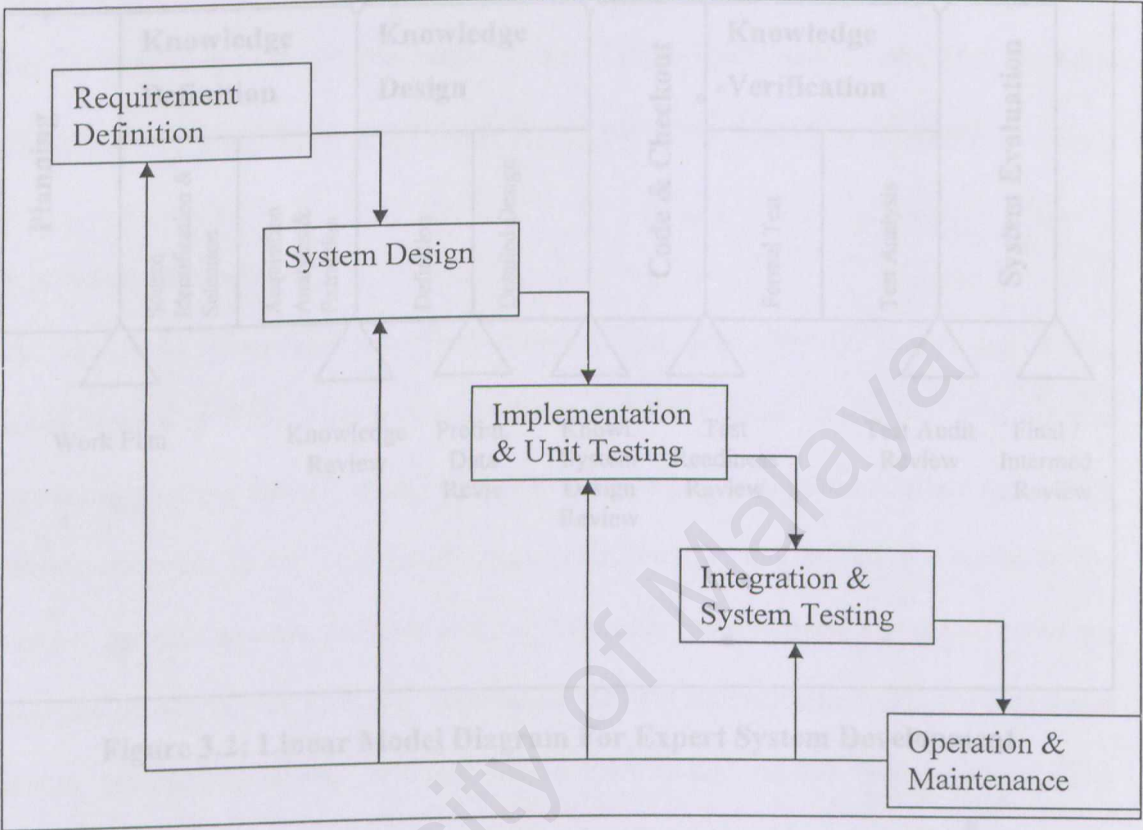


Figure 3.1: Waterfall Model

3.1.2 Linear Model of Experts System Development

Knowledge engineering has been accepted as one of the best approaches for developing an expert system. Most of the books on expert systems define knowledge engineering as the 'process of building an expert system' [4]. Knowledge engineering is a highly iterative process whereby the interest of the system designers lies in the problem knowledge. The designers partially build the system, test it and then modify it. This process is repeated throughout the project where the system's knowledge and designers understanding grows with each test.

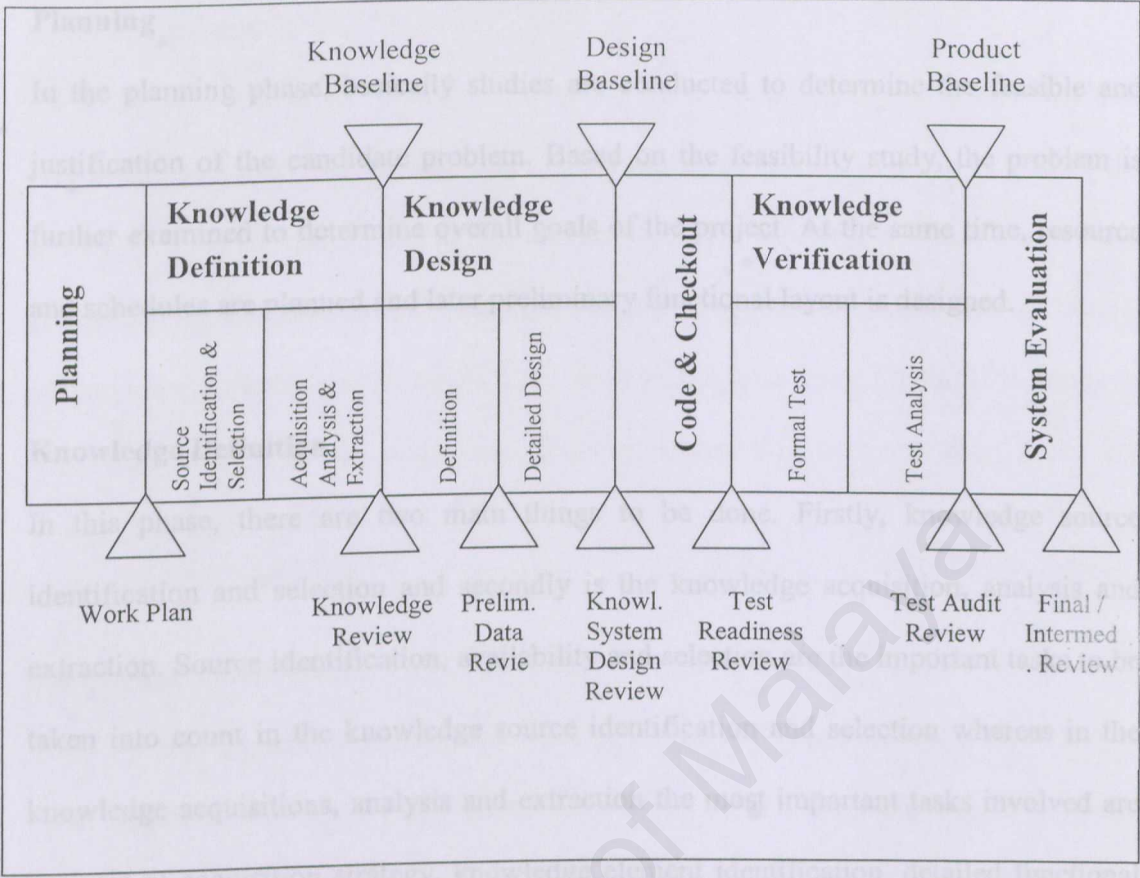


Figure 3.2: Linear Model Diagram For Expert System Development

The above figure depicts the main processes in developing an expert system. There are six main steps involved in the development of an expert system such as:

- Planning
- Knowledge Definition Knowledge
- Knowledge Design
- Code & Checkout
- Knowledge Verification
- System Evaluation

Planning

In the planning phase, basically studies are conducted to determine the feasible and justification of the candidate problem. Based on the feasibility study, the problem is further examined to determine overall goals of the project. At the same time, resource and schedules are planned and later preliminary functional layout is designed.

Knowledge Definition

In this phase, there are two main things to be done. Firstly, knowledge source identification and selection and secondly is the knowledge acquisition, analysis and extraction. Source identification, availability and selection are the important tasks to be taken into count in the knowledge source identification and selection whereas in the knowledge acquisitions, analysis and extraction the most important tasks involved are analysis of acquisition strategy, knowledge element identification, detailed functional layout, preliminary control flow and knowledge baseline. At this stage techniques like Data Flow Diagram (DFD), Entity Relationship Diagram (ERD), Structural Chart and State Transition Diagram (STD) can be used to understand the behavior of the system graphically. At this stage, both the functional and non-functional requirements is identified which are important to make sure the success of a system.

Knowledge Design

Knowledge Design is divided into two main tasks such as knowledge definition and detailed design. Once a thorough analysis has been performed the knowledge design will take over where the knowledge is defined in terms of knowledge representation, internal fact structure, preliminary user interface and initial test plan and then a detailed design is done such as designing the structure, implementation strategy, detailed user interface and detailed test plan.

Code & Checkout

Once the structure of the planned system has been designed, the code and checkout phase will take place. In this phase, an initial prototype is built to serve as vehicle for obtaining a better understanding of the problem. By building a small system and reviewing the test results with a domain expert, insights are gained into additional system requirements. The prototype also serves as the focal point for further interviews with the expert. At this stage user manuals, installation and operation guide are prepared as well.

Knowledge Verification

This is a testing phase. There are two types of testing will take part here such as formal tests, which include knowledge elicitation session with the domain expert to obtain further information that might have been overlooked in the earlier sessions, and refinements are made to the system. At this stage a test procedure will be used to examine the system and test reports are produced at the end of this phase. Next, test analysis is used to evaluate the test results and feeds recommendation to future enhancements.

System Evaluation

Finally, the verified system is then presented to the user to confirm meets their requirements. Once finalized, the completed system will be validated and final report or documentation is produced. The Figure 3.3 shows how the above is done phase-by-phase.

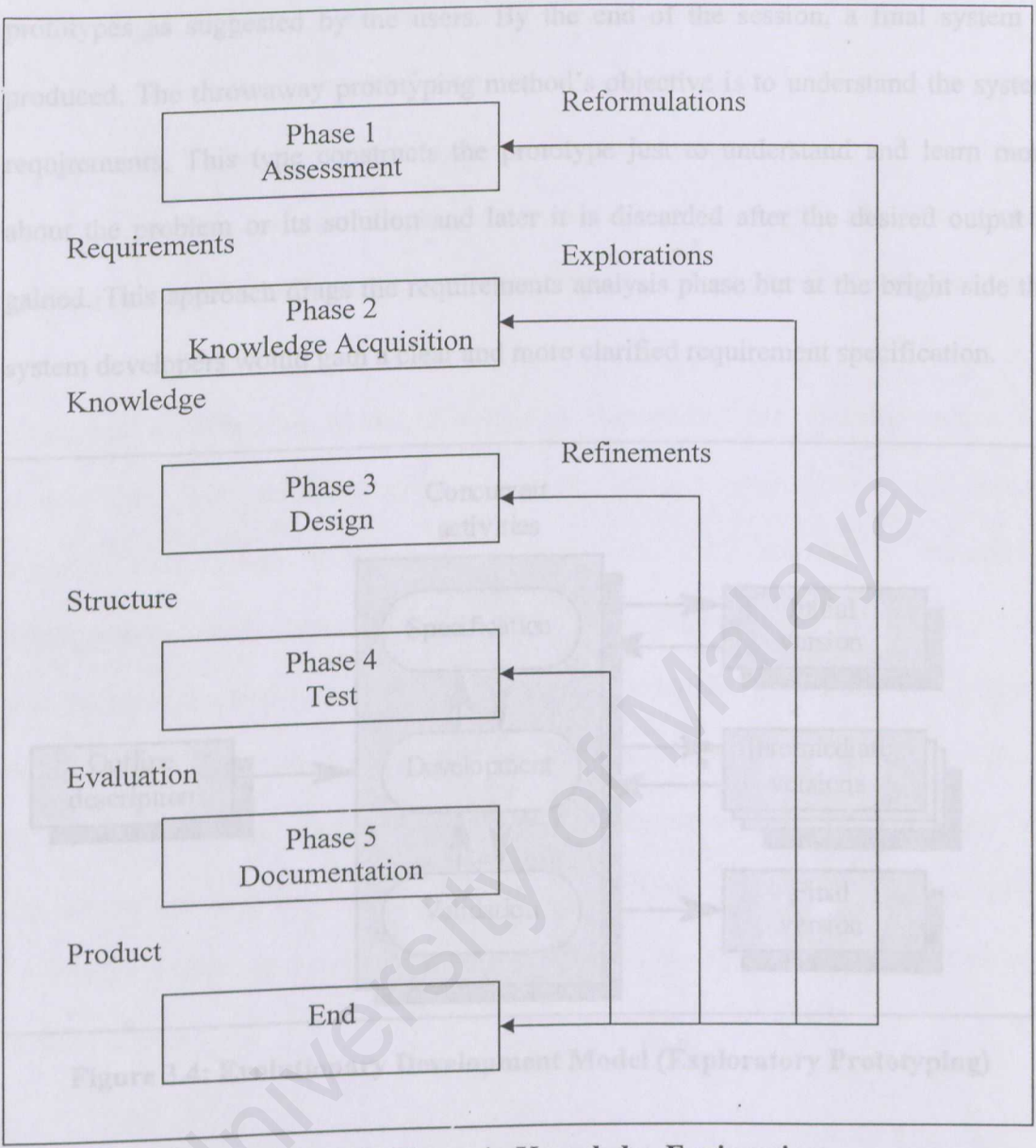


Figure 3.3: Phases in Knowledge Engineering

3.1.3 Evolutionary Development Model

Evolutionary Development Model as depicted in the Figure 3.4 is another popular development model available for our usage and reference. There are two types of evolutionary developments such as Exploratory Prototyping and Throwaway Prototyping. The exploratory prototyping is to work with the customers and to evolve a final system from an initial outline specification. This method should start with well-understood requirements. The development evolves by adding features to the

prototypes as suggested by the users. By the end of the session, a final system is produced. The throwaway prototyping method's objective is to understand the system requirements. This type constructs the prototype just to understand and learn more about the problem or its solution and later it is discarded after the desired output is gained. This approach drags the requirements analysis phase but at the bright side the system developers would gain a clear and more clarified requirement specification.

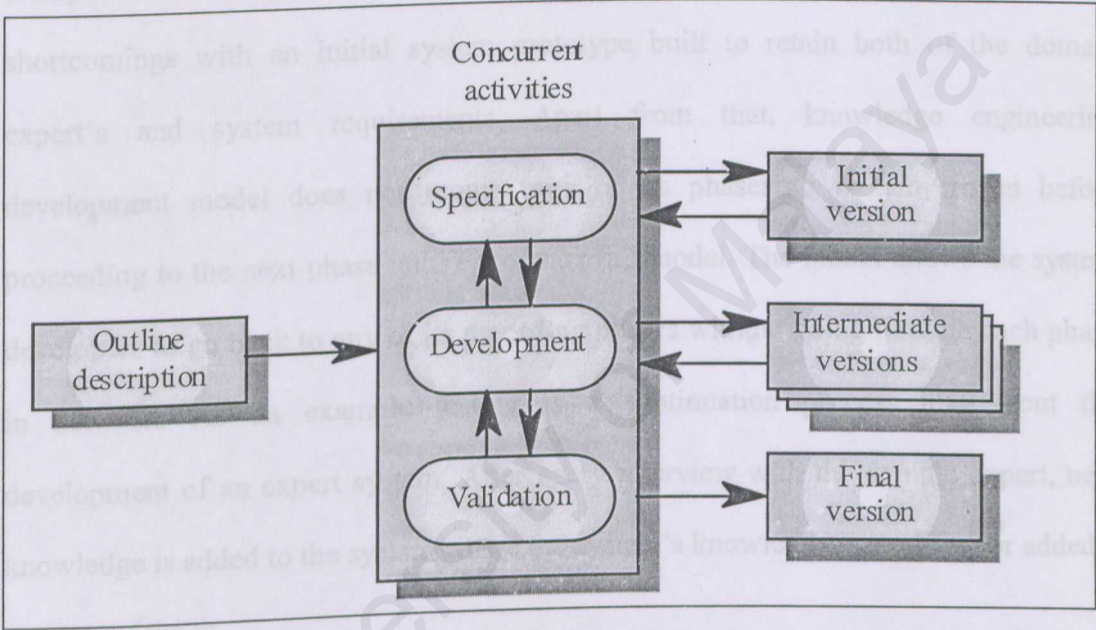


Figure 3.4: Evolutionary Development Model (Exploratory Prototyping)

3.1.4 Analysis and Synthesis

There are many Process Model or Life Cycle models available to choose as a methodology to develop our system but at the same time, the nature of the planned system is very important when selecting the process model. While nearly all system development efforts engage in some combination of the main four tasks such as analysis, design, coding testing and maintenance, they can be differentiated by the feedback and control methods employed during development and the timing of activities. Problem solving in expert system is predominantly heuristics based on trial

and error and guided by some reference to a predetermined goal. Problem solving in conventional software is usually through the algorithmic methods, which is procedural in nature. In, summary expert system encodes the domain dependent knowledge of everyday practitioners in some field and uses this knowledge to solve problems, instead of using comparatively domain-independent methods derived from Computer Science.

Compared to Waterfall Model, Knowledge engineering can actually reduce the shortcomings with an initial system prototype built to retain both of the domain expert's and system requirements. Apart from that, knowledge engineering development model does not require any of its phases to remain frozen before proceeding to the next phase unlike the waterfall model. The model allows the system developers to go back to any of its preceding phases without going through each phase in between. As an example testing is a continuation process throughout the development of an expert system. After every interview with the domain expert, new knowledge is added to the system where the system's knowledge is modified or added a number of times.

Knowledge engineering's prototype is built as a means to capture the problem domain correctly rather than as a tool for requirement analysis as what in the prototyping model make use of the prototype. Knowledge engineers only interact with the domain expert to capture possible system knowledge organization and structure and the intended users only interact with the prototype after the completed system is released to them unlike the prototyping model where the users are allowed to interact with it to assess further the user requirements.

As a conclusion, Linear Model of Knowledge engineering suits the best for developing CTutorial4u since it's a Rule Based System because it is better to adopt a system development strictly designed for expert system. In that case, let's look at how the main six phases in Linear Model of Knowledge Engineering helped to mould CTutorial4u to be a successful knowledge based system.

Knowledge Design is how we convert the defined knowledge at knowledge definition the form of system. All the classes in the class diagram were converted to be

Planning

During the planning phase of CTutorial4u, studies were conducted to determine the feasibility of the project through surfing the internet for similar systems, discussion with supervisor and questionnaire analysis among the higher learning students in Malaysia. Based on the feasibility study, the problem is further examined to determine overall goals of the project such as confirmation of intended users of the system, scope and objective of the project. At the same time, resources and schedules are planned and later preliminary functional layout is designed. This phase is actually equivalent to the System Conceptualization of typical system development methodology.

Knowledge Definition

As stated earlier, knowledge definition phase is divided into two sub phases which are knowledge source identification and selection and secondly is the knowledge acquisition, analysis and extraction. Knowledge identification phase is where all the sources, basically the rules to develop the system are identified. Through the literature review objects or entities, classes and use cases are identified. The knowledge acquisition phase helps to identify the relationship among the entities which visualizes the system in real world. The UML diagram is used to understand the behavior of the system graphically. The flow chart and structure chart were used to visualize the flow of the system. This is a combination of system requirement and project adoption and

project scoping phase of typical system development methodology. At this stage, both the functional and non-functional requirements are identified which are important to make sure the success of a system which will be explained further in chapter 4.

Knowledge Design

Knowledge Design is how we convert the defined knowledge at knowledge definition phase into the form of system. All the classes in the class diagram were converted to be tables in Microsoft Access 97 whereas based on the flow chart and structure chart, user interfaces were designed in Visual Basic 6.0. System design, specification of software requirements, architectural design and detailed design phases of typical software development methodology were done collectively at this phase of Linear Model of Knowledge Engineering.

Code & Checkout

In this phase, an initial prototype was built to serve as a vehicle for obtaining a better understanding of the problem. By building a small system and reviewing the test results with 5 students with intention to gain additional system requirements. The prototype also serves as the focal point for further interviews with the domain users (students). At this stage user manuals and installation were prepared. When compared to typical system development methodology it is same as system integration & testing phase.

Knowledge Verification

Formal test done to verify all requirements were met and include knowledge elicitation session with the domain user to obtain further information that might have been overlooked in the earlier sessions, and refinements are made to the system. User Acceptance Test (UAT), a test procedure was used to examine the system and test

reports are produced at the end of this phase. Next, test analysis is used to evaluate the test results and feeds recommendation to future enhancements. It is same stage as site testing and acceptance of Linear Model.

3.2.1.1 Discussion with the project supervisor

System Evaluation

Finally, the verified system is then presented to the user to confirm meets their requirements. The completed system was validated, implemented and final report or documentation is produced. Students were produced with user manual and tutors and lecturers were produced with installation guide. The system will be maintained by the tutors and lecturers in the future. So, this phase is equivalent with training and documentation, implementation and maintenance of the Linear Model of Knowledge Engineering.

3.2.1.2 Survey about C among students using questionnaires

3.2 Requirement Analysis

Preliminary study and analysis was done to collect data and information. Requirement collection and analysis is the process of collecting and analyzing information about the system that will be developed and to identify the user's requirements of the new system. A well collected data and information is vital to meet the objective of the project.

3.2.1 How to elucidate user requirement

There are many ways to collect user requirements such as interview, survey, study similar system or project and so on. Actually, information required for this project was gathered through discussion with the project supervisor, survey using questionnaires to gather student's perception of C programming taught in their institutes and expectation

about the knowledge based tutorial system and by surfing the internet for more information on the similar system available.

3.2.1.1 Discussion with the project supervisor

Discussion with the project supervisor, Puan Nazean Jomhari was done earlier to determine objective and scope of the project, the system requirements, information and data needed and the expected result of the project. Apart from that, from time to time clarification on the requirements is discussed with Puan Nazean. Other than that, she also gave some ideas on how the system should look like and through that her expectation was notified. The discussion with the supervisor helped a lot in designing the system.

3.2.1.2 Survey about C among students using Questionnaires

The questionnaire as attached in the Appendix A was distributed to a few local universities and private colleges in Klang Valley area in order to investigate the way C programming subject being taught in higher learning institutes of Malaysia. The survey results were then evaluated on how to improve the passing rates among the students using knowledge based C tutorial system designed in rule-based system architecture. The next section which is the functional requirements depicts all the analysis with the graphs from the questionnaire.

Firstly, a set of questions were prepared based on the requirements and information needed in developing the system. The questionnaire is divided into 3 parts; part A is the personal information, part B is general question about C programming and part C is about the proposed rule-based C tutorial system. Then, the higher learning institutes in Klang Valley were identified to take part in the survey. The questionnaires were

distributed randomly to the students who are taking C Programming subject in every particular institute. The students were asked to answer the questionnaire forms. All the answered questionnaires were collected and analyzed. There are 112 students from 15 higher learning institutes involved in the survey. They are from Universiti Putra Malaysia (UPM), Universiti Malaya (UM), Universiti Islam Antarabangsa (UIA), Universiti Kebangsaan Malaysia (UKM), Universiti Teknologi Mara (UITM), Universiti Multimedia (MMU), Universiti Tenaga Nasional (UNITEN), Politeknik Shah Alam, Cosmopint College, Informatics College, Sedaya College, Kolej Yayasan Mara, Kolej PTPL Shah Alam, Stamford College and Federal Institute of technology (FIT).

Part A of the questionnaire helps to identify background of the student. Question 1 identifies the student's higher learning institute and whereas question 2 helps to find out the semester and the year of the student in their respective institutes and question 3 is to find out the field or program the students are pursuing in that institutes. This will help in identifying the standard of the system whether to be easy or higher level.

Basically part B of the questionnaire helped to identify when exactly the C Programming subject is being offered in all the institutes in Malaysia. The questionnaire's core motive is to further seek the students' perception towards this subject. Question 1 of the Part B is to find out when the C programming subject being offered in their university and question 2 will clarify are there are any prerequisites required to register for this subject. Then the students' opinion about the subject was identified in question 3 where some of them think that it is very easy, moderate and there are also students think that it is very difficult. The intention of this question is to find out how many students think that C programming is difficult. Their perception

differs based on their family background and whether they have done programming subjects in their school. For those who never had any experience in programming in their secondary or primary school this subject looks like quite difficult to be adopted. Question 4 is to find out whether the students agree or disagree with the statement that assignment and lecture materials prepared by the lecturers are good. From this question reasons for the students failing or getting poor marks for this subject can be identified where if many disagree then improvement is required in teaching materials.

Question 5 is to identify how this subject is being taught in their institutes so that this system will be designed to be more helpful rather than normal teaching materials such as lecture notes, lab session and assignments. Question 6 is to verify whether the lecturer who is teaching this subject has all the characteristics listed. This is to make sure whether the university is selecting a qualified person as the lecturer to teach C programming. The average mark of the students in all their tests or quizzes is asked in question 7 to seek ways to improve and deliver a better tutorial system if most of them get low marks and adopt the lecturer's style of teaching if most of the students from those institutes that get high marks. Question 8 is to find out students problems with their current lectures. This question will help the developer to cater and eliminate those problems. At the same time, student's suggestion and recommendation is captured as well in the Question 9 on how to improve the course.

Apart from identifying the student's opinion about programming, there is also another part, Part C designed to identify the student's expectations of the proposed system. This section rendered a helping hand in designing the flow and features of the system. Question 1 of Part C to capture the features expected by the students such as whether the students wants the tutorial to cover all topics from a programming book, short and

long quizzes with answers, consultation session, sample sources code and search function. If the features given have more than 50% respondents who favor it then that feature will be included in the new rule-based tutorial system. Question 2 seeks the student's preferences on how the system to be designed whether by selected topics, by application or by all chapters following a reference book. Question 3 is to find out how long a student needs to finish a quiz, which consists of 10 simple questions. From this question the short and long quizzes can be designed and the time out can be set based on the majority of the answers. Question 4 is to find out how to organize the quizzes whether after every chapter or mixed questions from all chapters or both ways. Moreover, question 5 helps to highlight what are the C programming topics to be given more emphasis and consideration. Lastly, Question 6 was designed to seek the students preferences whether the system need to be standalone system or an online system.

3.2.1.3 Surf through the Internet and research on reading materials

Internet is a vast network of interconnected computers. The deluge of web pages has generated a strong commentary on the tragedy of the flood of information. Surf the Internet on existing systems to adopt some ideas and to collect relevant pictures and graphics. So, books on developing an expert system were read. Then, Microsoft Image Composer and Adobe Photoshop 7.0 were explored on how to manipulate the pictures. The reason to use some multimedia tools is to develop a system, which can attract the users and make the system interesting and attractive.

3.2.2 Functional requirements

Functional requirements describe the interaction between the system and its environment. Functional requirements capture the tasks that a system must perform and as such does not include implementation details such as what hardware or software the

system must use. The functional requirements of CTutorial4u include administration session, lecture notes session, tutorial session and quiz session. The features planned to be embedded in the system in the beginning is as in the Figure 3.5 but based on the visibilities of all the features the real features are administration session, lecture notes, tutorial session and quiz session.

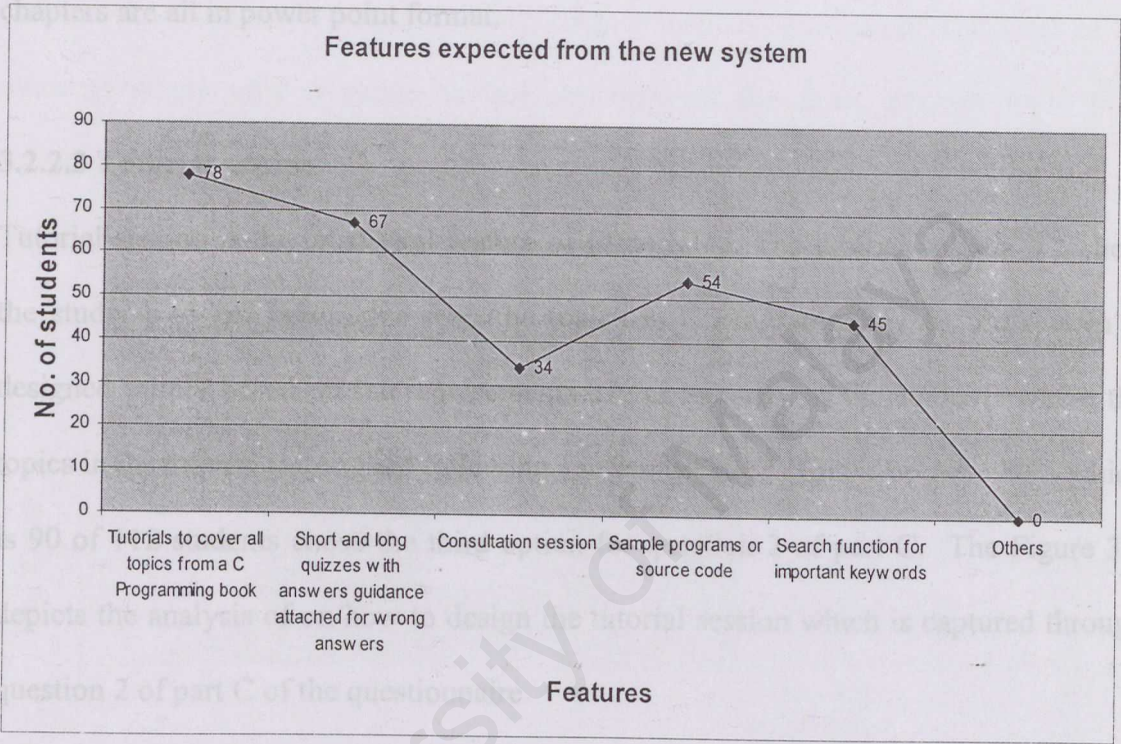


Figure 3.5: Line graph shows the features expected from CTutorial4u

3.2.2.1 Administration session

The main important function is the administration session where the system administrator has extra authority that allows them to maintain the system functionality such as add and modify the queries in the knowledge base of tutorial and quiz session. Any modification to the system knowledge base could occur when an error is detected or if the students decide to add more queries to the quiz session to enhance its usability because only the lecturer or tutors who act as the administrator is allowed to modify the system.

3.2.2.2 Lecture notes session

Lectures notes are an essential session to the students to explore before they try the tutorial session and quiz session. Lecture notes contains ten chapter such as Introduction, Condition, Program Controls, Functions, Arrays, Pointers, Characters and strings, Formatted input/output, Structures and Unions and File processing. These chapters are all in power point format.

3.2.2.3 Tutorial session

Tutorial session is the most vital feature of Ctutorial4u. The tutorial session is to help the students to get familiarize with the topics in C programming. As the system is designed purely based on the requirements of the majority of the students where the topics in the tutorial is designed following a reference book. This is because 80% which is 90 of 112 students chose the third option for question 2 of part C. The Figure 3.6 depicts the analysis of on how to design the tutorial session which is captured through question 2 of part C of the questionnaire.

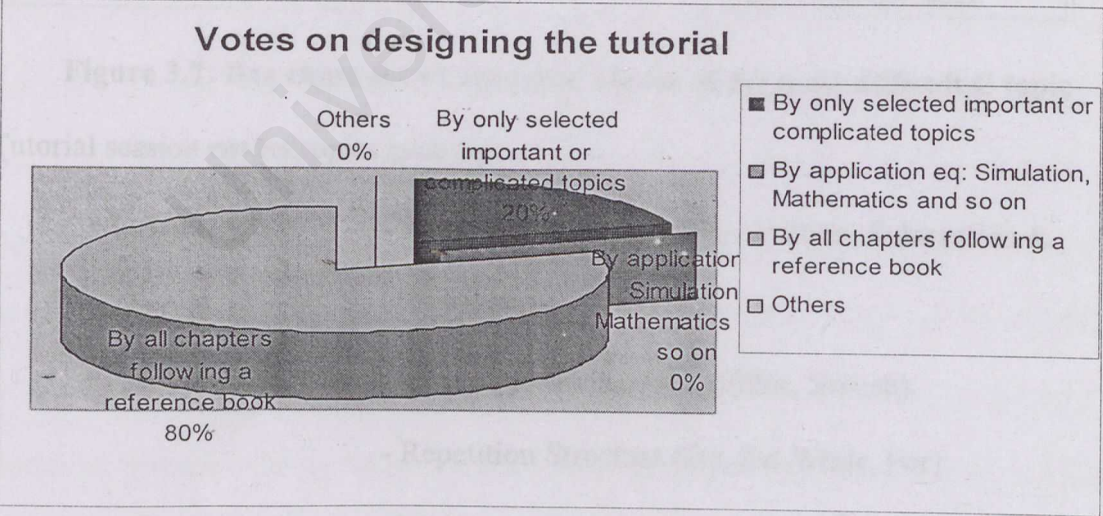


Figure 3.6: Pie Chart shows votes on ways to design the tutorial

At the same time, there will be more solutions and examples available for the arrays and sorts then followed by pointers, counters, control structures and string functions and loops. Most of the students, 45 chose arrays and sorts as the toughest topic and followed by 42 students who answered pointers as the most difficult chapter to understand. Subsequently, only 12 students say counters and only 8 students chose strings and loops as the most difficult chapter in C Programming. The least of the students where only 5 chose control structures as the most difficult topic in C programming. This information was captured from question 5 of Part C of the questionnaire and Figure 3.7 shows the results for that question.

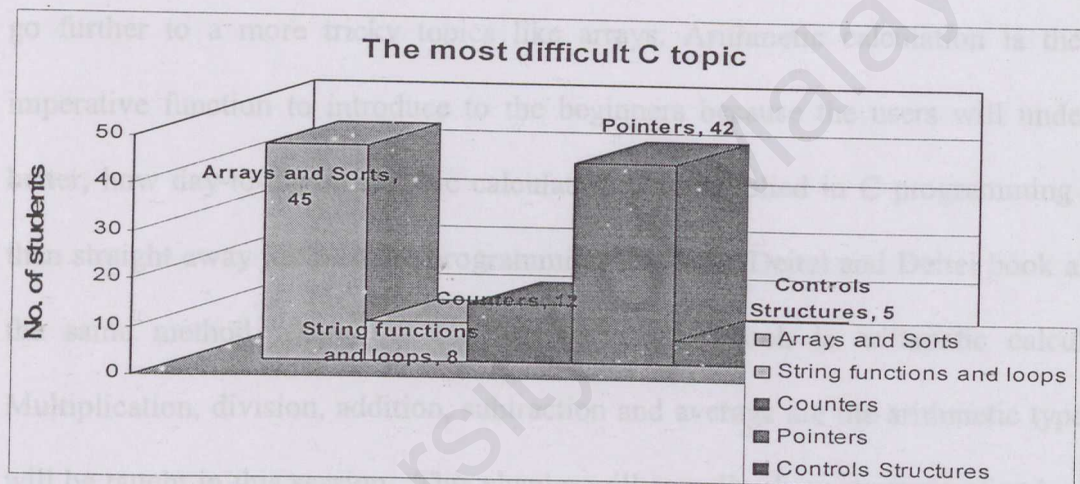


Figure 3.7: Bar chart shows students' choice of the most difficult C topic

Tutorial session covers topics such as:

- Arithmetic - Multiplication, Division, Addition, Subtraction & Average.
- Control Structures - Selection structure (If, If/Else, Switch)
 - Repetition Structure (Do, Do-While, For)
- Functions - Maximum & Minimum
- Array - Size of array from 1 to 10 & with or without histogram.
- Pointers - Call by reference & call by value
- File Processing - Create a Sequential file

- Writing to a Random Access file

- Reading a Random Access file

The purpose of the session is to display the source code based on the user's selection from the options provided for each chapter. Then, when the users select to execute the code, the system will execute the code and show the output. The details of this session can be clearly seen in the user manual in Appendix C.

Firstly, arithmetic topic is not planned to be covered in this session but at the end advised by the supervisor to cover it since it would be a stepping stone for the users to go further to a more tricky topics like arrays. Arithmetic calculation is the most imperative function to introduce to the beginners because the users will understand better, how day-to-day arithmetic calculation being applied in C programming rather than straight away jump to the programming side. The Deitel and Deitel book also do the same method where the first chapter of the book is arithmetic calculation. Multiplication, division, addition, subtraction and average are the arithmetic types that will be taught in this session. This chapter will actually show example of codes based on the user input for arithmetic types, variable type (integer, double and float) and value of variable A and B. Once user gives inputs for all these mandatory fields, the source code will be displayed in the list box on the right panel. After that, the user can select whether to execute the code or not. The beauty of this system is to show how the real output in DOS prompt when source code is being executed in Visual C++ compiler. Please refer to Appendix C for more details on this topic at Figure C.9 and C.10.

Control structure is divided into two types which are selection structure (if, if/else & switch) and repetition structure (so, do-while & for). The source code will be displayed

based on the user's input for the test grade. The selection structure will be based on a scenario to print whether the student has passed the examination or failed based of the marks or grade entered where the pre-condition set in the working memory is 60 for passing grade. So if the grade is more than 60, then message "Passed" will appear and the other way round if it is less than 60, where "Failed" message will appear as in Figure C.12 of Appendix C. The repetition structure is to calculate the average marks for a student based on five tests grades.

Figure C.17 and C.18 in Appendix C for more details.

The function topic, actually introduces maximum and minimum function. The system will display the source code to identify the maximum and minimum value based on the user's input for the integer 1, 2 and 3. When the source code being executed, the integer values keyed in by the user will be verified and the system will show the maximum or minimum value based on the selection for the function type selected by the user. The screen design of this topic is depicted in Figure C.13 and C.14 of Appendix C.

The topic array is introduced in this system to show how array as a group of memory locations related by the fact that they all have the same name and the same type and users needs to specify the type of each element and the number of elements required by each array so that the computer may reserve the appropriate amount of memory. Moreover the Deitel and Deitel book introduces histogram in conjunction with every value of the element in the group of array. Based on the array size selected by the user (minimum 2 and maximum 10 array size) and value of each element also captured from the user's input and then the source code will be displayed. So, when the source code being executed, the output will be ("Element, Value, Histogram") as in the Figure C.16 of Appendix C.

The topic pointer covers ways to pass argument such as call value and call by reference. Call by value uses direct passing argument based on the value integers keyed in, the system will cube the value and square the value based on the user's selection. In call by reference the concept of array plays a major role. The pointer will be pointing to the variable in the array and call the function cube by reference or square by reference. Then there is an option whether to sort the integer variables in the array or not. If the user selects to sort, the system will display the code for bubble sort. Please refer to Figure C.17 and C.18 in Appendix C for more details.

The file processing topic contains processes like Create a Sequential file, Writing to a Random Access file and Reading a Random Access file. Based on the variable type, variable name and file name, there will be source code displayed to create or to write to a sequential file as in Figure C.19. Whereas for reading a random access file, the system will display pre-coded program saved in the working memory.

3.2.2.4 Quiz session

There will be quiz organized in both after every chapter and mixed question from all chapters because 57% of the students chose to have the quiz both ways after every chapter and as well as have mixed questions from all chapters at the end of the course. This user needs was captured from question 4 Part C and this is depicted in Figure 3.8. The quiz session covers 10 simple questions from chapter arithmetic, control structures, functions, arrays, pointer, file processing, and mixed questions from all chapters.

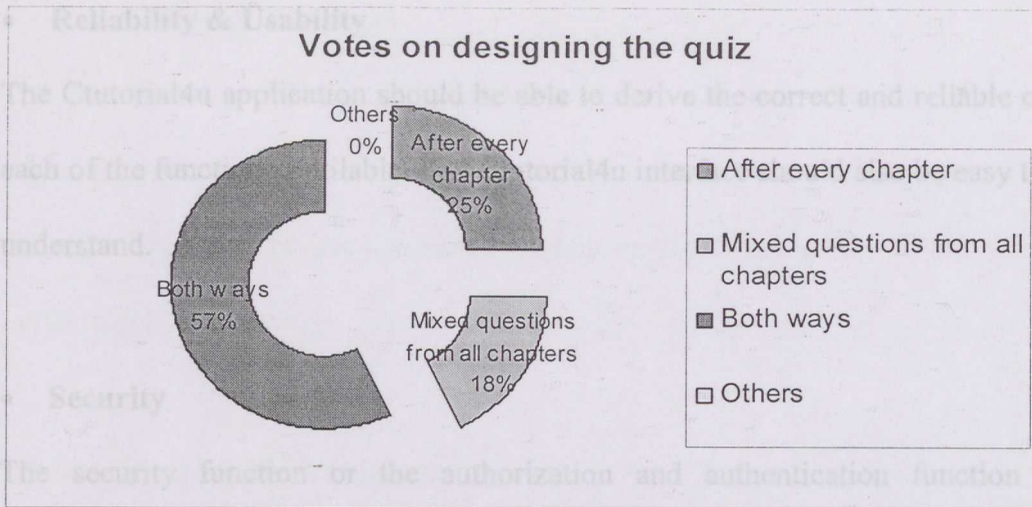


Figure 3.8: Doughnut chart shows the responses on how to design the quizzes

The time limit to finish ten simple questions is fixed to 15 minutes just to meet requirements of the majority users where 67% of the students chose 15 minutes for the question 3 of part C and the result of the survey is shown clearly in the pie chart Figure 3.9.

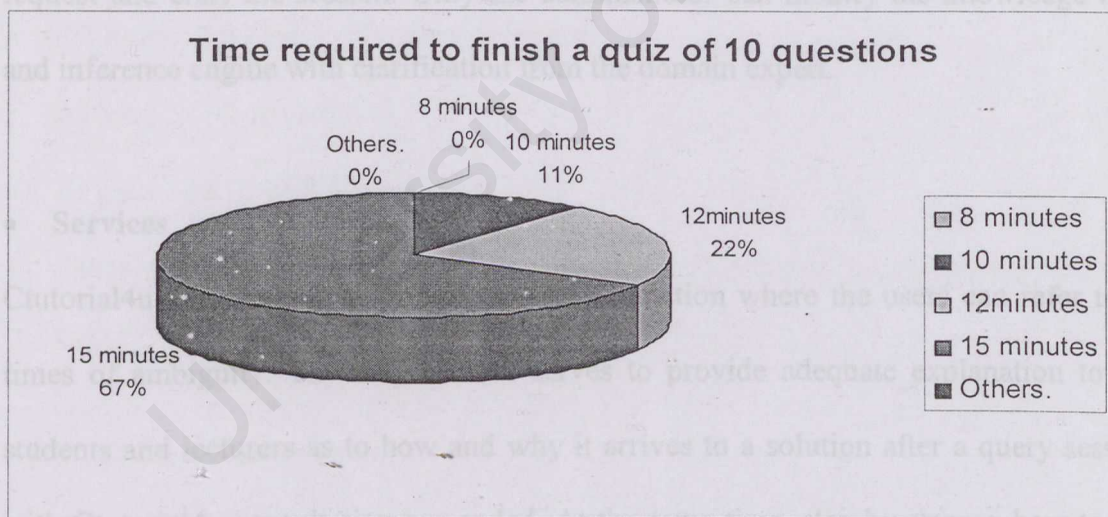


Figure 3.9: Pie chart on time to finish a quiz consists of 10 simple questions

3.2.3 Non-Functional Requirements

The non-functional requirement is the added-features of a system where it can't be seen directly but it is embedded in the system itself. Among the non-functional requirements for the Ctutorial4u project includes:

- **Reliability & Usability**

The Ctutorial4u application should be able to derive the correct and reliable output for each of the functions available. The Ctutorial4u interface should also be easy to use and understand.

- **Security**

The security function or the authorization and authentication function prevents unauthorized users from gaining access to the system and looking at others scores and information. This function will provide the system a sense of security in an electronic based environment. When a user login to the system a special user login request is sent to the system to validate the user and establish a user type (admin and user). If the password and user ID provided by the user is incorrect then the system will reject the request and ends the session. Only the administrator can modify the knowledge base and inference engine with clarification from the domain expert.

- **Services**

Ctutorial4u will be able to provide a 'Help' function where the users can refer to in times of ambiguity. The help section serves to provide adequate explanation to the students and lecturers as to how and why it arrives to a solution after a query session with Ctutorial4u consultation has ended. At the same time, step by step on how to use the system is given in this section. This requirement is not gathered from the questionnaire but by surfing the internet and looking in other similar products.

- **Consistency & Efficiency**

The interfaces for all the pages are consistent in the use of colors, font sizes, text structure, available graphics and menu bar. Consistent and efficient screens will allow the users to use the features provided without having to memorize all the commands to perform certain tasks.

3.3 Development Environments

Development requirements include hardware specifications and software requirements required to develop the system.

3.3.1 Hardware Requirements

CTutorial4u would interface with the following hardware requirements:

- Pentium II 100 MHz processor or higher
- At least 2 GB of hard disk
- At least 32 MB of RAM
- Other standard computer peripherals – monitor, keyboard, mouse printer and CD-ROM.

3.3.2 Software Requirements

The Table 3.1 provides the minimum software requirements required to develop the CTutorial4u.

Table 3.1: Software requirements for Ctutorial4u

Software Type	Required Software	Remarks
Operating System	Windows 98	Minimum OS required is Win 98
Database	MS Access 2000	MS Access is required to develop the database
Software Development Tool	Visual Basic 6.0 (VB6) Adobe Photoshop 7.0 MS Image Composer MS Office 2000 MS Visual C++ 6.0	VB6 and will be used to develop the user interface. MS Visual C++ 6.0 is to compile and run the code before adding to the knowledge base. Adobe Photoshop 7.0 and MS Image Composer are required to design the image needed whereas MS Office 2000 is required to produce the documentation of the system.

Operating System

Microsoft Windows 98 was chosen as the development platform because it is easy to use and flexible. It is also supports Visual Basic as well as other development tool. Windows 98 is also very user-friendly, safe and highly reliable.

Database

The database chosen for the CTutorial4u is Microsoft Access 2000.

Software Development Tool

Visual Basic 6.0 is chosen as the software development toll to develop the user interface and at the same time code is developed in VB6.0 development environment. Adobe Photoshop and Microsoft Image Composer helps to design the image needed to use in designing the interface of the system. Microsoft Word 97 is used develop the documentation of the system. MS Visual C++ 6.0 is to compile and run the code before adding to the knowledge base.

Chapter 4 System Design

4.1 Overview of Ctutorial4u Architecture

A system development should be based on architecture. CTutorial4u is designed based on rule-based system architecture as depicted in Figure 2.3 in chapter 2 but further enhanced as in Figure 4.1. This is because architecture basically decides on the development technology, hardware and software required to develop the system. At the same time, system design decides on the database design, process and interface flow of the system. Ctutorial4u is basically client-based system. It can be installed in any Windows-based client.

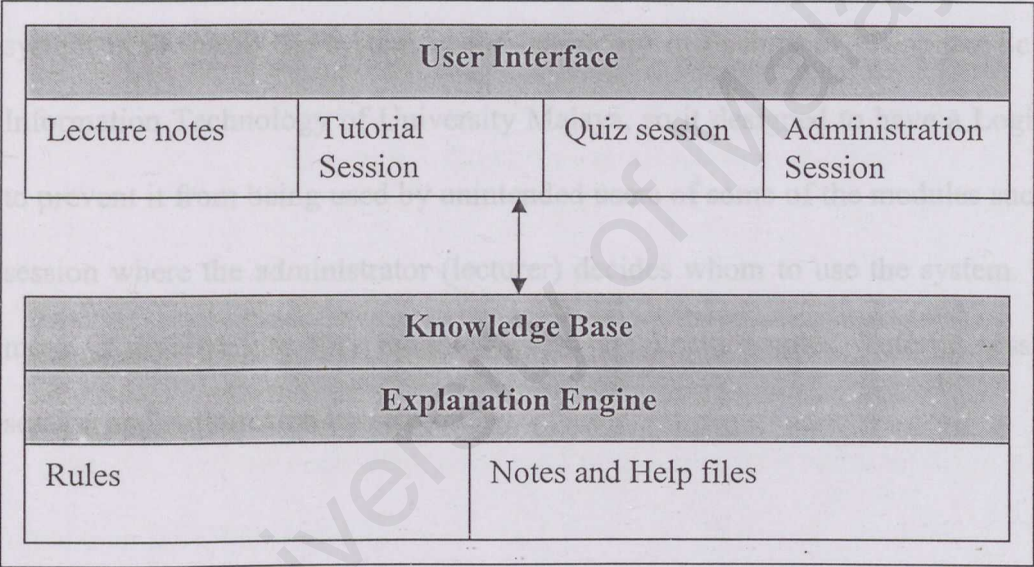


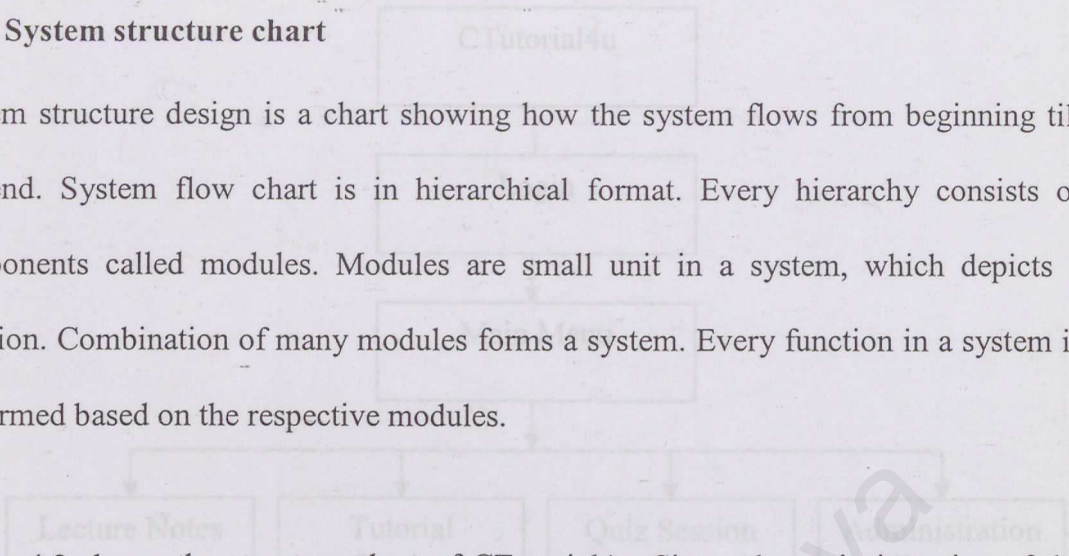
Figure 4.1: Architecture of CTutorial4u

The architecture of CTutorial4u was developed by linking the user interface with the knowledge base and explanation engine. All the rules, static notes and help files are stored in the knowledge base. The tutorial session and quiz session is developed by adding a layer of functions to form the explanation engine. It is supported by a collection of static text files containing explanatory notes, help instructions and rules used by the system which is stored in the knowledge base.

4.2 Process design

4.2.1 System structure chart

System structure design is a chart showing how the system flows from beginning till the end. System flow chart is in hierarchical format. Every hierarchy consists of components called modules. Modules are small unit in a system, which depicts a function. Combination of many modules forms a system. Every function in a system is performed based on the respective modules.



```
graph TD; CTutorial4u[CTutorial4u] --> LectureNotes[Lecture Notes]; CTutorial4u --> Tutorial[Tutorial]; CTutorial4u --> QuizSession[Quiz Session]; CTutorial4u --> Administration[Administration];
```

Figure 4.2 shows the structure chart of CTutorial4u. Since, the main intention of the system is to install the system in the laboratory in Faculty of Computer Science and Information Technology of University Malaya, so it designed to have a Login module to prevent it from being used by unintended users of some of the modules such as Quiz session where the administrator (lecturer) decides whom to use the system. The main menu is divided into four modules which are Lecture notes, Tutorial session, Quiz session and Administration session.

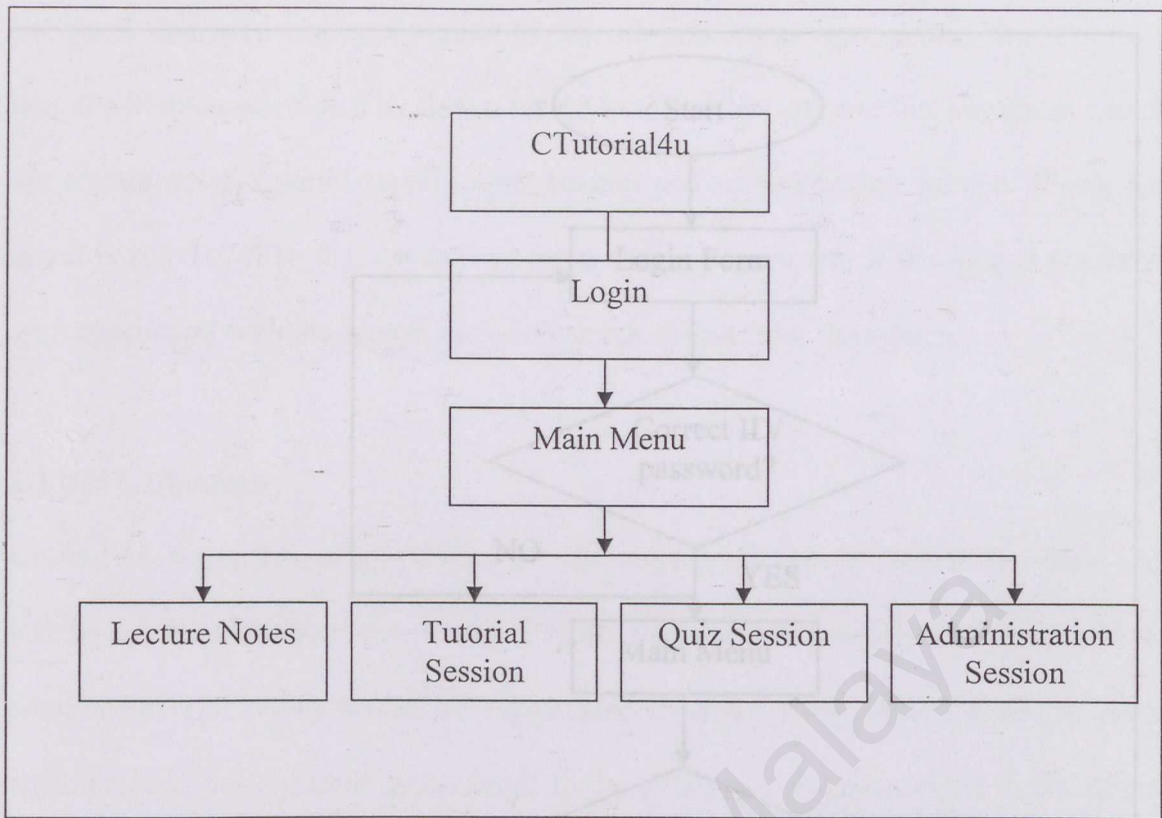


Figure 4.2: Structure chart of CTutorial4u

4.2.2 Flow chart diagram

Flow Chart Diagram graphically characterized data processes and flows in the business system. So, the flow chart diagram is used to describe the system inputs, processes and outputs in the CTutrorail4u.

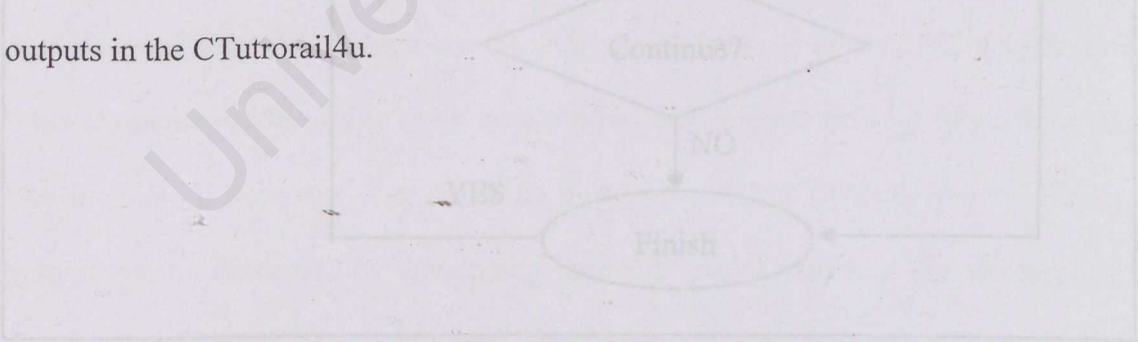


Figure 4.3: Flow chart of CTutorial4u

The Figure 4.3 shows flow chart of the system. Firstly, the system will verify the username and the password supplied by the user and if it doesn't match with the username and password in the database, then a prompts will indicate warning of wrong

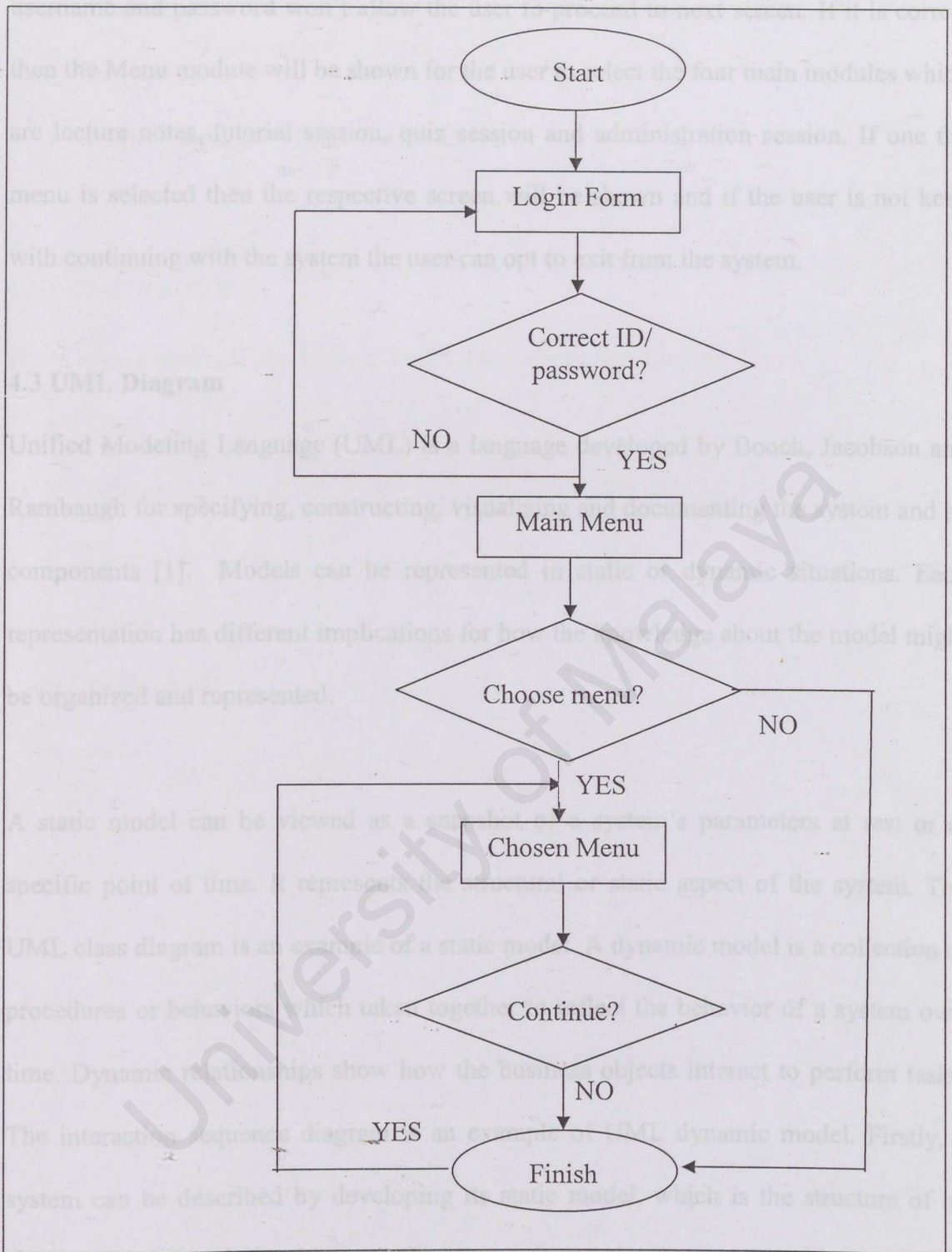


Figure 4.3: Flow chart of CTutorial4u

The Figure 4.3 shows flow chart of the system. Firstly, the system will verify the username and the password supplied by the user and if it doesn't match with the username and password in the database, then a prompt will indicate warning of wrong

username and password won't allow the user to proceed to next screen. If it is correct then the Menu module will be shown for the user to select the four main modules which are lecture notes, tutorial session, quiz session and administration session. If one the menu is selected then the respective screen will be shown and if the user is not keen with continuing with the system the user can opt to exit from the system.

4.3.1 Class diagram

There are six main classes in the Ctutorial4u such as listed below:

4.3 UML Diagram

Unified Modeling Language (UML) is a language developed by Booch, Jacobson and Rumbaugh for specifying, constructing, visualizing and documenting the system and its components [1]. Models can be represented in static or dynamic situations. Each representation has different implications for how the knowledge about the model might be organized and represented.

• Results

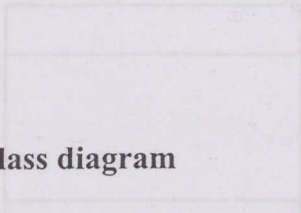
A static model can be viewed as a snapshot of a system's parameters at rest or at specific point of time. It represents the structural or static aspect of the system. The UML class diagram is an example of a static model. A dynamic model is a collection of procedures or behaviors which taken together to reflect the behavior of a system over time. Dynamic relationships show how the business objects interact to perform tasks. The interaction sequence diagram is an example of UML dynamic model. Firstly, a system can be described by developing its static model, which is the structure of its objects and their relationship to each other at frozen time, a baseline. Then, dynamic models are developed to examine changes to the objects and their relationship over time.

The UML approach adopted for the Ctutorial4u will include the three basic UML graphical diagrams:

- Class diagram (static model)

- Use-case diagram

- Sequence diagram (dynamic diagram)



4.3.1 Class diagram

There are six main classes in the CTutorial4u such as listed below:

- User
- Admin
- Quiz
- Tutorial
- Explanation
- Results



Table 4.1: Class Notation Table

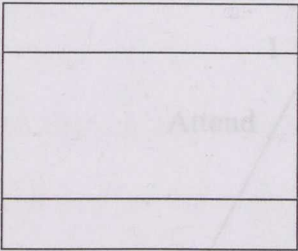
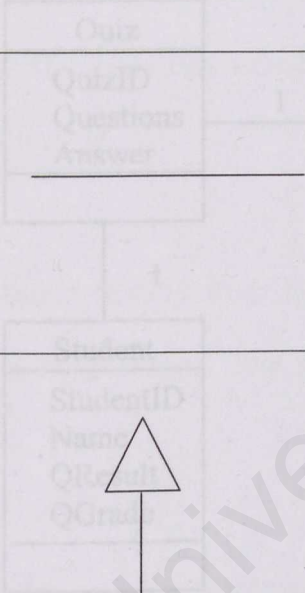
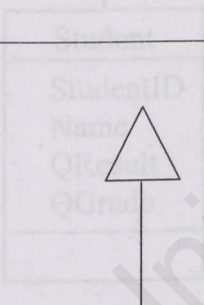
Component	Explanation
	<p>Class notation</p> <ul style="list-style-type: none">Shows a snapshot of the detailed state of the system at a point of timeRectangle with 3 components. Top compartment represents class name, middle compartment represents its attribute and bottom compartment represents a list of operations.
	<p>Binary associations</p> <ul style="list-style-type: none">Shows associations between two classes. May have an association name.
	<p>Generalization</p> <ul style="list-style-type: none">Shows relationship between a more general class and a more specific class.

Figure 4.4: Class Diagram of CTutorial4u

The Class diagram depicted in Figure 4.4 shows that 'Users' class is the super class for 'Admin' class. Both the user and administrator are presented in 'User' super class. The diagram mainly depicts the relationship between the user and the classes that associates with it in the Tutorial Mo. Each user has the choice to access the Tutorial Session, hence the (0..*) cardinality. In the Tutorial Session, there are a variety of queries on

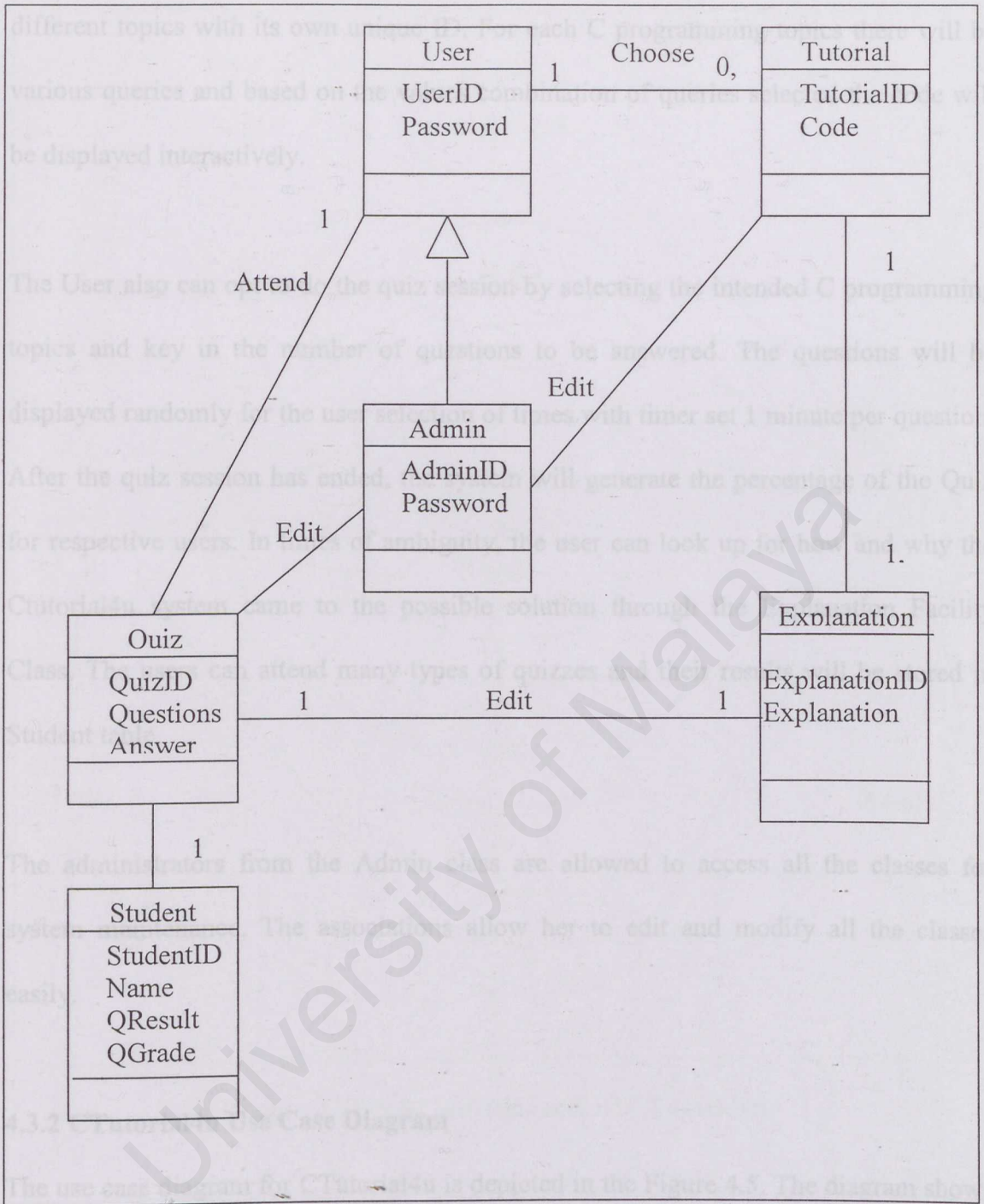


Figure 4.4: Class Diagram of CTutorial4u

The Class diagram depicted in Figure 4.4 shows that ‘Users’ class is the super class for ‘Admin’ class. Both the user and administrator are presented in ‘User’ super class. The diagram mainly depicts the relationship between the user and the classes that associates with it in the Tutorial Mode. Each user has the choice to access the Tutorial Session, hence the (o,*) cardinality. In the Tutorial Session, there are a variety of queries on

different topics with its own unique ID. For each C programming topics there will be various queries and based on the values combination of queries selected the code will be displayed interactively.

The User also can opt to do the quiz session by selecting the intended C programming topics and key in the number of questions to be answered. The questions will be displayed randomly for the user selection of times with timer set 1 minute per question. After the quiz session has ended, the system will generate the percentage of the Quiz for respective users. In times of ambiguity, the user can look up for how and why the Ctutorial4u system came to the possible solution through the Explanation Facility Class. The users can attend many types of quizzes and their results will be stored in Student table.

The administrators from the Admin class are allowed to access all the classes for system maintenance. The associations allow her to edit and modify all the classes easily.

4.3.2 CTutorial4u Use Case Diagram

The use case diagram for CTutorial4u is depicted in the Figure 4.5. The diagram shows the relationship between the actors, the 'User' (Student and Administrator), 'Admin' (Administrator) and use cases.

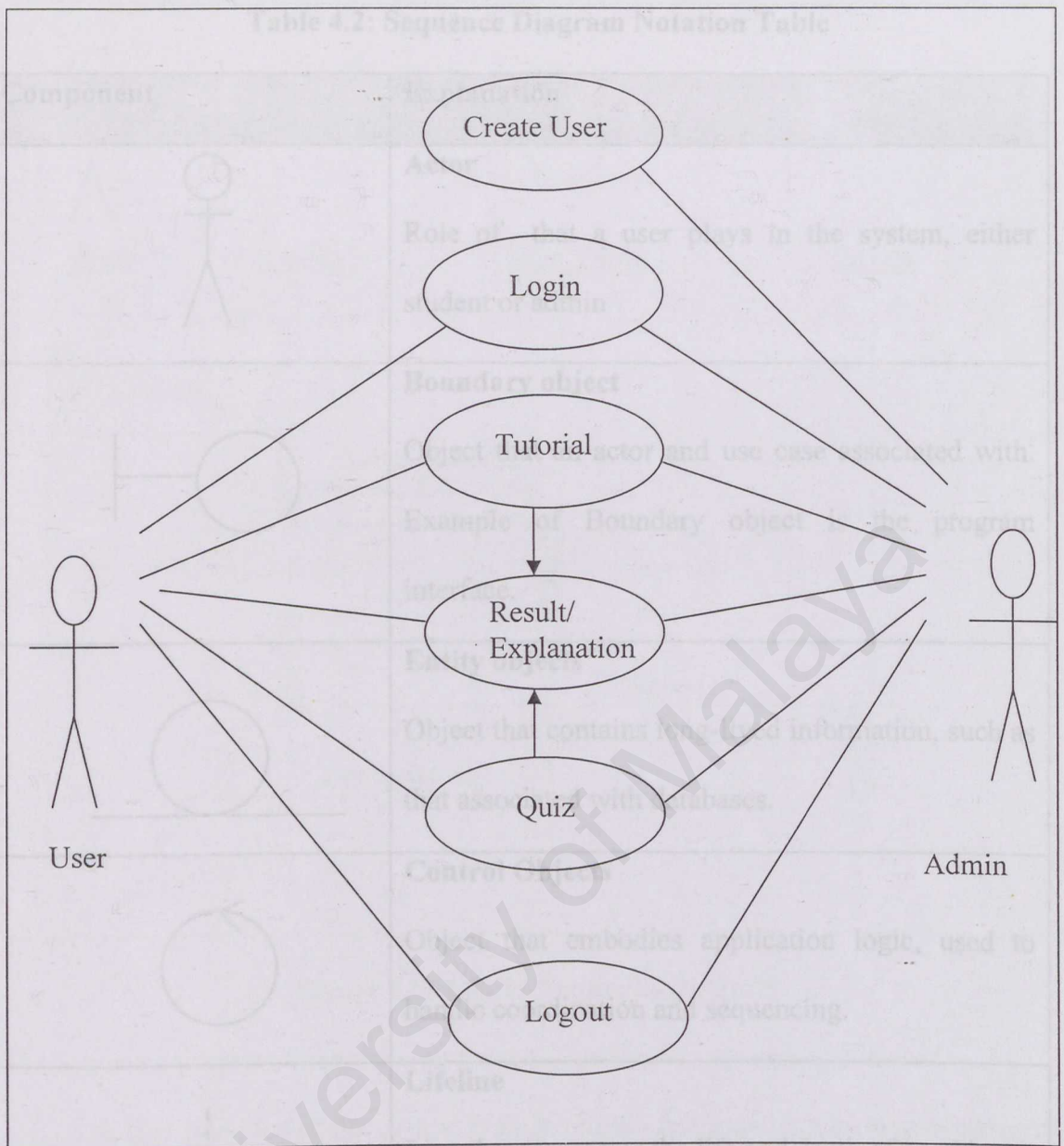

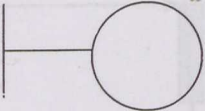
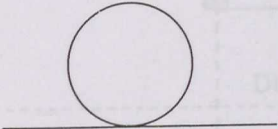
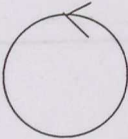




Figure 4.5: Use Case Diagram of CTutorial4u

4.3.3 Sequence Diagram

Sequence diagram shows the interaction between objects in time ordering fashion. It is used in a system to show the interaction between user, screen, objects and entity in the system. It is also to decide the object and class involved in a scenario. Other than that, message series being sent between objects to accomplish scenario function also can be identified.

Table 4.2: Sequence Diagram Notation Table

Component	Explanation
	Actor Role of that a user plays in the system, either student or admin
	Boundary object Object that an actor and use case associated with. Example of Boundary object is the program interface.
	Entity objects Object that contains long-lived information, such as that associated with databases.
	Control Objects Object that embodies application logic, used to handle coordination and sequencing.
	Lifeline Line that represents the life and death of an object.
	Focus of control Shows period of time which the object is in control of the flow.

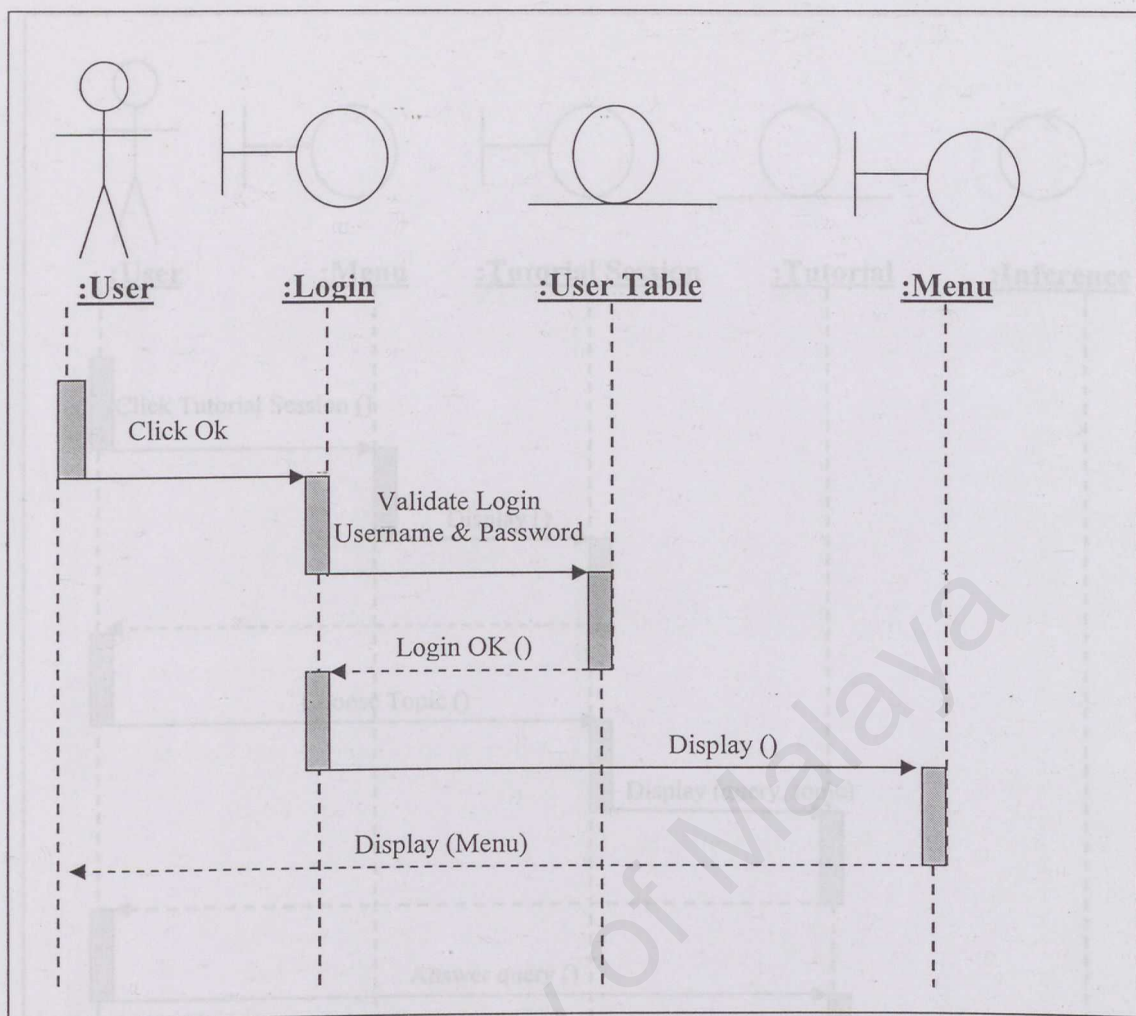


Figure 4.6: Log In Sequence Diagram

The sequence diagram time ordering for the CTutorial4u Log In is as follows:

- I. The first interface of the system is Login form. There will a prompt requesting the user to key in their Username and password.
- II. The user enters their Username and password and then click Ok button.
- III. The system validates the login information against persistent data in User – Account Table.
- IV. Then system will display the Menu interface to the User

Figure 4.7: Overall Tutorial Session Sequence Diagram

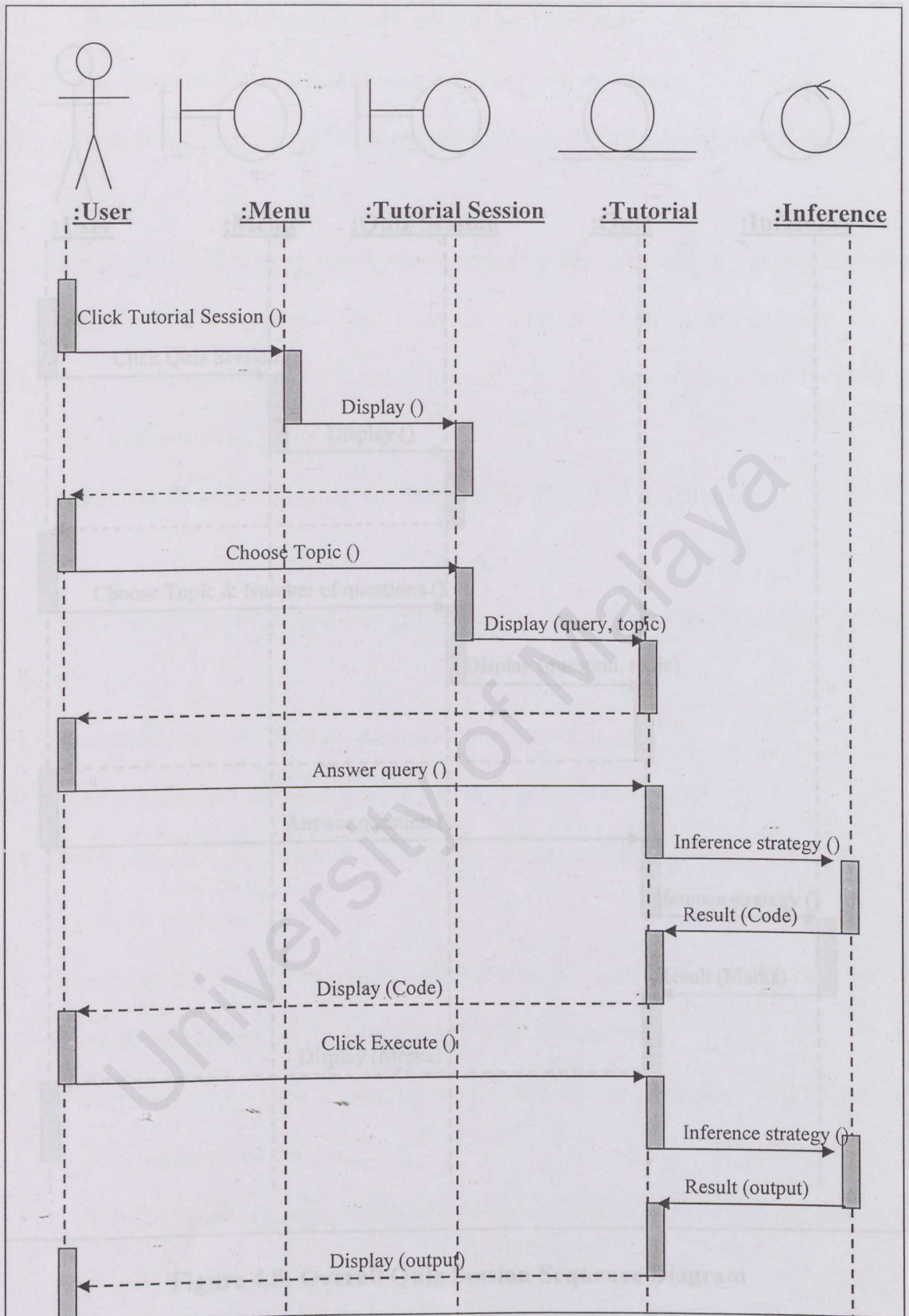


Figure 4.7: Overall Tutorial Session Sequence Diagram

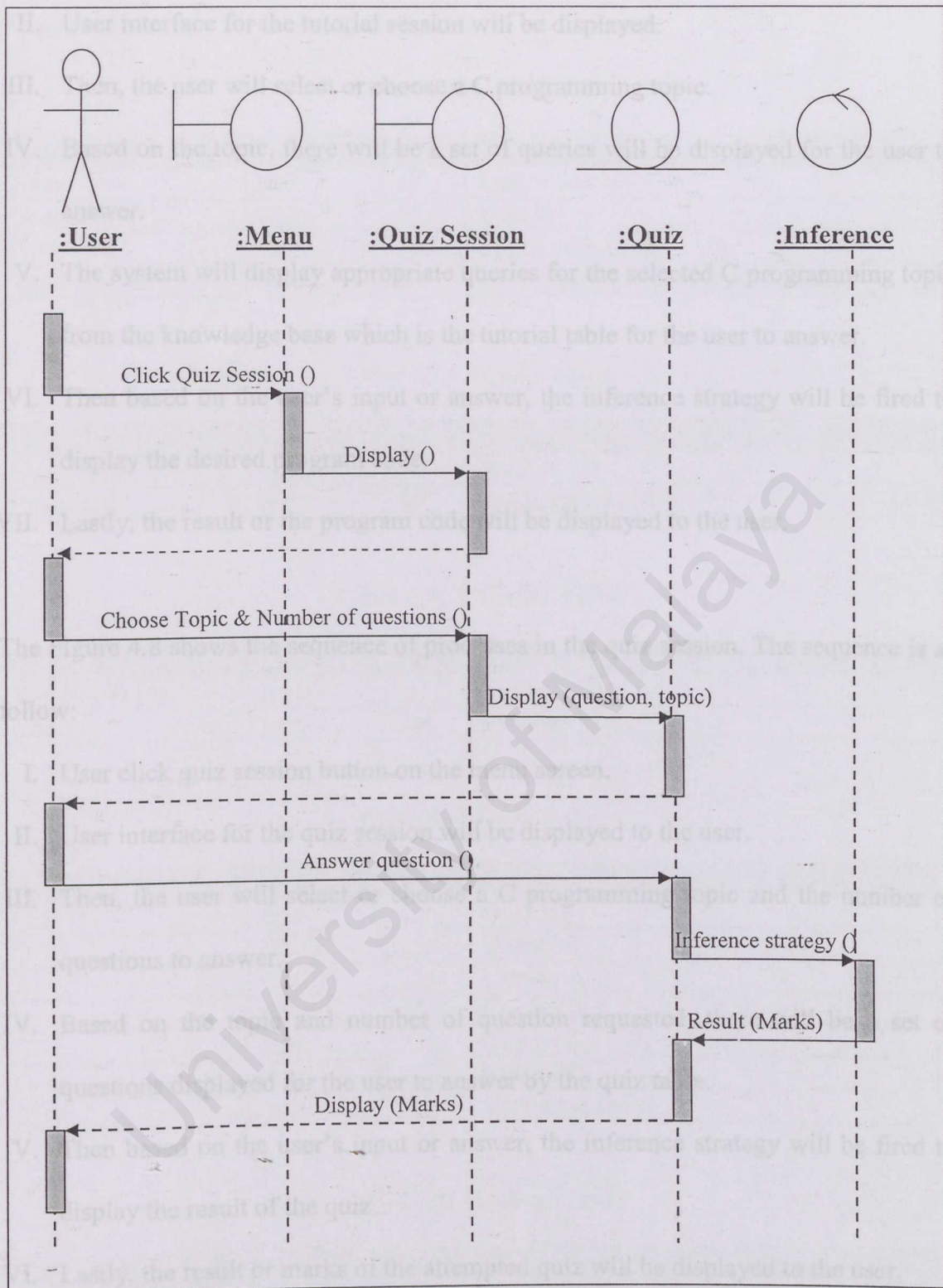


Figure 4.8: Overall Quiz Session Sequence Diagram

The Figure 4.7 shows the sequence of processes in the tutorial session. The sequence is as follow:

- I. User click tutorial session button on the menu screen.

- II. User interface for the tutorial session will be displayed.
- III. Then, the user will select or choose a C programming topic.
- IV. Based on the topic, there will be a set of queries will be displayed for the user to answer.
- V. The system will display appropriate queries for the selected C programming topic from the knowledge base which is the tutorial table for the user to answer.
- VI. Then based on the user's input or answer, the inference strategy will be fired to display the desired program code.
- VII. Lastly, the result or the program code will be displayed to the user.

The Figure 4.8 shows the sequence of processes in the quiz session. The sequence is as follow:

- I. User click quiz session button on the menu screen.
- II. User interface for the quiz session will be displayed to the user.
- III. Then, the user will select or choose a C programming topic and the number of questions to answer.
- IV. Based on the topic and number of question requested, there will be a set of questions displayed for the user to answer by the quiz table.
- V. Then based on the user's input or answer, the inference strategy will be fired to display the result of the quiz.
- VI. Lastly, the result or marks of the attempted quiz will be displayed to the user.

4.4 Suggested Database Design

There are few tables created in order to be used in order to process data the CTutorial4u using knowledge base system.

(*) primary key

Table 4.3: User Table

Field Name	Data type	Size	Description
Name	text	50	User's name
Username (*)	text	50	User ID
Password	text	20	Login password
Type	text	20	User type (student/ Administrator)

Table 4.4: Admin Table

Field Name	Data type	Size	Description
Username (*)	text	50	Admin' ID
Password	text	20	Login password

Table 4.5: Tutorial Table

Field Name	Data type	Size	Description
TutorialID (*)	text	50	Tutorial ID
Code	text	255	Program code

Table 4.6: Quiz Table

Field Name	Data type	Size	Description
QID (*)	text	50	Quiz ID
QUE	text	255	Question
A	text	255	Choice A
B	text	255	Choice B
C	text	255	Choice C
D	text	255	Choice D
ANS	text	255	Correct Answer

Table 4.7: Student Table

Field Name	Data type	Size	Description
StudentID(*)	text	50	Student ID
Name	text	100	Student Name
QResult	text	50	Quiz result
QGrade	text	255	Quiz Grade

Table 4.8: Explanation Table

Field Name	Data type	Size	Description
ExplanationID(*)	text	50	Explanation ID
Explanation	memo	3000	Explanation content

4.5 Interface Design

The ‘Ten Usability Heuristics’ by Jacob Nielsen [8] is taken into account to design CTutorial4u. These are ten general principles for user interface design. They are called "heuristics" because they are more in the nature of rules of thumb than specific usability guidelines.

▪ Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time. Every page has their respective name on the top to indicate to the users they are at which page or transaction.

▪ Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order. In term of this principle, the system is designed in English language and targeted users are students of Computer Science and Information Technology.

▪ User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. CTutorial4u users can exit from the page or system if they don't wish to continue.

▪ Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions. All the pages are consistently design where the structure of the navigational bar in every page is designed to be at the bottom.

▪ **Error prevention**

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. But at the same time, user's unpredicted actions sometimes require the user to cater for error messages. The Ctutorial4u was designed with high consideration given on how to avoid an error in the system and at the same time if error occurs how the users can accelerate from it.

▪ **Recognition rather than recall**

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions to use the system should be visible or easily retrievable whenever appropriate. Ctutorial4u is a simple windows-based system and to help the users to easily understand what they supposed to do next there will be instructions provided at some point of time otherwise the student will easily understand what to do.

▪ **Flexibility and efficiency of use**

Accelerators unseen by the novice user may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions. CTutorial4u is designed in such a way that it can be easily used by first year student who do not know what is C programming as a learning tool and also intended to be a reference for the tutors and lecturers.

- **Aesthetic and minimalist design**

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility. The warning and error messages are straight forward and simple to understand to avoid the misinterpretation among the users.

- **Help users recognize, diagnose, and recover from errors**

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution. As an example, there are error messages or information messages in tutorial session when there is lack of information to process the logic to deliver the program code to the user.

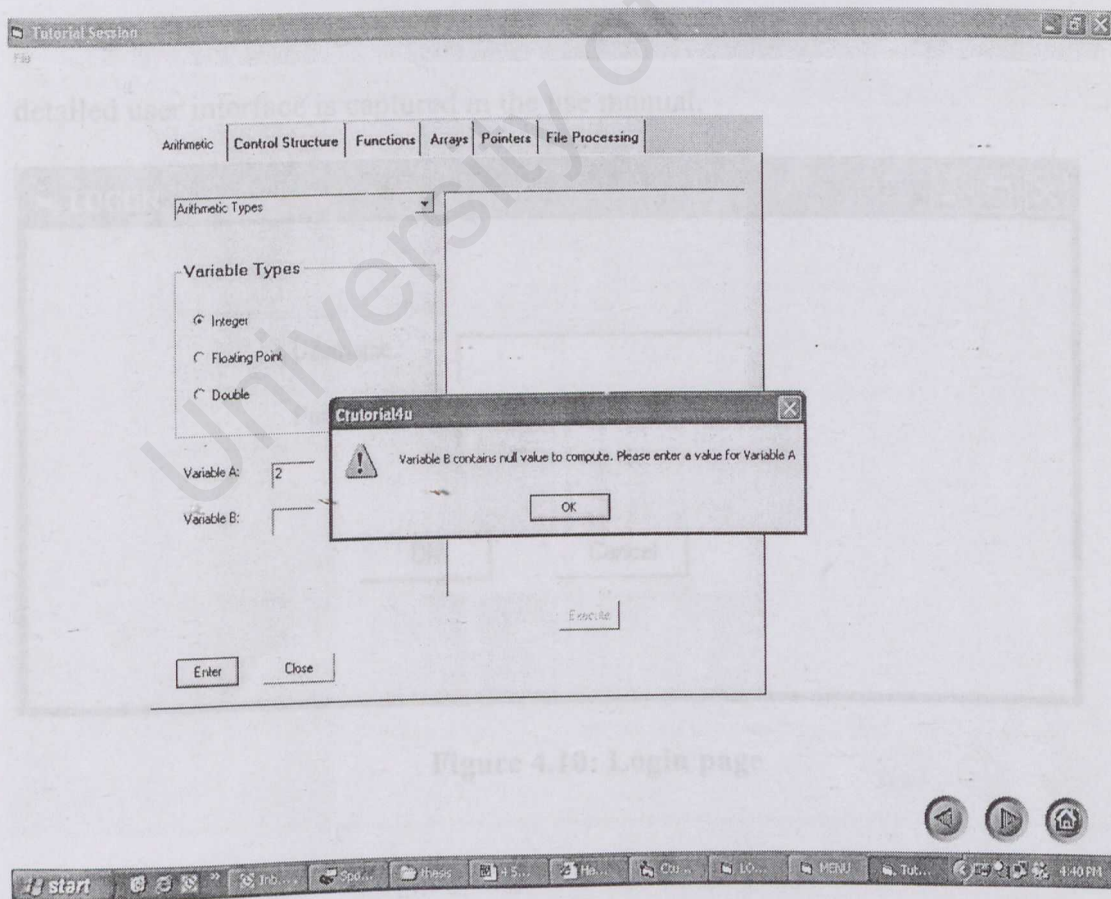


Figure 4.9: Sample of error message in the Tutorial Session

The warning message in Figure 4.9 informs the user that one of the variable field contains null value and suggests what to do to diagnose and recover from the problem.

- **Help and documentation**

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large. There is a summary about the CTutorial4u, the contents of the CTutorial4u and technical support who is the person to contact for further information about the system. All these are listed in the menu bar.

4.5.1 Sample User Interfaces

The Figure 4.10 until Figure 4.17 is the sample screen shots of the system. The rest and detailed user interface is captured in the use manual.

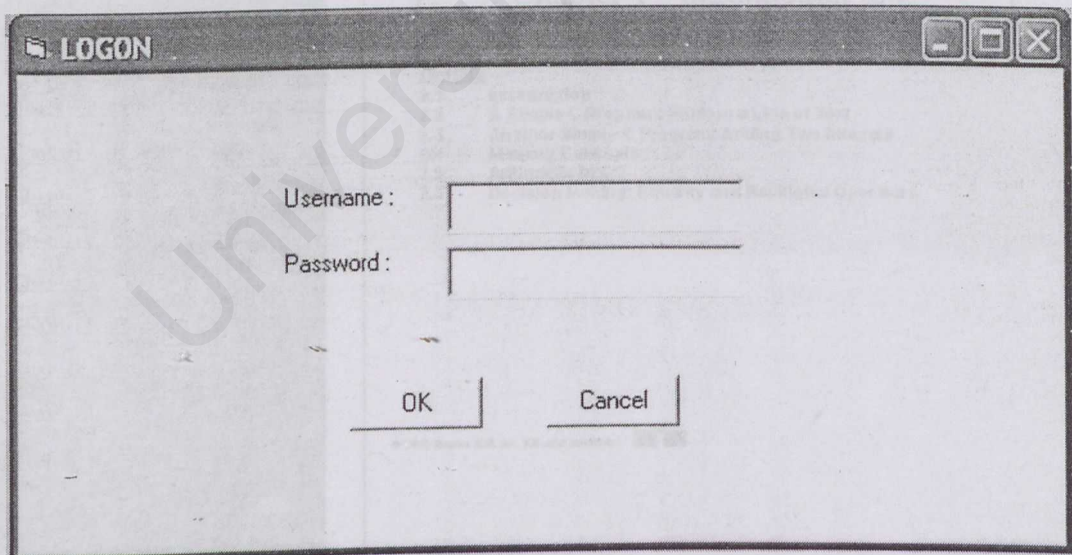
A screenshot of a Windows-style dialog box titled "LOGIN". The dialog box has a standard title bar with minimize, maximize, and close buttons. Inside the dialog, there are two text input fields. The first field is labeled "Username:" and the second field is labeled "Password:". Below these fields are two buttons: "OK" and "Cancel". The background of the dialog box is light gray, and the text is in a standard sans-serif font.

Figure 4.10: Login page

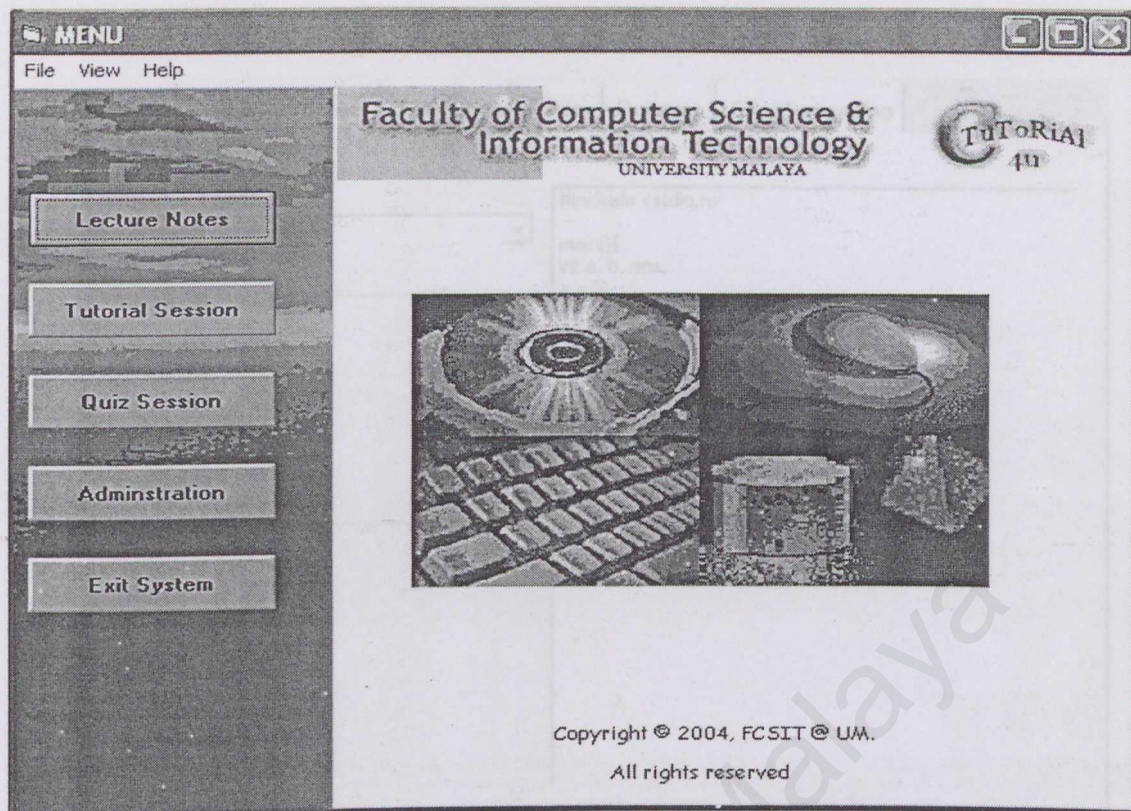


Figure 4.11: Main menu page

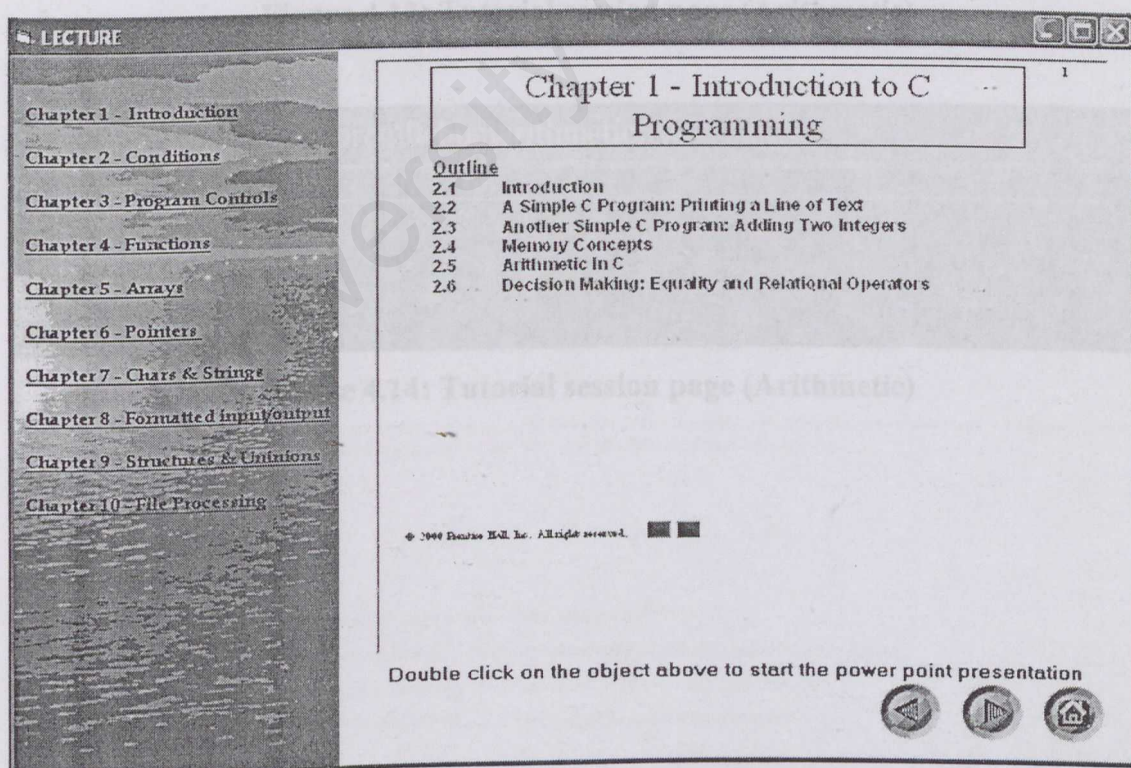


Figure 4.12: Lecture notes page

Arithmetic
Control Structure
Functions
Arrays
Pointers
File Processing

Arithmetic Type:
Multiplication

Variable Types

☒ Integer
☐ Floating Point
☐ Double

Variable A: 5
Variable B: 7

Enter
Close

```
#include <stdio.h>

main(){
int a, b, ans;
a = 5;
b = 7;
ans = a*b;
printf("Here is the answer...\n")
printf("%d\n",ans);

}
```

Execute

Figure 4.13: Tutorial session page (Arithmetic)

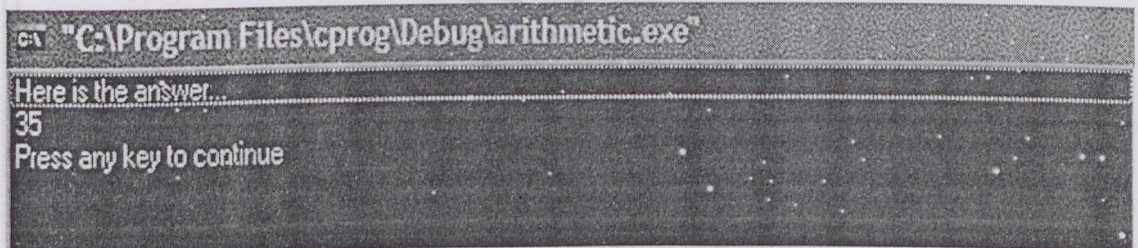


Figure 4.14: Tutorial session page (Arithmetic)

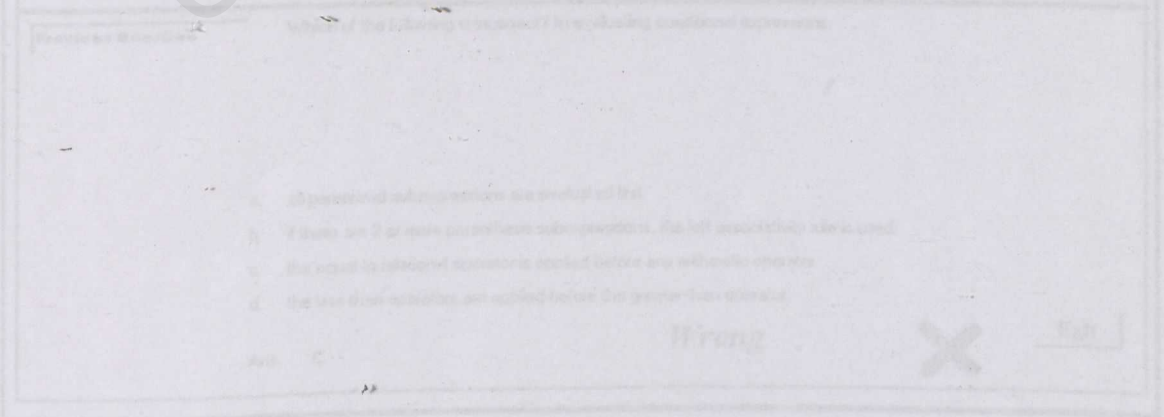


Figure 4.16: Quiz to Check Choice page

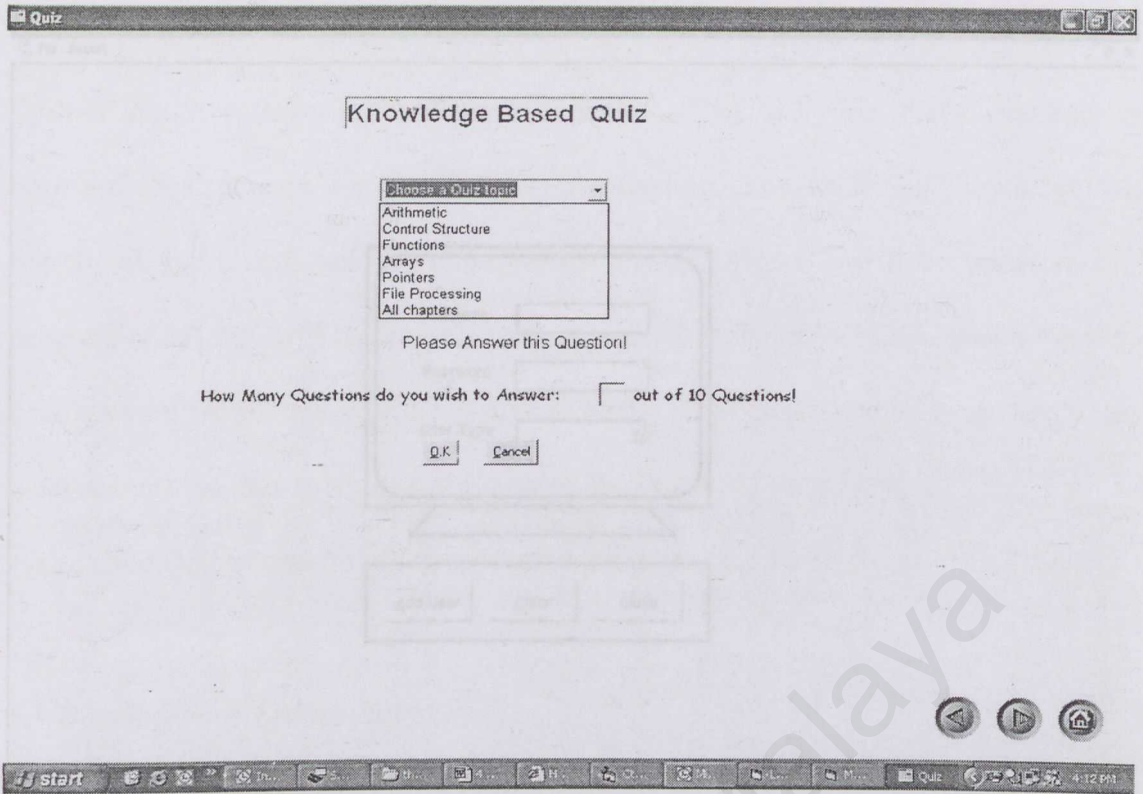


Figure 4.15: Quiz session page

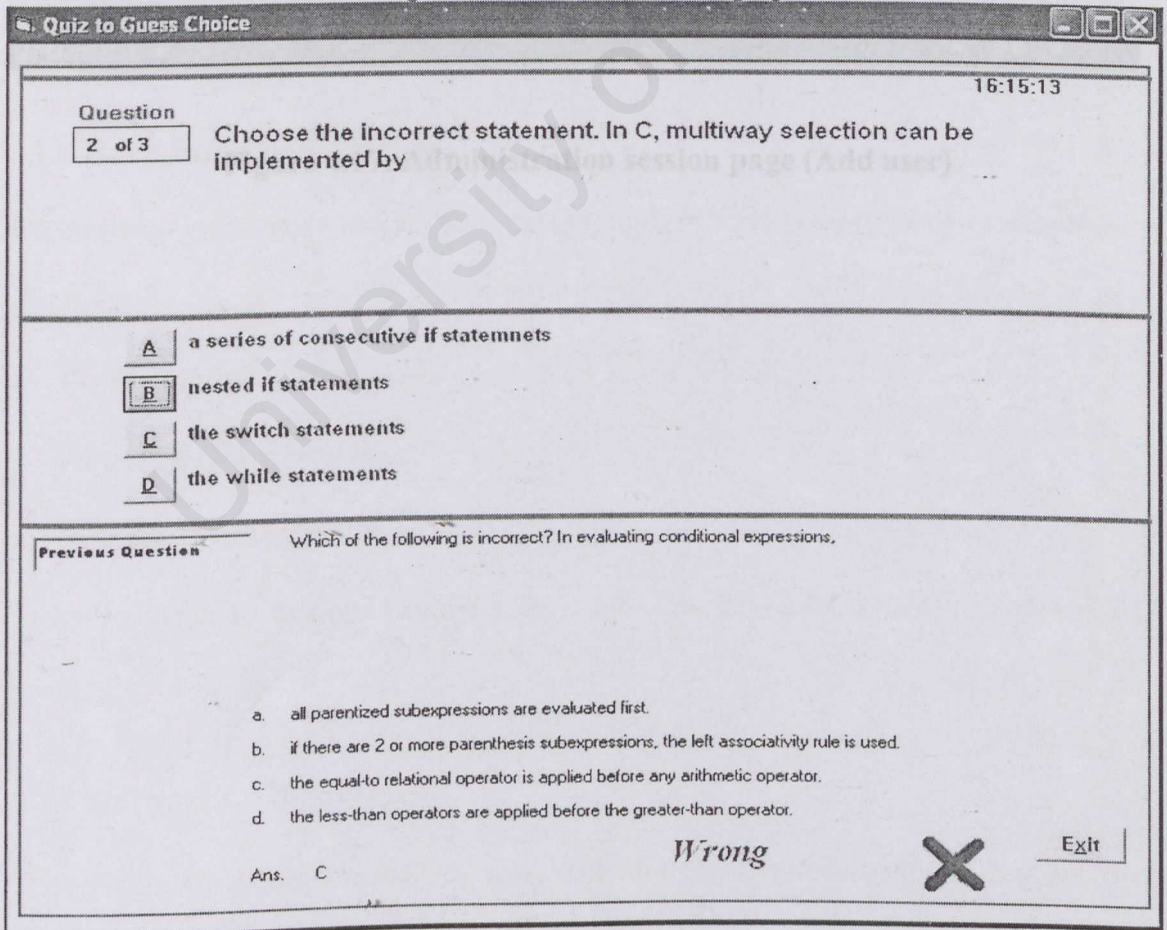


Figure 4.16: Quiz to Guess Choice page

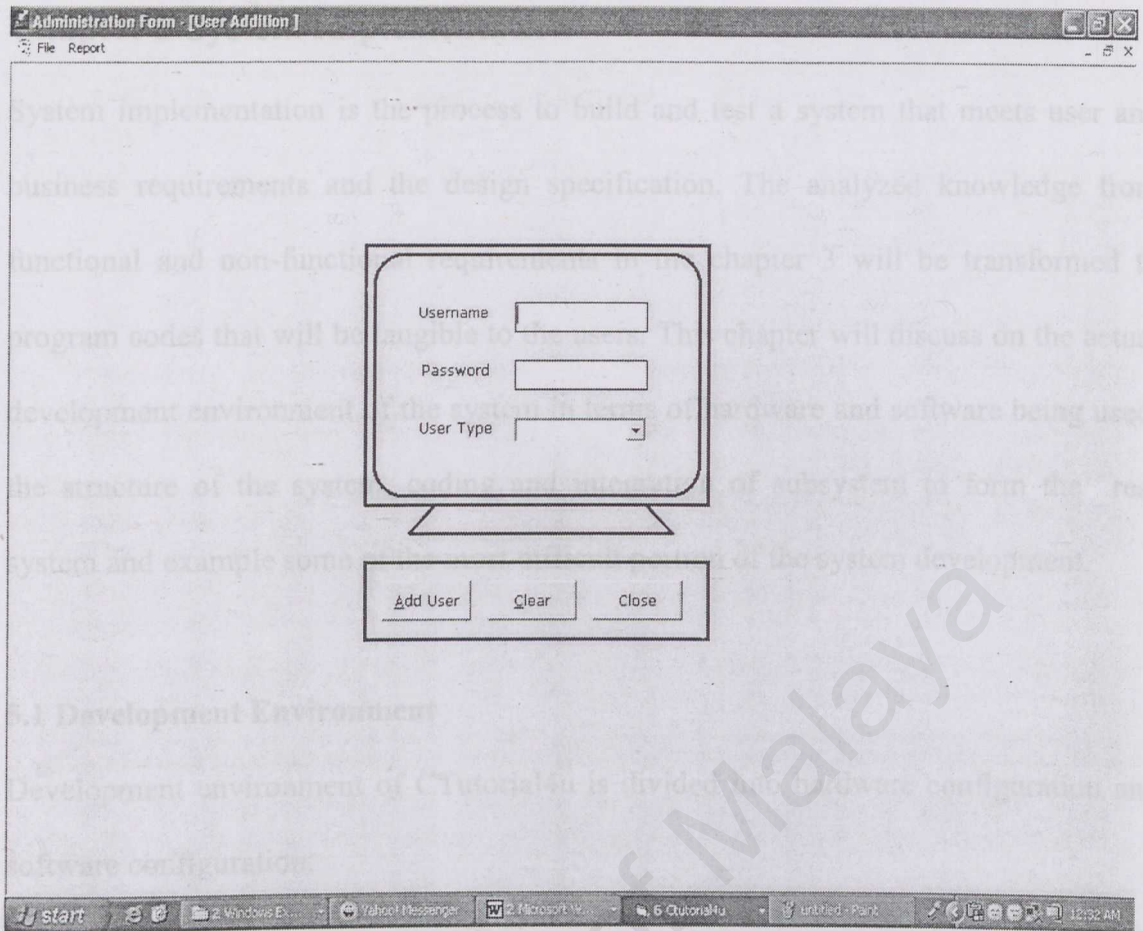


Figure 4.17: Administration session page (Add user)

5.1.1 Hardware

Tutorial4u would interface with the following hardware requirements as discussed in chapter 3:

- Pentium II 300 MHz processor or higher
- At least 2 GB of hard disk
- At least 32 MB of RAM
- Other standard computer peripherals – monitor, keyboard, mouse, printer and CD-ROM.

5.1.2 Software Configurations

The system remains to use the following software types to develop the system and there are some minor changes to the software being used to develop this system.

Chapter 5 System Implementation

System implementation is the process to build and test a system that meets user and business requirements and the design specification. The analyzed knowledge from functional and non-functional requirements in the chapter 3 will be transformed to program codes that will be tangible to the users. This chapter will discuss on the actual development environment of the system in terms of hardware and software being used, the structure of the system, coding and integration of subsystem to form the real system and example some of the most difficult portion of the system development.

5.1 Development Environment

Development environment of CTutorial4u is divided into hardware configuration and software configuration.

5.1.1 Hardware Configurations

Ctutorial4u would interface with the following hardware requirements as discussed in chapter 3:

- Pentium II 100 MHz processor or higher
- At least 2 GB of hard disk
- At least 32MB of RAM
- Other standard computer peripherals – monitor, keyboard, mouse printer and CD-ROM.

5.1.2 Software Configurations

The system remains to use the following software types to develop the system and there are some minor changes in the software being used to develop this system.

Table 5.1: CTutorial4u Software Configuration

Software Type	Required Software	Remarks
Operating System	Windows 98	no changes to the OS.
Database	MS Access 2000	- MS Access 2000 remains to be the database tool but after saving the data it needs to be converted to prior version which is Access 97 to be able to communicate with VB 6.0 interface.
Software Development Tool	Visual Basic 6.0 (VB6) Adobe Photoshop 7.0 MS Image Composer MS Office 2000 Ms Project MS Visual C++ 6.0	No-changes to the software development tool. Added Ms Project 2000 as a tool to develop the project schedule.

5.2 Project Development

Project development started with literature review, requirement analysis, system design and output from this phases made the developer to proceed to next stage of development phase which is the implementation using the tools analyzed in the previous phases. The right tool is the most essential part of the development phase. There was some problem when accessing and reading the data from Microsoft Access 2000 from Visual Basic 6.0 interface. So, it is needed to be changed from Ms Access 2000 to Ms Access 97. After changing to prior version, the interface works perfectly to call the data from the database.

5.2.1 Data Preparation

Data required to develop this system is collected through questionnaire from the students and the requirements is listed in chapter 3 (System Analysis). The data collected from questionnaire is only the functionalities of the system. The most

important part of this system is the tutorial session, so the essential requirement is to get the if-then rules right. Apart from this data, suitable still images and graphics to be included in the system were collected by browsing and surfing the internet. Data required is important to be in hand before starting to build a system to avoid delay in the deliverables.

5.2.1.1 Still Images and animated graphics

Most of the images are related to education and computer since CTutorial4u is a educational package.

5.2.1.2 Database preparation

Based on the data collected from questionnaire and similar projects, essential data are listed and Microsoft Access 2000 is used to create the database tables which will be called in the VB codes to display the data. (Table 2.2, 2.3 to 2.8) in chapter 2 were converted into database tables in MS Access 2000 as if-then rules tables for arithmetic, control structures, functions, arrays, pointers and file processing chapters in tutorial session. Since, it is a client-based system and standalone so Microsoft Access 2000 is sufficient enough to cater the system.

5.2.1.3 Input form design

Prototypes with minimum coding of the interface were designed to get the feedbacks from the users. The prototypes then shown to selected students and colleagues to get comments on the design. Their comments is considered when implementing the real system and started with coding phases.

5.2.1.4 User interface design

User interface developed based on the comments and tested with the same students to verify their satisfaction. The ‘Ten Usability Heuristics’ by Jacob Nielsen [8] was taken as the guideline to design the user interfaces. Based on the functional and non-functional requirement sated in chapter 3, the system interfaces were developed. All the screens is clearly depicted in Appendix A, user manual.

5.2.2 Coding

Coding is the process of transforming the program logic into written codes without any errors. Coding phase is where design phase is translated into computer programming codes. Firstly, the pseudo code will be written in paper to check the logic and if its logic sounds workable then it will be coded using Visual Basic 6.0. Microsoft Visual Basic 6.0 It is easy to be learned and deployed by novice users. The coding phase is tedious and need to be done carefully because it plays a very important role in providing effective input to the information system using techniques of good screen design. The example of coding in tutorial session for the topic of arithmetic is shown in Figure 5.1, control structure in Figure 5.2 and function in Figure 5.3.

Figure 5.1: Abstract of ‘Arithmetic’ Algorithm in the Tutorial session

```

Sub Arithmetic()
Dim Response As String
Dim s As New Stream
Dim str
Dim fso As New FileSystemObject, txtfile, _
    fill As File, ts As TextStream
Set fill = fso.GetFile(App.Path & "\ & "Arithmetic.txt")
Set ts = fill.OpenAsTextStream(ForReading)
List1.clear
While (Not ts.AtEndOfStream)
str = ts.ReadLine
If str = "main()" Then
    List1.AddItem "main()"
    If rdoInteger.Value Then
        List1.AddItem "int a, b, ans;"
    ElseIf rdofloating.Value Then
        List1.AddItem "float a, b, ans;"
    ElseIf rdoDouble.Value Then
        List1.AddItem "double a, b, ans;"
    End If
    List1.AddItem "a = " + txtVarA.Text + ";"
    List1.AddItem "b = " + txtVarB.Text + ";"
    ArithType = cboArith.ListIndex
    If ArithType = 0 Then
        List1.AddItem "ans = " + "a" + "*" + "b;"
    ElseIf ArithType = 1 Then
        List1.AddItem "ans = " + "a" + "/" + "b;"
    ElseIf ArithType = 2 Then
        List1.AddItem "ans = " + "a" + "+" + "b;"
    ElseIf ArithType = 3 Then
        List1.AddItem "ans = " + "a" + "-" + "b;"
    ElseIf ArithType = 4 Then
        List1.AddItem "ans = " + "(a + " + "b)" + "/2;"
    End If
    List1.AddItem "printf("""Here is the answer...\n" + """)"
    If rdoInteger.Value Then
        List1.AddItem "printf("""%d\n""",ans);"
    ElseIf rdofloating.Value Then
        List1.AddItem "printf("""%f\n""",ans);"
    ElseIf rdoDouble.Value Then
        List1.AddItem "printf("""%lf\n""",ans);"
    End If
    Else
        List1.AddItem str
    End If
Wend

```

Figure 5.1: Abstract of ‘Arithmetic’ Algorithm in the Tutorial session


```

Sub CtrlStruct()
Dim s As New Stream
Dim str
Dim str1
Dim str2
Dim fso As New FileSystemObject, txtfile, _
    fil1 As File, fil2 As File, fil3 As File, ts As TextStream, ts2 As TextStream, ts3 As
TextStream
List4.clear
Set fil1 = fso.GetFile(App.Path & "\" & "if.txt")
Set fil2 = fso.GetFile(App.Path & "\" & "switchstruc.txt")
Set fil3 = fso.GetFile(App.Path & "\" & "ifelse.txt")

Set ts = fil1.OpenAsTextStream(ForReading)
Set ts2 = fil2.OpenAsTextStream(ForReading)
Set ts3 = fil3.OpenAsTextStream(ForReading)
List4.clear

    If cboselection.ListIndex = 0 Then
        While (Not ts.AtEndOfStream)
            str = ts.ReadLine
            List4.AddItem str
        Wend

    ElseIf cboselection.ListIndex = 1 Then
        While (Not ts3.AtEndOfStream)
            str2 = ts3.ReadLine
            List4.AddItem str2
        Wend

    ElseIf cboselection.ListIndex = 2 Then

        While (Not ts2.AtEndOfStream)
            str1 = ts2.ReadLine
            List4.AddItem str1

        Wend

    End If

    ts.Close
    ts2.Close
    ts3.Close
End Sub

```

Figure 5.2: Abstract of 'Control structure' Algorithm in the Tutorial session

```

Sub Function()
Dim s As New Stream
Dim str
Dim fso As New FileSystemObject, txtfile, _
    fil1 As File, ts As TextStream
Set fil1 = fso.GetFile("C:\Tutorial4u\function.txt")
Set ts = fil1.OpenAsTextStream(ForReading)
List2.clear
While (Not ts.AtEndOfStream)
    str = ts.ReadLine

    If str = "int main()" Then

        List2.AddItem "int " + cbomaxmin.Text + "(int , int, int)"
        List2.AddItem "int main (){"
        List2.AddItem "int a, b, c;"
        List2.AddItem "printf(" + """"Enter integers: """)"
        List2.AddItem "scanf(" + """"%d" + "%d" + "%d" + """" , &a, &b &c);"
        List2.AddItem "printf("" + cbomaxmin.Text + "is:" + "%d\n"" + "," +
cbomaxmin.Text + "( a, b, c );"
        List2.AddItem "int(int x, int y, int z){"
        List2.AddItem "int ans = w;"
        List2.AddItem "if ( x" + txtFuncSym.Text + "ans )"
        List2.AddItem "ans = x"
        List2.AddItem "if ( y" + txtFuncSym.Text + "ans )"
        List2.AddItem "ans = y"
        List2.AddItem "if ( z" + txtFuncSym.Text + "ans )"
        List2.AddItem "ans = z"

    Else
        List2.AddItem str
    End If

Wend

ts.Close

End Sub

```

Figure 5.3: Abstract of 'Function' Algorithm in the Tutorial session

5.2.2.1 Database connection *message will require the user to enter correct username*

The database to be connected is globally defined in the module as depicted in the Figure 5.4. The database is defined in the Data Sources (Open Database Connectivity ODBC) at the System DSN as CTutorial which the location is at C:\CTutorial4u\CI.mdb. So, every time to open the connection to the database just need to call the function conn_open and conn_close to close the connection. This will ease the developer's programming task which does not require the developer to define in every form.

```
Public conn As ADODB.Connection
Public rs As ADODB.Recordset
Public rsques As New ADODB.Recordset
Public rspoint As New ADODB.Recordset
Public Arithmetic As Integer
Public logout_hour
Public logout_min
Public logout_sec
Public duration

Public Sub conn_open()
Set conn = New ADODB.Connection
conn.Open "CTutorial"
End Sub

Public Sub conn_close()
conn.Close
End Sub
```

Figure 5.4: Module page

5.2.2.2 Authenticate member

Figure 4.10 shows the login form in the page 121 where this is the page which authenticates the users to access the system. If either the username or password supplied by the user does not match with those records in the database, the system will display a warning message that incorrect logon and as the remedy or help to recover

from the problem system the message will require the user to enter correct username and password. The users who are not authorized need to contact the administrator to add them as the authorized user.

5.2.2.3 Process with Database

The Figure 5.5 is the abstract of command add question in frmQuestion form where the administrator add the question to arithmetic table after choosing arithmetic chapter in combo box cboQuizchap. Firtsly, conn_open is called to connect the C1 database and select statement is used to connect the arithmetic table and reques is the recordset defined globally in the module.

Example of VB6.0 code:

```
rsques.Open "select * from Arithmetic where QID = '" & txtQID & "'", conn,  
adOpenDynamic, adLockOptimistic
```

Figure 5.5: Abstract of 'Add New Question' in Administration session


```
Private Sub CmdAdd_Click()
```

```
conn_open
```

```
If cboQuizchap.ListIndex = 0 Then
```

```
rsques.Open "select * from Arithmetic where QID = '" & txtQID & "'", conn,  
adOpenDynamic, adLockOptimistic
```

```
'rsAdmin.Open "select * from User where Username = '" & Username1 & "'", conn,  
adOpenDynamic, adLockOptimistic
```

```
    If rsques.EOF = True Then
```

```
        rsques.AddNew
```

```
        rsques!QID = txtQID.Text
```

```
        rsques!Que = txtQuestion.Text
```

```
        rsques!A = txtA.Text
```

```
        rsques!B = txtB.Text
```

```
        rsques!C = txtC.Text
```

```
        rsques!D = txtD.Text
```

```
        rsques!ans = txtanswer.Text
```

```
        rsques.Update
```

```
'rsAdmin!UserName = Username1.Text
```

```
'rsAdmin!Password = Password1.Text
```

```
'rsAdmin.Update
```

```
    End If
```

```
rsques.MoveNext
```

```
'rsAdmin.MoveNext
```

```
MsgBox "Question Successfully Added..."
```

```
txtQID.Text = ""
```

```
txtQuestion.Text = ""
```

```
txtA.Text = ""
```

```
txtB.Text = ""
```

```
txtC.Text = ""
```

```
txtD.Text = ""
```

```
txtanswer.Text = ""
```

```
rsques.Close
```

```
End If
```

```
    conn_close
```

```
End Sub
```

Figure 5.5: Abstract of 'Add New Question' in Administration session

```

(General) rndmz

frmgrid.Show vbModal 'Display the frmgrid only
End Sub

Public Sub rndmz()
Dim rndnum As Integer 'To store the random number value
Dim cho As Integer 'To store the choice value from the obtained total number records
Visible 'Call the function visible
    Data1.Recordset.MoveFirst 'Move the record pointer to first record
    cho = Data1.Recordset.RecordCount 'Count the total number of records in the
    cho = cho - 1 'Subtract 1 from the total number of records
    Randomize 'This is a library function which is used to randomize
    rndnum = Int((cho * Rnd) + 1) 'Obtain the random number and store it in the rnd
    Data1.Recordset.Move rndnum 'Move the record pointer to the random numbers
    If cnt = Data1.Recordset.RecordCount Then
MsgBox "Sorry Only " & frmQuizArith.Data1.Recordset.RecordCount & " Questions Available
Unload Me
Load frmgrid
frmgrid.Show vbModal
End If
End Sub

Public Sub random()
cnt = cnt + 1 'Count the number of questions
lblQueno.Caption = cnt + 1 'To display the number of current question
If cnt < Val(Label15.Caption) Then 'To check whether the question is less than user cho:
rndmz 'Call the rndmz sub program
Else
Visible 'call the visible sub program
Label15.Visible = False 'Disable the label

```

Figure 5.6: Abstract of Rndmz function in the Quiz session

The function rndmz shown in the figure 5.6 uses data control which defines the database name for Data1 to be C:\CTutorial4u\C1.mdb and record source to be Arithmetic table. Once connected to the database, it will move to the first record and count total number of records. Then using the random function (Rnd) the function randomize and store the random number in memory and display the next question as the question of random number. This is how the data control is being used as database connection.

5.2.3 System Integration

System integration is the process of combining the all the subsystems together to form a complete system. Once all the pages are designed then connection of one page to another is done by using command buttons and navigational buttons.

Since, CTutorial4u developed phase by phase, so each phase especially this linker were tested.

Once it is determined that all the section or pages are error free, all these sections are combined to test the interaction with each other. This is to ensure the navigation between these sections will be error free. Periodic testing was done with the project supervisor to determine that the system is user friendly and attractive. Furthermore, this ensures that the system fulfills the required scope. Lastly, the system was tested among the students and colleagues for further improvement.

6.3 Testing Phase

The testing strategies are designed to test the CTutorial4u works as required by the users and eliminate any unwanted bugs during the quiz session or tutorial session.

Testing phases includes Unit testing, sub-system or module testing and overall system testing.

6.2.1 Unit Testing

Unit testing is the process of testing smaller building blocks which means individual programs or subroutines is tested while the code is being written. It is important to test the program even a simple if-then-else rule before proceeding to another step or code. Incremental testing is being used to test the every single sub routines and functions.

Chapter 6 System Testing

6.1 Introduction to System Testing

Testing is very important to make sure a system executes free of error and display the desired output. Early detection of error will ensure less effort to detect in the later part. Since, CTutorial4u developed phase by phase, so each phase especially the links were tested.

Once it is determined that all the section or pages are error free, all these sections are combined to test the interaction with each other. This is to ensure the navigation between these sections will be error free. Periodic testing was done with the project supervisor to determine that the system is user friendly and attractive. Furthermore, this ensures that the system fulfils the required scope. Lastly, the system was tested among the students and colleagues for further improvement.

6.2 Testing Phase

The testing strategies are imperative to test the CTutorial4u works as required by the users and eliminate any unwanted bugs during the quiz session or tutorial session. Testing phases includes unit testing, sub-system or module testing and overall system testing.

6.2.1 Unit Testing

Unit testing is the process of testing smaller building blocks which means individual programs or subroutines is tested while the code is being written. It is important to test the program even a simple if-then-else rule before proceeding to another step or code. Incremental testing is being used to test the every single sub routines and functions.

6.2.2 Module Testing

Module testing is testing every function in a module or a page to identify any errors within that module. Every form or page of the CTutorial4u is tested until there are no bugs in terms of its functionalities. The testing approach used in this system 'Bottom Up' approach where the system was tested from small module and when there was no error then only the whole system was tested.

6.2.3 Overall system Testing

Overall system testing is full blonde test of the system to measure its full capabilities and to uncover its limitations. The system testing is done via integration testing which uses 'Sandwich Technique' that is a combination of 'Top-Down' approach and 'Bottom-Up' approach. This way is a more effective way where a single function is tested through both directions through main function to smaller blocks of units and from smaller units of function to main functions. The technique of 'Bottom Up' testing is illustrated in Figure 6.1 where each one of the function is tested individually before combining them to become a working system. The CTutorial4u was installed in one of the laboratories of FCSIT, UM for the tutors and the students to test. The test script was used to derive the user acceptance of the system.

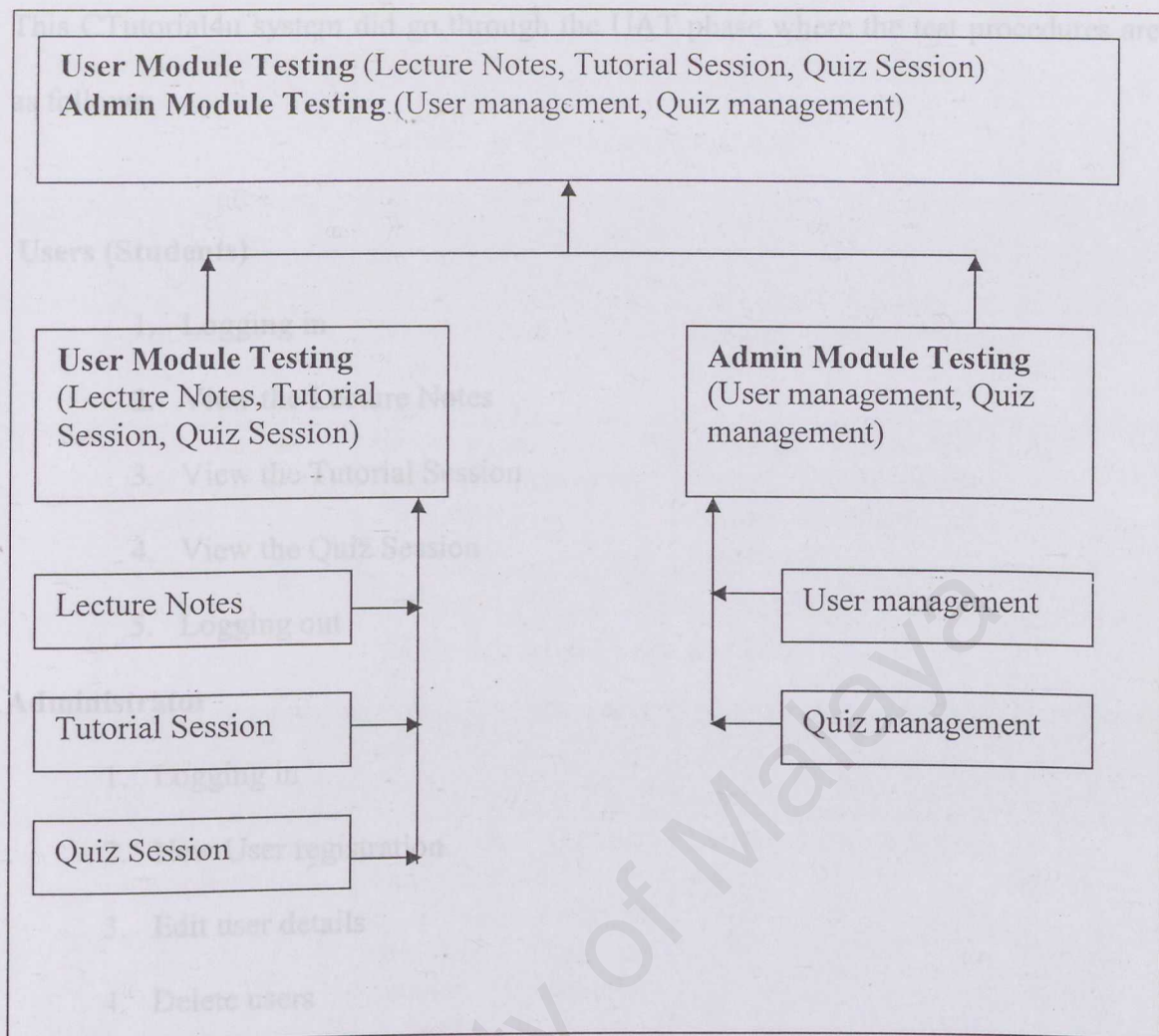


Figure 6.1: Bottom up Testing

6.2.4 Test Case

User Acceptance Test (UAT) is the most essential part in the project hand over phase. In large commercial system developments once UAT is being signed by the users then it means the system has been tested and accepted. Firstly the proper test scenarios or cases will be identified. Then there are many stages in conducting UAT as well where there will be first round of UAT to identify the errors or bugs and the comments given will be reviewed to modify the system accordingly. The second UAT is to verify by the users that the comments given in the first UAT has been adjusted and amended in the system.

This CTutorial4u system did go through the UAT phase where the test procedures are as follows:

Table 6.1: Lagon page test case

Users (Students)		Expected Output	Action
1.	Logging in	Display the menu page	OK ✓
2.	View the Lecture Notes		
2.	View the Tutorial Session	Message: Incorrect login. Please enter correct username & password	OK ✓
4.	View the Quiz Session		

5. Logging out

Table 6.2: Menu page test case

Administrator

1.	Logging in		
2.	New User registration	Display Lecture notes page	OK ✓
3.	Edit user details		
4.	Delete users		
5.	Add new questions	Display Tutorial Session page	OK ✓
6.	Modify questions	Display Quiz Session page	OK ✓
7.	Delete questions	Display Administration page	OK ✓
8.	Logging out		

6.2.4.1 User Module Test Case

Table 6.1: Logon page test case

Test #	Scenario	Procedure	Expected output	Actual Output / Remarks
1.	Authorized User login to the system	Enter : Thava Password:** Click [OK]	Display the menu page	OK. √
2.	Unauthorized user login to the system	Enter : user Password: password Click [OK]	Message: Incorrect logon. Please enter correct username & password	OK. √

Table 6.2: Menu page test case

Test #	Scenario	Procedure	Expected output	Actual Output / Remarks
1.	View Lecture notes page.	1. Click [Lecture Notes] button. 2. Double click on the box after the chapter 1 information is displayed.	Display Lecture notes page.	OK. √
2.	View Tutorial session page.	Click [Tutorial session] button.	Display Tutorial Session page.	OK. √
3.	View Quiz session page.	Click [Quiz session] button.	Display Quiz Session page.	OK. √
4.	View the Admin page.	Click [Administration] button.	Display Administration page.	OK. √

Table 6.3: Lecture Notes page test case




Test #	Scenario	Procedure	Expected output	Actual Output / Remarks
1.	View chapter 1 slide show.	Click [Chapter 1: Introduction]. 2. Double click on the box after the chapter 1 information is displayed.	Power point show starts when double click on the box where the first page of chapter 1 is displayed.	OK. ✓
2.	Stop the slide show	Click [ESC] on the keyboard.	Power point show stops and back to the lecture notes page.	OK. ✓
3.	Go back	Click  navigation button.	Redirect to Menu page	OK. ✓
4.	Go next	Click  navigation button.	Display Tutorial session page.	OK. ✓
5.	Go Home /Menu	Click  navigation button.	Display Menu page.	OK. ✓

Table 6.4: Tutorial Session page test case







Test #	Scenario	Procedure	Expected output	Actual Output / Remarks
1.	Choose Arithmetic chapter.	1. Click [Arithmetic].	Arithmetic screen appears	OK. ✓
2.	Display Code	1. Choose Arithmetic type as multiplication. 2. Choose variable types as integer. 3. Variable A: 3 Variable B: 4	# include <stdio.h> main(){ int a, b, ans; a=3; b=4; ans=3*4; printf("Here is the answer"); printf ("%d",ans);}	OK. ✓
3.	Execute the code	Click [Execute] button	Execution result: 12	OK. ✓
4.	Go back	Click  navigation button.	Redirect to Menu page	OK. ✓
5.	Go next	Click  navigation button.	Display Quiz session page.	OK. ✓
6.	Go Home /Menu	Click  navigation button.	Display Menu page.	OK. ✓

Table 6.5: Quiz Session page test case

Test #	Scenario	Procedure	Expected output	Actual Output / Remarks
1.	Choose Arithmetic chapter.	1. Choose [Arithmetic] topic and 4 as the number of questions to answer.	Display Quiz page	OK. ✓
2.	Starts to attempt the quiz within 4 minutes.	1. Answer all the questions correctly.	1. Display the previous question and answers at the bottom with the correct. 2. Image ✓ displayed for correct answer. 3. Display total number of correct and wrong answers and percentage of the results.	OK. ✓
3	Timer	2. Timer is to 4 minutes. Wait for 4 minutes to elapse to test whether the timer works or not.	Exceeding 4 minutes will prompt timed out and exit form the page	OK. ✓
4.	Go back	Click  navigation button.	Redirect to Tutorial session page	OK. ✓
5.	Go next	Click  navigation button.	Display Administration page.	OK. ✓
6.	Go Home /Menu	Click  navigation button.	Display Menu page.	OK. ✓

6.2.4.2 Admin Module Test Case

Table 6.6: Administration page test case

Test #	Scenario	Procedure	Expected output	Actual Output / Remarks
1.	Add User	1. Choose [Add User] from the menu bar. 2. Enter Username: Aini Password: ** User type: student	Message: User created successfully	OK. √
2.	Add Question	1. Choose [Add Question] from the menu bar. 2. Enter: Question: trial A:a, B:b, C:c, D:d Ans:A	Message : Question added successfully	OK. √

6.3 Compliance of the system to its scope and requirements

The example of test scenarios tested by ten of the FCSIT’s student is listed in the table 6.1 until table 6.6. Based on the test results, it is concluded that the system meets the expected output and ready to be delivered. At the same time, the system meets the system requirements and scope of the system specified in the earlier stage of requirement analysis. A study, consisting of 1-hour lab session has been performed with ten computer science students. There were problems to get students to test the system and evaluate the functionality they are not willing to spend time to test the system. Despite this, there were ten students gathered to perform the testing. Three of the FCSIT’s tutors, Miss Wan Hasnira Wan Husin, Mr. Yeong Pong Meau and Miss Noor Aziamh Hassan were merely involved in testing the system as well. The students’ test results were recorded and the students filled out a questionnaire at the end of the

session. The users were welcomed to give comments about the system and suggestions to improve the systems. The suggestion will be rectified whether to be included in the system or not together with the project supervisor and if the more research required to include that functions then it will be added in the future enhancement. The subjective responses from the questionnaire revealed that the students enjoyed learning C programming with CTutorial4u and appreciated its adaptive features. The next chapter; System Evaluation will describe further regarding the evaluation of CTutorial4u using the evaluation form. The comparison between before and after using CTutorial4u is also evaluated in next chapter.

Table 7.1: Problems Encountered and Solutions

No.	Problem	Solution
1.	Cannot retrieve data from the record source of the database.	Converted MS Access 2000 to its prior version which is the MS Access 97.
2.	Difficulty in designing the Tutorial session in the tab format to reduce number of scroll with the same design. Confusing results.	Referred with other system developed in VB 6.0 which has the tab functions.
3.	Lecture notes to be in the power point format.	Inserted the reference power point and called the power point format in the code.
4.	Program code in the Tutorial session to be compiled in Visual Studio C compiler and display the output from C.exe.	Failed to call the Visual C compiler and if it is to be done then it will slow down the processing and increase the response time to the user. So as a consial the exact execution file is designed in Visual Basic 6.0 where interface design looks the same as how it looks in command prompt when C program is executed.
5.	Difficult to get the students to test and evaluate the system.	Installed in one of FCSIT's lab and tutors helped to get the students test the system.

Chapter 7 System Evaluation

7.1 Problems Encountered and Solutions

There won't be any system being developed without problems and the same goes for CTutorial4u where it is not without its fair share of problems. Firstly, not sure whether knowledge based system can be developed as standalone system and once it is verified with the supervisor Puan Nazean Jomhari then proceeded to develop the system in Microsoft Visual Basic 6.0 and Microsoft Access 2000 as the database tool. Other than that, there are other minimal problems encountered which had occurred during the coding and testing phase. Table 7.1 describes the problems and solution during coding and testing phase.

Table 7.1: Problems Encountered and Solutions

No.	Problems	Solution
1.	Cannot retrieve data from the record source of the database.	Converted MS Access 2000 to its prior version which is the MS Access 97.
2.	Difficulty in designing the Tutorial session in the tab format to reduce number of screen with the same design. Coding is unsure.	Referred with other system developed in VB 6.0 which has the tab functions.
3.	Lecture notes to be in the power point format.	Inserted the reference power point and called the power point format in the code.
4.	Program code in the Tutorial session to be compiled in Visual Studio C compiler and display the output from C.exe	Failed to call the Visual C compiler and if it is to be done then it will slow down the processing and increase the response time to the user. So as a remedial the exact execution file is designed in Visual Basic 6.0 where interface design looks the same as how it looks in command prompt when C program is executed.
5.	Difficult to get the students to test and evaluate the system.	Installed in one of FCSIT's lab and tutors helped to get the students test the system.

7.2 System Strengths

The functionalities such as user friendliness, password protected reliable system with error handling, interactive, simplicity and consistency and has validation on the user input. There is an evaluation form as depicted in Appendix B distributed to the users which was used to evaluate the system in terms of the functionalities described earlier. Most of the feedback received are very encouraging and favors the system considerably.

The Ctutorial4u was evaluated in terms of its contribution to help them in learning the system in easier way. When asked to rate how much they learned from working with the system, mostly rated 3 where the choices were from 1 to 5 (1 = Little, 5 = Very much). This is shown clearly in Figure 7.1. The majority of the students appreciated the exploratory, hands-on approach, learning their own pace and found that learning with CTutorial4u is more personal than lectures.

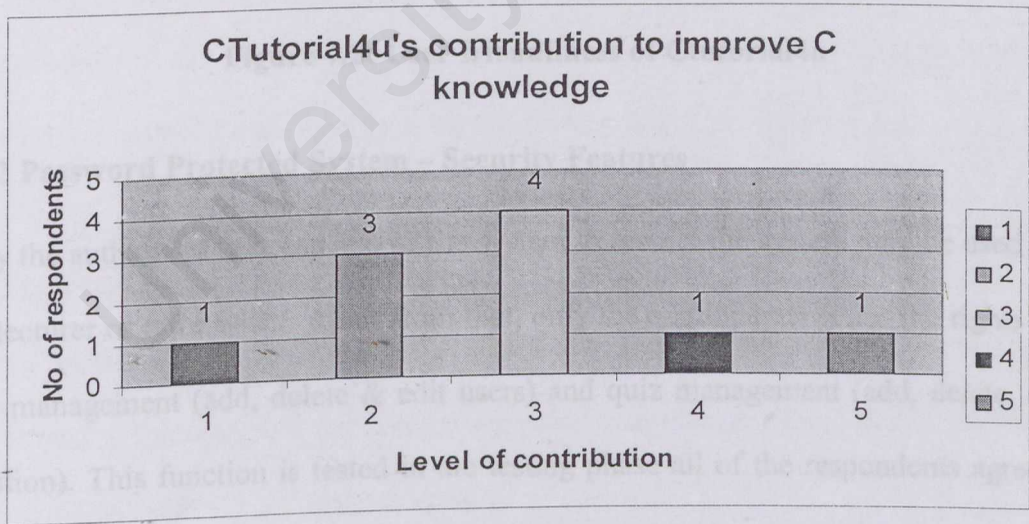


Figure 7.1: CTutorial4u’s contribution to improve C knowledge

7.2.1 User Friendliness

The system is user friendly with minimal of unrecognized format of button. The user can easily know what to do next when they are at one of the page. It has simple and

consistent user interface. The program flow of each sub-module was designed to be as easy as possible to avoid misunderstanding and confusion among the users. The evaluation form distributed, requires users to rate the system in terms of user friendliness. There were rating options of user friendliness from 1 to 5 (1=Not at all, 5=Very). Majority of the respondents rated Ctutorial4u as 4 were it's considered more than average rating. The Figure 7.2 shows the results of user's rating for Ctutorial4u in terms of user friendliness.

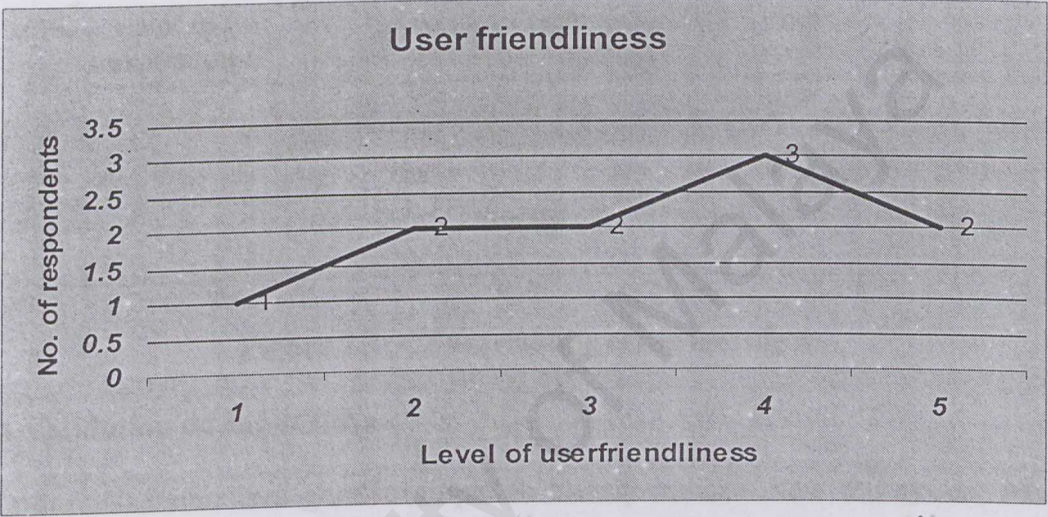


Figure 7.2: User friendliness of Ctutorial4u

7.2.2 Password Protected System – Security Features

Only the authorized user can access the system to restrict the system only be used when the lecturer or tutor wish. Apart from that, only the administration has the rights to do user management (add, delete & edit users) and quiz management (add, delete, &edit question). This function is tested in the testing phase all of the respondents agree that the system only allows authorized users to change and login to the system.

7.2.3 Reliable System with Effective Error Handling

All the error messages are effective and reliable where the error message also gives instruction to avoid the problem again. Figure 7.3 depicts the responds of users whether

they understood the system's error messages clearly or not during the testing. The 60% of the respondents said that the understood well but 40% disagrees that the error messages are understandable. In future enhancements phase, the system's error messages need to be given more importance.

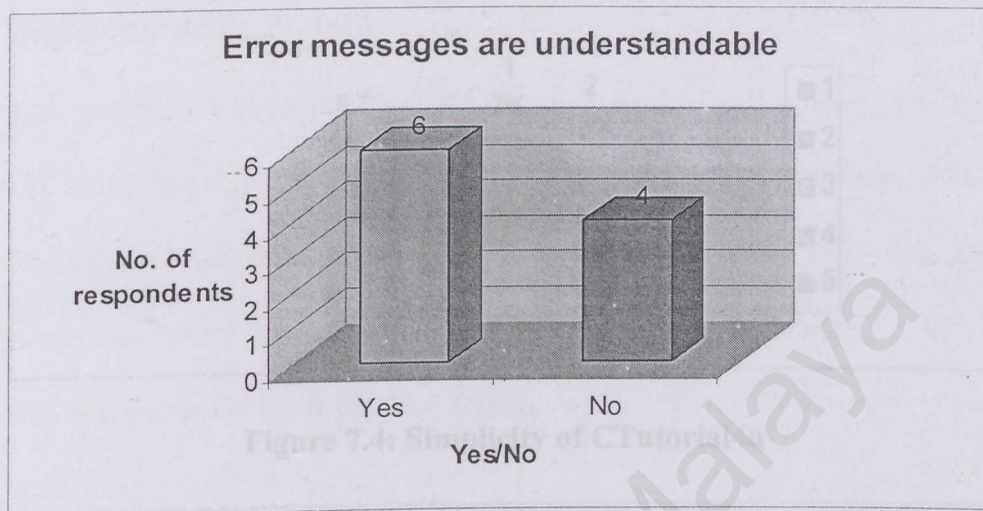


Figure 7.3: CTutorial4u's error messages

7.2.4 Validation on Input Data

The tutorial session is a good example to clearly indicate how the system has the validation on input data feature. When the user selects integer in Arithmetic topic, only numbers are allowed and other keys on the keyboard will not be accepted.

7.2.5 Interactivity

One of the added values to this system is the interactivity where user data are captured and stored in memory and the code is amended accordingly based on the user's input data. This feature is in the tutorial session.

7.2.6 Simplicity & Consistency

The system is simple to use and the clear instruction made the users to be easy start to use and exit from the system. The system is simple to adopt because screen design is

consistent throughout each session in the system. Figure 7.4 shows the rating among the users regarding simplicity of the system where majority which is 33% rated 1 for the scale (1=Simple, 5 = Very complicated).

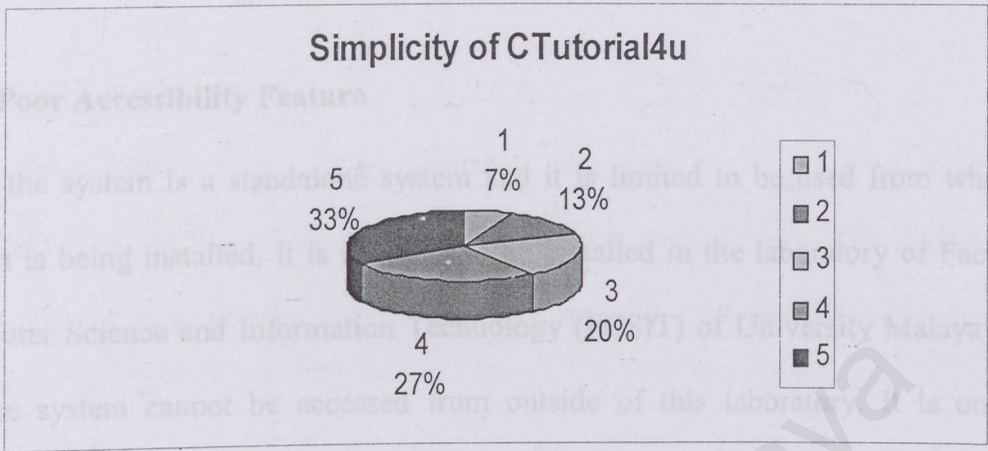


Figure 7.4: Simplicity of CTutorial4u

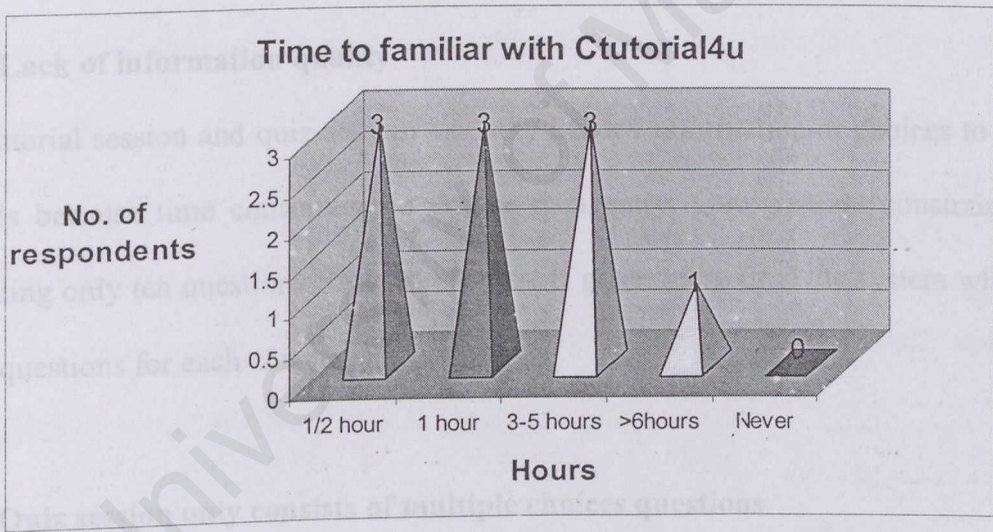


Figure 7.5: Time taken to familiarize with the system

The Figure 7.5 depicts that there are 3 users take ½ hours time, 3 users 1 hour and 3 users takes 3 to 5 hours time to familiarize with the system. When the system is consistent in terms of screen design and has all the good design features described by Jacob Nielsen then it is easy to understand and used by novice users.

7.3 System Limitations

On contrary to the system strengths, the system has a few limitations which need further works to enhance the system to be zero-defect.

7.3.1 Poor Accessibility Feature

Since the system is a standalone system and it is limited to be used from where the system is being installed. It is intended to be installed in the laboratory of Faculty of Computer Science and Information Technology (FCSIT) of University Malaya (UM). So, the system cannot be accessed from outside of this laboratory. It is one of a constraint but the requirement was in a way.

7.3.2 Lack of information quality

The tutorial session and quiz session has very limited information or choices to select. This is because time consuming to do the data entry. Quiz session constrained by providing only ten questions for each chapter. If given more time the system will have more questions for each chapter.

7.3.3 Quiz session only consists of multiple choices questions

There is only multiple choice questions available in the quiz session. The system did not cater subjective question where the user writes the code and the system verify the code is acceptable or not. This function needs more time where every string and character is searched one by one and if the characters matches with the code provided then the system should indicate it is acceptable.

7.4 Future Enhancements

Despite of the system limitation discussed earlier, the system requires further enhancements.

7.4.1 Web- based system

In the future, the system will be published developed as a web based system and published to World Wide Web using VB.Net to be accessed by everyone. This will be a tool to learn C programming and as online quiz to test user’s capability of programming using C.

7.4.2 Improve Information Quality

Data entry of all the possible questions to be asked should be entered to the database of quiz session.

7.4.3 Quiz session to be more interactive

Allow the users to key in their program codes and check whether their codes are acceptable by comparing them with the records in the database.

7.5 Comparison between before and after using CTutorial4u

The Table 7.2 actually describes how CTutorial4u helps the students in the aspects of time, exercise, interactivity, simplicity and passing rates. From that table, it is clear that CTutorial4u actually contributed a lot to the students in improving their knowledge in C.

Table 7.2: Comparison between before and after using Ctutorial4u

Aspects	Before	After
Time	<ul style="list-style-type: none"> • Learn through books, time consuming to search for quality books. • Not timely feedback from the book or manual tutorial. 	<ul style="list-style-type: none"> • Saves time where Ctutorial4u act as a reference & installed in laboratories and everyone can use it. • Timely feedback for tutorial and quiz session.
Exercise	<ul style="list-style-type: none"> • Students go through quiz or exercise from the book; some exercises no answers. 	<ul style="list-style-type: none"> • Students enjoy going through quiz or tutorial session with immediate feedback.
Interactivity	<ul style="list-style-type: none"> • Books and classroom lectures doesn't cater individual needs. • Difficult for students to know whether the code they write in paper is correct. 	<ul style="list-style-type: none"> • CTutorial4u provide tutorial session as a tool where it gathers user input to manipulate codes stored in database. • Validate user input where if variable type is integer then only whole numbers are allowed to be selected for variable value in tutorial session for the topic of arithmetic.
Simplicity	<ul style="list-style-type: none"> • Students find topics such as array and pointers difficult. 	<ul style="list-style-type: none"> • Simplified 6 major topics; arithmetic, control structures, functions, array, pointer and file processing in tutorial session using rule-based architecture.
Passing rates	<ul style="list-style-type: none"> • Low passing rates in C programming in local universities. 	<ul style="list-style-type: none"> • Tool to increase passing rates in C programming. (Not proven but an added exercise/reference for poor students).

Chapter 8 Conclusion

Finally, the system has been completed on time within the timeframe given. The system was developed step-by step from identifying the scope, literature review, requirement collection and analysis, system design, implementation, testing and evaluation of the system. Throughout the development process there were no major obstacles faced since everyone involved in the system has given full-cooperation to make this project as a success.

The development of CTutorial4u was incremental, so the system was tested during the coding phase. Unit testing, sub-system testing and integration testing was done and problems were rectified immediately earlier before the modules were combined to form the real system. User acceptance test were done once the system is fully integrated to make sure the system is ready to be delivered.

The system has its own strengths and weakness. The limitation listed actually to be further enhanced by other students. As such, the system can be used as a reference or groundwork for future research by adding on more functionality.

As a nutshell, the system comply all the initial scope and requirements and provides other value-added features which reveal it as a more reliable, user friendly, interactive and efficient.

References

- [1] Bahrami. A (1999) *Object Oriented System Development Using The Unified Modeling Language*. McGraw-Hill International Edition. Singapore.
- [2] Choi. Jinmu., *A Rule-Based Expert System Using an Interactive Question-and-Answer Sequence*. Department of Geography, University of Georgia. Available at <http://www.cobblestoneconcepts.com/ucgis2summer2002/choi/choi.htm>. Accessed on September 5, 2004.
- [3] Deitel H.M. and Deitel. P.J C: *How to Program*. 2nd Edition. Prentice Hall International Editions, New Jersey. p 148 -180.
- [4] Durkin. J. (1994) *Expert Systems Design & Development*. Macmillan. New York, NY.
- [5] Hughes.B and Cotterell.M (1999). *Software Project Management (2nd Edition)*. The McGraw-Hill International (UK).p 65-66.
- [6] Jackson. P. (1999). *Introduction To Expert Systems*. 3rd Edition. Addison –Wesley. London, England. p 89-100.
- [7] Kay G.Schulze, Robert N. Shelby, Donald J. Treacy, Mary C. Wintersgill, Kurt Vanlehn and Abigail Gertner. *ANDES: An Intelligent Tutor for Classical Physics*. The Journal of Electronic Publishing, September 2000 [journal online]; Available at <http://www.press.umich.edu/jep/06-01/schulze.html>. Internet; Accessed on December 18, 2004.
- [8] Nielsen, J., *Designing User Interfaces for International Use: Ten Usability Heuristics*. Elsevier Science Publishers, 1990 [journal online]: Available at http://www.useit.com/papers/heuristic/heuristic_list.html. Internet: Accessed on March 12, 2004.
- [9] Ueno, H., *A Generalized Knowledge-Based Approach to Comprehend Pascal and C Programs*, in *Knowledge-Based Software Engineering* (Eds, Navrat, P. and Ueno, H, IOS Press. pp132-139. 1998)
- [10] Ueno,H., Inoue, T., "A Shared Intelligent Programming Environment on the Internet for Learning C Programming. Proc ICCE, 1999. Available at

<<http://research.nii.ac.jp/~ueno/ronbun/icce99final.pdf>>. Internet; Accessed on December 18, 2004.

[11] Ueno,H., *Knowledge Based Intelligent Programming Environment - From the Point of View of Program Comprehension*, Information Processing, Vol.2, No.10, pp.1280-1296 (1987).

Carnegie Learning Integrated Math Available at <http://www.carnegielearning.com/products/integrated_math/> Accessed on July 24, 2004.

C Programming Tutorials. Available at <<http://www.computer-training-software.com/justc.html>>. Accessed on 6th July 2004.

Cprogramming.com-Your Resource for C and C++. Available at <<http://www.cprogramming.com/tutorial/quiz/quiz1.html>>. Accessed on December 21, 2003.

Holmes, Steve., *C Programming*. University of Strathclyde Computer Centre, January 1995. Available at <<http://www.strath.ac.uk/IT/Doc/Course/>>. Internet; Accessed on September 3, 2005.

Nielsen, J., and Molich, R. (1990). Heuristic evaluation of user interfaces, *Proc. ACM CHI'90 Conf.* (Seattle, WA, 1-5 Apr. 90), 249-256.

Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. *Proc. ACM CHI'94 Conf.* (Boston, MA, April 24-28), 152-158.

Nielsen Online version 3 at <<http://www.nielsenonline.com/sourcecode.html>> Accessed on November 20, 2005.

Softsys GmbH. Available at <<http://www.softsys.com/prerequisite.html>> Accessed on November 30, 2003.

Tauk, Len. C Tutorial. September 11, 1997. Available at <http://kdel.cac.calpoly.edu/~hauck/THESIS/C_Tutorial.html> Accessed on September 4, 2003.

Bibliography

A1 VBCode - The VB Source Code Site. Available at <http://www.a1vbcode.com/default.asp>> Accessed on May 6, 2004.

Carnegie Learning - Integrated Math. Available at http://www.carnegielearning.com/products/integrated_math/> Accessed on July 24, 2004.

C Programming Tutorials. Available at <http://www.computer-training-software.com/justc.htm>>. Accessed on 6th July 2004.

Cprogramming.com-Your Resource for C and C++. Available at <http://www.cprogramming.com/tutorial/quiz/quiz1.html>>. Accessed on December 23, 2003.

Holems, Steve., *C Programming*. University of Strathclyde Computer Centre, January 1995. Available at <http://www.strath.ac.uk/IT/Docs/Ccourse/>>. Internet; Accessed on September 3, 2003.

Nielsen, J., and Molich, R. (1990). Heuristic evaluation of user interfaces, *Proc. ACM CHI'90 Conf.* (Seattle, WA, 1-5 April), 249-256.

Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. *Proc. ACM CHI'94 Conf.* (Boston, MA, April 24-28), 152-158.

Niloc Online version 3 at <http://www.niloconline.com/csourcecode.html>> Accessed on November 30, 2003.

Softsyst GmbH. Available at <http://www.softsyst.com/prequizef.html>> Accessed on November 30, 2003.

Tauk, Lee. C Tutorial. September 11, 1997. Available at http://kdat.csc.calpoly.edu/~ltauck/THESIS/C_Tutorial.html> Accessed on September 4, 2003.

Appendix A: Questionnaire Form

SURVEY ON RULE-BASED C TUTORIAL SYSTEM

Purpose: To investigate the way C programming subject being taught in higher learning institutes of Malaysia and evaluate how to improve the passing rates among the students using knowledge based C tutorial system designed in rule-based system architecture.

A. PERSONAL INFORMATION

1. University or College Name: _____
2. Semester & year: _____
3. Field or program of study in the university or college:

<input type="checkbox"/>	Computer Science
<input type="checkbox"/>	Information Technology
<input type="checkbox"/>	Computer Networks and Security
<input type="checkbox"/>	Others, please specify: _____

B. GENERAL QUESTIONS ON C PROGRAMMING COURSE

1. In what year this subject being offered in your university or college for your program of study?

2. Are there any prerequisites to take this subject? If yes please specify the prerequisites course:

<input type="checkbox"/>	Yes. Course: _____	<input type="checkbox"/>	No
--------------------------	--------------------	--------------------------	----
3. What do you think of this course and its level of complexity?

<input type="checkbox"/>	Very easy	<input type="checkbox"/>	Easy	<input type="checkbox"/>	Moderate	<input type="checkbox"/>	Difficult	<input type="checkbox"/>	Very Difficult
--------------------------	-----------	--------------------------	------	--------------------------	----------	--------------------------	-----------	--------------------------	----------------
4. Assignments, lecture materials and projects prepared for this course was good.

<input type="checkbox"/>	Strongly agree
<input type="checkbox"/>	Agree
<input type="checkbox"/>	Disagree
<input type="checkbox"/>	Others, please specify: _____
5. What are the methods being used to teach C Programming in your university or college?

<input type="checkbox"/>	Slides or PowerPoint presentation
<input type="checkbox"/>	Regular tests and quizzes
<input type="checkbox"/>	Lab sessions to practice lessons taught in class
<input type="checkbox"/>	Group project and assignment
<input type="checkbox"/>	Others, please specify: _____
6. Does your course lecturer have the following characteristics? Please specify Y=Yes and N=No for the listed characteristics below.

<input type="checkbox"/>	Able to explain course material clearly
<input type="checkbox"/>	Able to stimulate student participation and understanding
<input type="checkbox"/>	Available for help and consultation
<input type="checkbox"/>	Communication skills (clear speech)
<input type="checkbox"/>	Good Programming Skills
7. What is the average mark that you normally get for this course (quiz or test)?

<input type="checkbox"/>	<50%
<input type="checkbox"/>	50% to 70%
<input type="checkbox"/>	70% to 90%
<input type="checkbox"/>	90% to 100%
8. Do you have any problems with this course? If yes, what are the problems?

<input type="checkbox"/>	No proper guidance from the course lecturer
<input type="checkbox"/>	Very few references available in the library
<input type="checkbox"/>	Do not have personal interest in this subject

- ☐ Lecture method does not improve my understanding and programming skills
☐ Others, please specify: _____

9. Suggestions or recommendations to improve the course:

- ☐ Course topics should be augmented by practical exercises and assignments
☐ Tutors or lecturers to conduct lab sessions for students to practice the lectures
☐ New rule-based C tutorial system installed in every computer labs as a tutoring tool
☐ Others, please specify: _____

C. QUESTIONS ON PROPOSED RULE-BASED C TUTORIAL SYSTEM

Note: Rule-based system is actually a system run based on some standard rules set in the working memory. The proposed C Tutorial system will be designed in this architecture. The rules for this tutorial system will remain the same for different runs to encourage modifiability, extendibility and scalability.

1. What are the features you expect from this system?

- ☐ Tutorials to cover all topics from a C Programming book
☐ Short and long quizzes with answers guidance attached for wrong answers
☐ Consultation session
☐ Sample programs or source code
☐ Search function for important keywords
☐ Others, please specify: _____

2. How do you want the tutorials to be designed? Please choose only one option below.

- ☐ By only selected important or complicated topics
☐ By application eg: Simulation, Mathematics and so on
☐ By all chapters following a reference book
☐ Others, please specify: _____

3. How long do you need to finish a short quiz, which consists of only 10 simple questions?

- ☐ 8 minutes
☐ 10 minutes
☐ 12 minutes
☐ 15 minutes
☐ Others, please specify: _____

4. How would you prefer the quizzes to be organized?

- ☐ After every chapter
☐ Both ways
☐ Mixed questions from all chapters
☐ Others: _____

5. Rank the C programming topics below based on their complexities as what you feel.

1 = Most difficult, 2 = Average, 3 = Moderate, 4 = Easy and 5 = Easiest

- ☐ Arrays and Sorts
☐ String functions and loops
☐ Counters
☐ Pointers
☐ Controls Structures

6. If given a choice, which form of the rule-based C Tutorial system do you prefer? Give your reasons.

- ☐ Standalone system installed in the lab
☐ Online system

Reasons:

~END~

Thank you for your Co-operation

Appendix B: Evaluation Form

EVALUATION FORM

CTutorial4u

Name : _____ Course: _____

Year : _____ Institute/University: _____

1. What do you think about CTutorial4u? 1= Simple, 5= Very complicated

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

2. How much of your C programming knowledge has improved using CTutorial4u?

(1-5) 1= Little, 5= Very much

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☒ 5

3. Did you find any difficulties in moving from one screen to another?

(1-5) 1= Not consistent, 5 = Consistent

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

4. Please rank the system in terms of user friendliness. (1-5) 1= Not at all, 5 =Very

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

5. Are there error messages prompting to your understanding?

☐ Yes ☐ No

6. How long did you take to familiarize with the system?

☐ 1/2 hour ☐ 1 hour ☐ 3-5 hours ☐ >6 hours ☐ Never

7. How do you rate the system in terms interactivity and efficiency?

(1-5) 1= Not at all, 5 =Very

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

8. Does exit from the system is easy?

☐ Yes ☐ No

9. Other value added comments to improve the system.

~End~

Thank You for your Co-operation

PART 1: About CTutorial4u

a) Introduction

CTutorial4u is developed based on knowledge design in rule based system architecture. It is windows based system to teach the students the basic programming concept in C programming and to ease the tutors in teaching C programming. This system consists of four main functions as depicted in the Figure C.1. The four functions are divided into User Module and Admin Module.

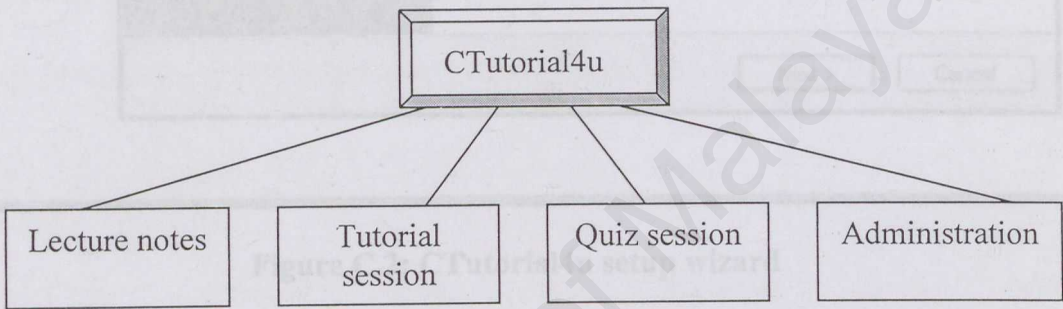


Figure C.1: Main functions of CTutorial4u

b) Hardware and Software Requirements

The minimum hardware and software tools required to install the CTutorial4u are Windows 98 as the operating system with Pentium II 100MHz speed. It is better to have at least 2 GB free space on the C:\ drive.

c) CTutorial4u Installation and Un-installation

Install Ctutorial4u

1. Insert the CTutorial4u CD to the CD-ROM.
2. Once the CD-ROM reads the CD, double click on the Setup file to install CTutorial4u. Then the setup wizard as in the Figure C.2 will guide further to the system installation.

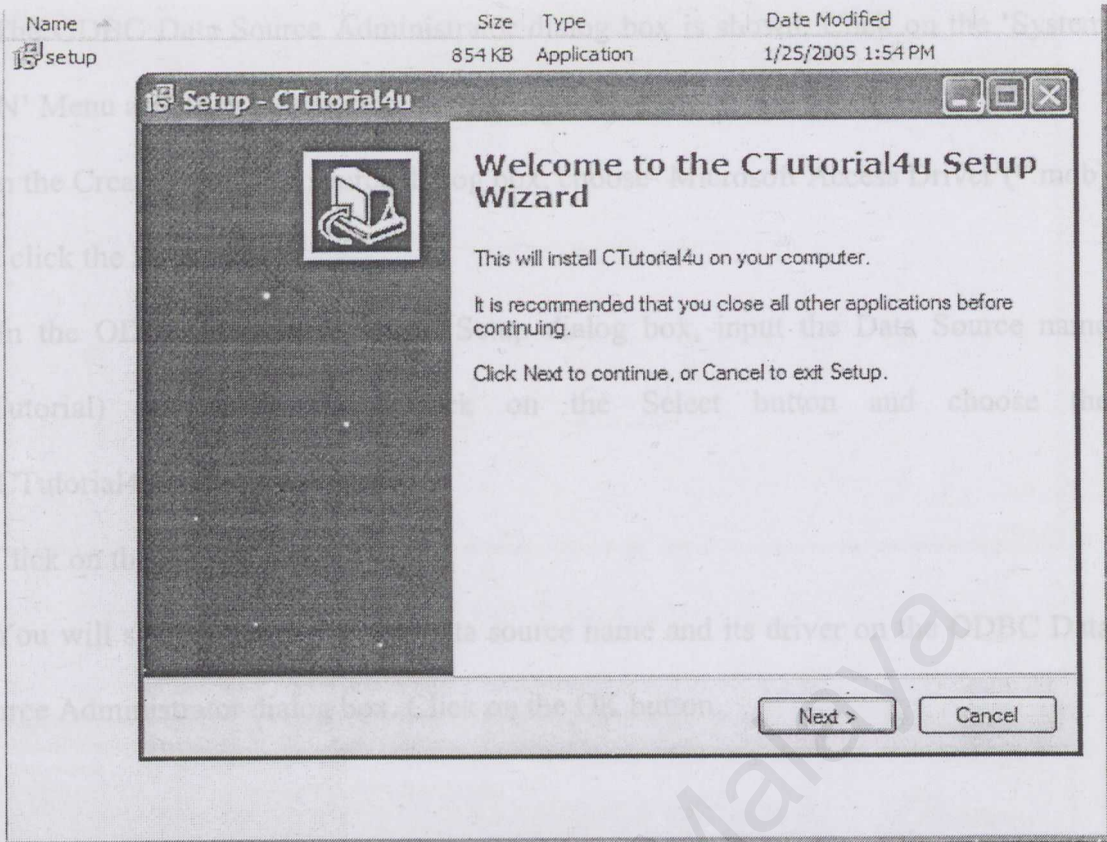


Figure C.2: CTutorial4u setup wizard

3. Click Next button and now is the selection for additional tasks which are optional to choose. There are options:

- Create a desktop icon - there will be an icon created on the desktop
- Create a quick launch icon – there will be quick launch icon on the desktop.

4. Click Next, and then choose Install and the system is ready to be install at location C:\CTutorial4u (defaulted).

5. Then choose to Launch CTutorial4u and click Finish to complete the installation and the system is ready to be used.

Configure System Data source

1. Click 'Start' from the Windows Start Menu, point to setting and click 'Control Panel'.

2. Double-click the 'Administrative Tools', and double-click 'Data Sources (ODBC)'.

3. The ODBC Data Source Administrator dialog box is shown. Click on the 'System DSN' Menu and click Add button.
4. In the Create New Data source dialog box, choose 'Microsoft Access Driver' (*.mdb) and click the Finish button.
4. In the ODBC Microsoft Access Setup dialog box, input the Data Source name (CTutorial) and description. Click on the Select button and choose the C:\CTutorial4u\C1.mdb database.
5. Click on the OK button
6. You will see the newly created data source name and its driver on the ODBC Data Source Administrator dialog box. Click on the OK button.

Uninstall CTutorial4u

1. Click uninstall from the Windows Start Menu.
Start>Programs>CTutorial4u>Uninstall.

PART 2: How to use

a. User Module

User can start using the system from start menu by pressing the CTutorial4u or from the desktop icon if it is selected during installation earlier. The screen below shows the Logon menu which is the first screen after the user activates CTutorial4u.exe. This form is to authenticate only authorized user access the system.

1. Key in correct username and password.
2. Press [OK] button to enter the main menu of the system. Press [Cancel] button to cancel the operation and exit from the system.

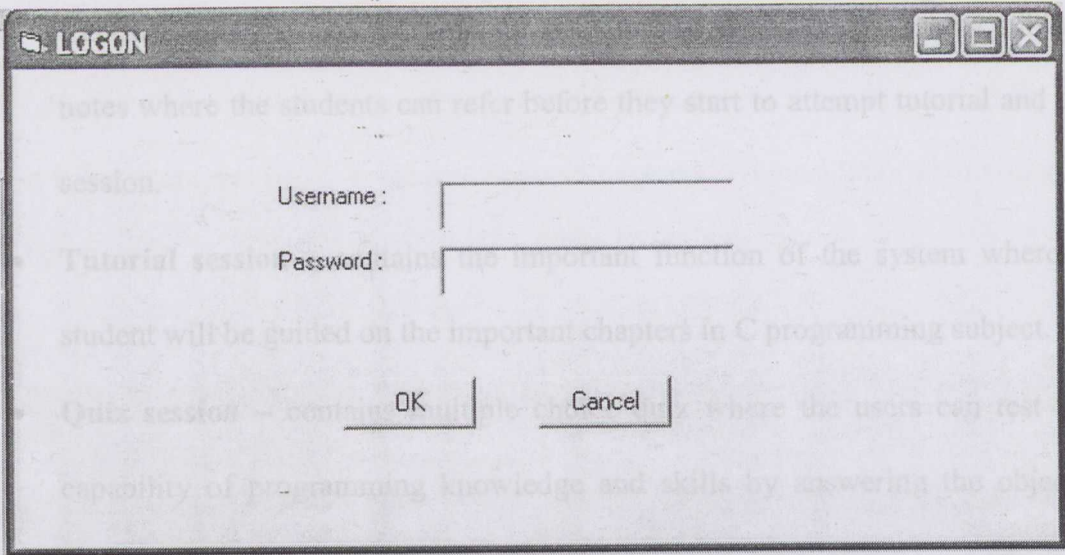


Figure C.3: Logon page

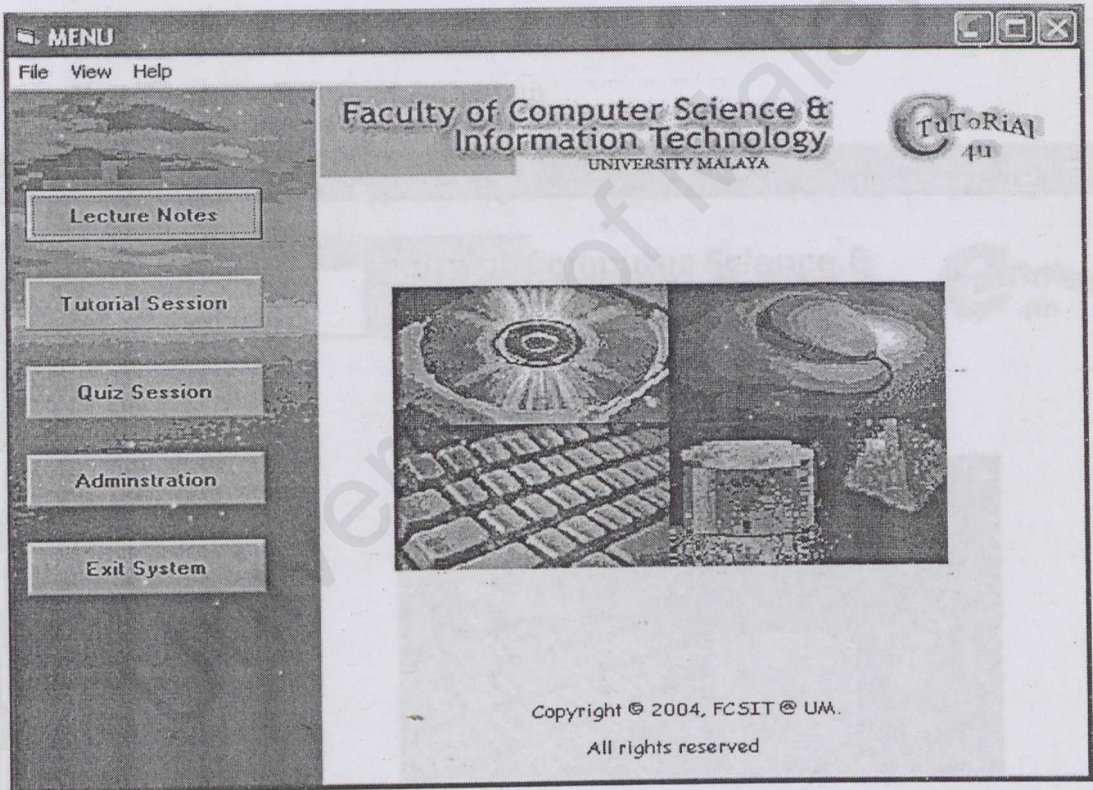


Figure C.4: Menu page

The main menu above will be displayed if the user correctly key in his or her username and password in the logon page.

Users have five command buttons to choose in the menu screen which are:

- **Lecture notes** – contains power point show of the C programming subject's notes where the students can refer before they start to attempt tutorial and quiz session.
- **Tutorial session** – contains the important function of the system where the student will be guided on the important chapters in C programming subject.
- **Quiz session** – contains multiple choice quiz where the users can test their capability of programming knowledge and skills by answering the objective quiz.
- **Administration** – enables the administrator to add, edit and delete users and the quiz questions.
- **Exit System** – Exit from the system

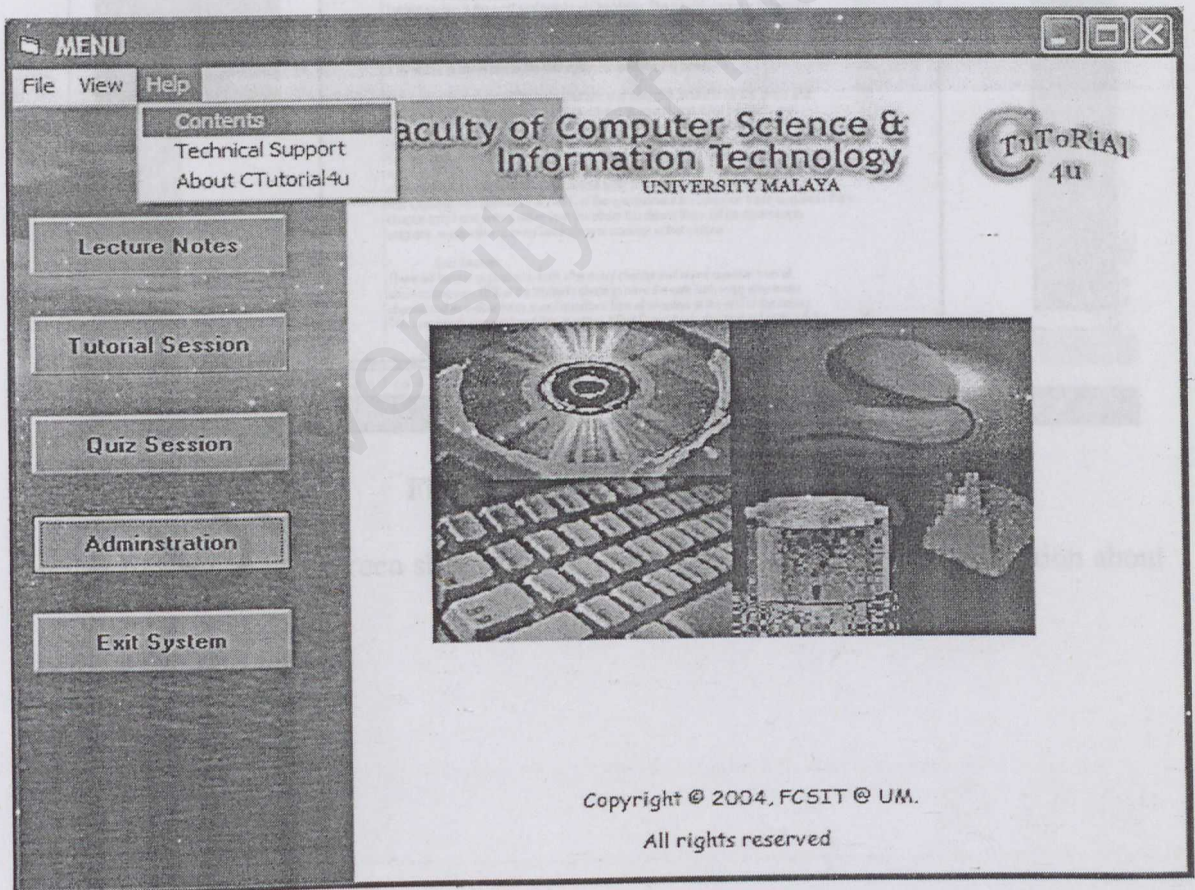


Figure C.5: Menu page highlighting the menus

The main menu also contain the menu bar where user can choose File menu to open a page, close the page and exit from the system. The Help menu provide guide to the user where it has Contents page, Technical Support page and About CTutorial4u (summary of CTutorial4u).

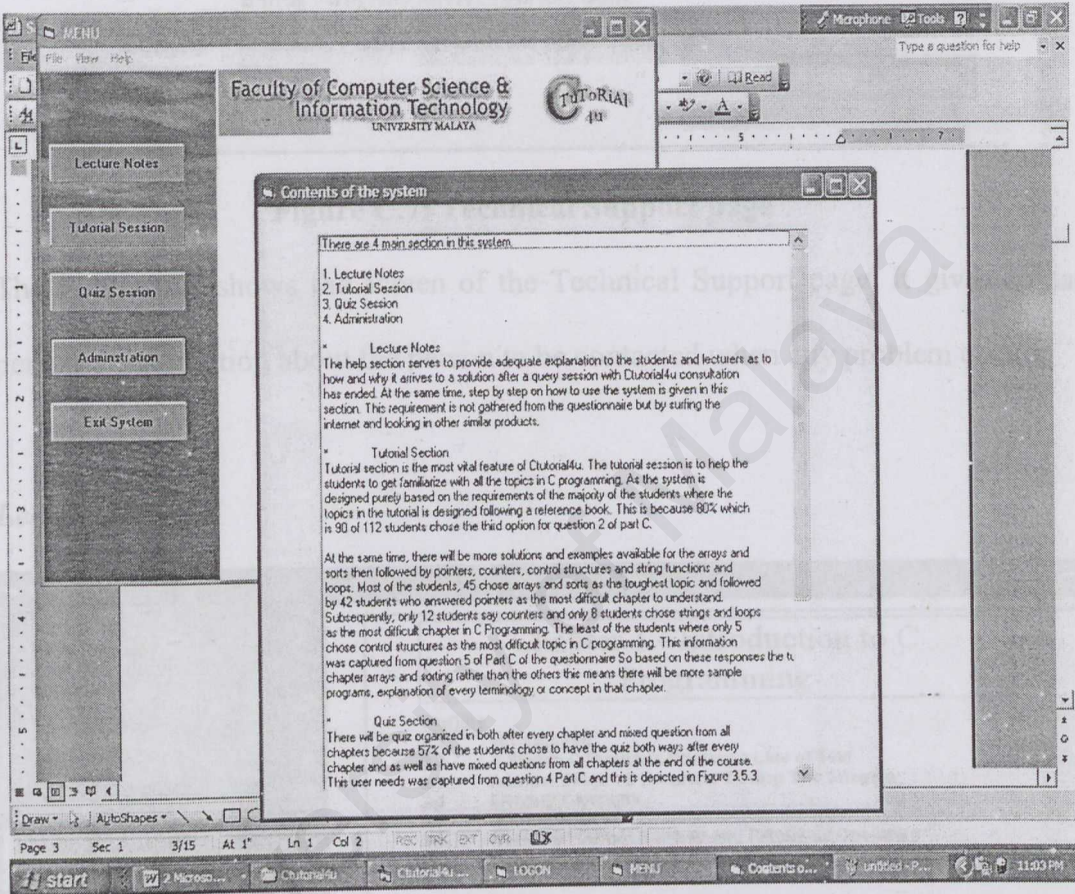


Figure C.6: Contents page

The figure C.6 is a screen shot of the contents page. It gives brief description about the system.



Figure C.7: Technical Support page

The Figure C.7 shows the screen of the Technical Support page. It gives contact person's information about the person to be contacted when any problem occurs.

Lecture Notes:

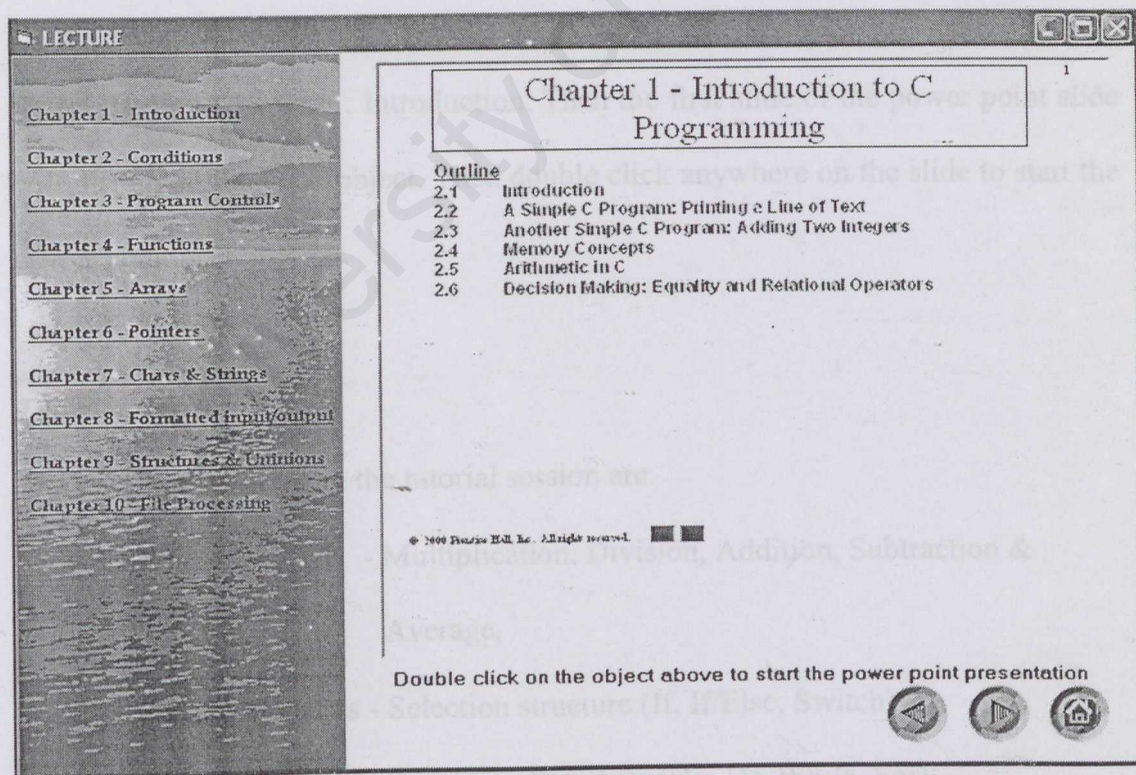


Figure C.8: Lecture notes page

As what explained earlier, the lecture notes page contains lecture notes in the format of power point slides. It contains the links to ten basic topics of C programming. The chapters available for reference are:

- Chapter 1: Introduction
- Chapter 2: Condition
- Chapter 3: Program Controls
- Chapter 4: Functions
- Chapter 5: Arrays
- Chapter 6: Pointers
- Chapter 7: Characters and strings
- Chapter 8: Formatted input/output
- Chapter 9: Structures and Unions
- Chapter 10: File processing

Select the link Chapter 1: Introduction. Then the first slide of the power point slide will display in the OLE object. Then double click anywhere on the slide to start the power point presentation.

Tutorial Session

The chapters available in the tutorial session are:

- Arithmetic - Multiplication, Division, Addition, Subtraction & Average.
- Control Structures - Selection structure (If, If/Else, Switch)
- Repetition Structure (Do, Do-While, For)
- Functions - Maximum & Minimum
- Array - Size of array from 1 to 10 & with or without histogram.

- Pointers - Call by reference & call by value
- File Processing - Create a Sequential file
- Writing to a Random Access file
- Reading a Random Access file

Table C.1 to Table C.6 is the listing of required field, selection and description of each and every topic in tutorial session.

- Arithmetic

Table C.1: Arithmetic Tab

Required input	Input type	Selection value	Description
Arithmetic Type	Combo box	Multiplication, Division, Addition, Subtraction & Average.	
Variable Type	Radio button	Integer, Floating point, Double	
Variable Value A: Variable Value B:	Text boxes	User's input.	If variable type is integer then only numbers are allowed but for floating point and double types allows ".".If correct values entered then only code will be displayed

Based on the arithmetic type selected, variable type and input for variable A and B, the source will be displayed in the list box as depicted in the Figure C.9. Then, when the code is executed the output is as in the Figure C.10. If the arithmetic type is multiplication, and variable type is integer, value variable A is 5 and value variable B is 7 then the output is 35.

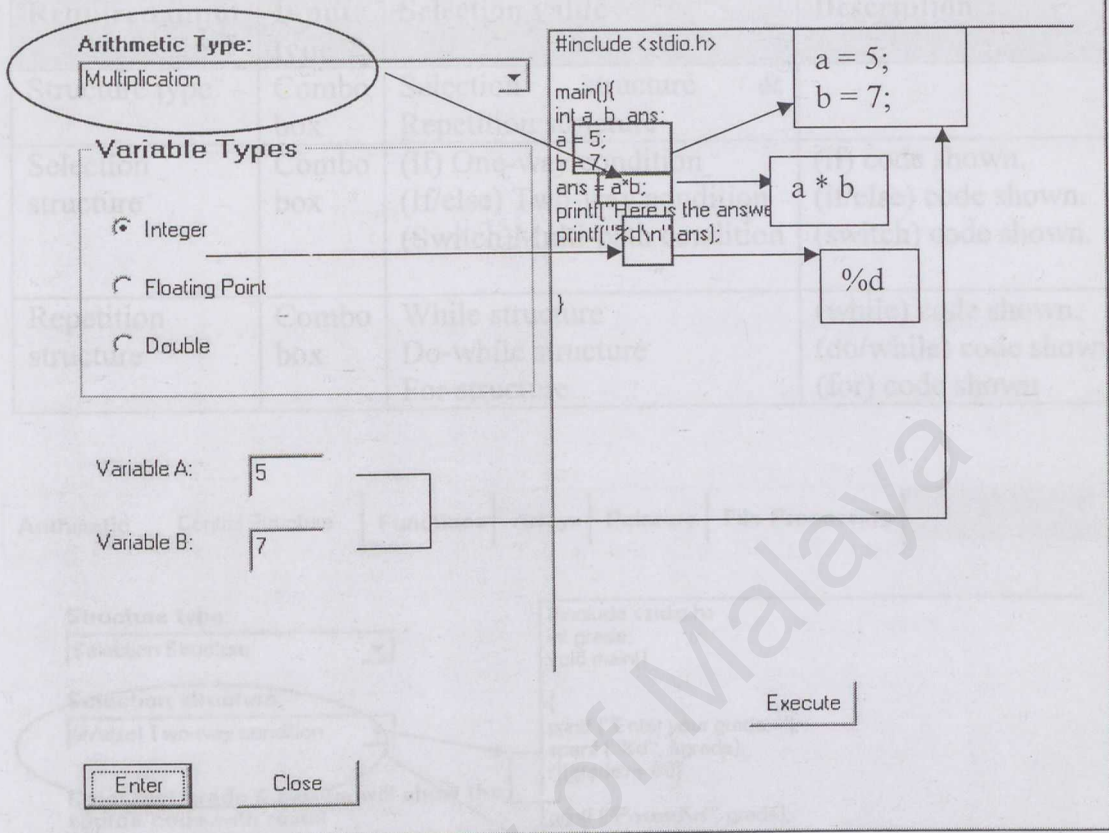


Figure C.9: Arithmetic tutorial page

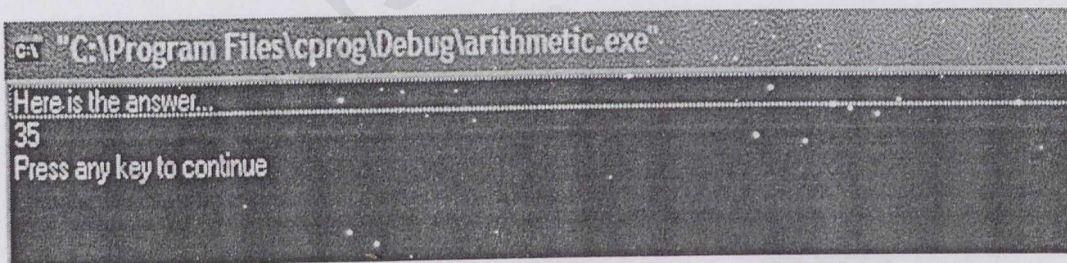


Figure C.10: Arithmetic.exe

Based on the arithmetic type selected, variable type and input for variable A and B, the source will be displayed in the list box as depicted in the Figure C.9. Then, when the code is executed the output is as in the Figure C.10. If the arithmetic type is multiplication, and variable type is integer, value variable A is 5 and value variable B is 7 then the output is 35.

- Control Structure

Table C.2: Control Structure Tab

Required input	Input type	Selection value	Description
Structure type	Combo box	Selection structure & Repetition structure	
Selection structure	Combo box	(If) One-way condition (If/else) Two-way condition (Switch) Multi-path condition	(if) code shown. (if/else) code shown. (switch) code shown.
Repetition structure	Combo box	While structure Do-while structure For structure	(while) code shown. (do/while) code shown. (for) code shown

Arithmetic
Control Structure
Functions
Arrays
Pointers
File Processing

Structure type:

Selection Structure

Selection structure:

(if/else) Two-way condition

Enter test grade & system will show the source code with result

Grade:

70

```

#include <stdio.h>
int grade;
void main()
{
    printf ("Enter your grade: ");
    scanf ("%d", &grade);
    if (grade >= 60)
        printf ("Passed\n", grade);
    else
        printf ("Failed\n", grade);
}
endif;

```

Execute

Enter

Close

Figure C.11: Control Structure tutorial page



Figure C.12: CtrlStructure.exe

Structure type and the grades are vital in getting the source code and the output right in the control structure topic. Figure C.11 shows the controls structure tutorial page and the figure C.12 depicts the execution page for control structure with the relevant input is keyed in. Passing marks is 60 (unknown to user), so when the grade is entered 70 then the system will execute and message Passed is displayed as in Figure C.12.

- **Function**

Table C.3: Function Tab

Required input	Input type	Selection value	Description
Function type	Combo box	Maximum & Minimum	--
Value Integer 1: Value Integer 2: Value Integer 3:	Text boxes	User's input.	Key in the values for all 3 integers. Program code to select the maximum or minimum integer among all the 3 values will be displayed.

Function type.

Maximum

Enter integer values & system will show source code to detect the maximum or minimum value with result.

Values

Value Integer 1: 4

Value Integer 2: 6

Value Integer 3: 8

Execute

Enter Close

```

#include <stdio.h>

int Maximum(int , int, int)
{
    int main () {
        int a, b, c;
        printf("Enter integers:");
        scanf("%d%d%d", &a, &b, &c);
        printf("Maximum is: %d\n", Maximum( a, b, c ));
        int (int x, int y, int z){
            int ans = w;
            if ( x>ans )
                ans = x;
            if ( y>ans )
                ans = y;
            if ( z>ans )
                ans = z;
            return 0;
        }
        return ans;
    }
}

```

Figure C.13: Function tutorial page

```

C:\Program Files\lcp\Debug\function.exe
Enter integers: 4 6 8
Maximum is: 8
Press any key to continue.....

```

Figure C.14: Function.exe

Function tutorial requires the user to choose their preferred function either maximum or minimum and key in the values for integer 1, 2 and 3. Based on the information retrieved from the user input, then source code will be displayed. When the code is executed the output is as in the Figure C.14. In the Figure C.13, integer 1, 2 and 3 has respective values of 4, 6 and 8. So, in the Figure C.14, 8 is concluded as the maximum number.

Table C.4: Array Tab

Required input	Input type	Selection value	Description
Size of array	Combo box	1, 2, 3, 4, 5, 6, 7, 8, 9, 10	
Value Integer 1: Value Integer 2: Value Integer 3: Value Integer 4: Value Integer 5: Value Integer 6: Value Integer 7: Value Integer 8: Value Integer 9: Value Integer 10:	Text boxes	User's input.	Key in the values for the integers. Number of integer values to be keyed in, is based on the selection in size of array combo box. If 1 is selected then only 1 value need to be keyed in. All the text boxes displayed are mandatory fields.
Choose yes/no	Combo box	Yes /No.	This combo box needs user to choose whether user wants to view the histogram of the value keyed in the value integer text boxes. If yes, the program code to show the histogram will be displayed. If not, the program code to show the array values will be displayed.



Figure C.16: Array.exe

The array topic is not that difficult as we thought after going through the lectures and do the tutorial session. This topic requires the user to select the array size, key in the values for the variables and choose to display histogram or not. Based on the selection and input the system will display the source code and when being executed the output is as at Figure C.16

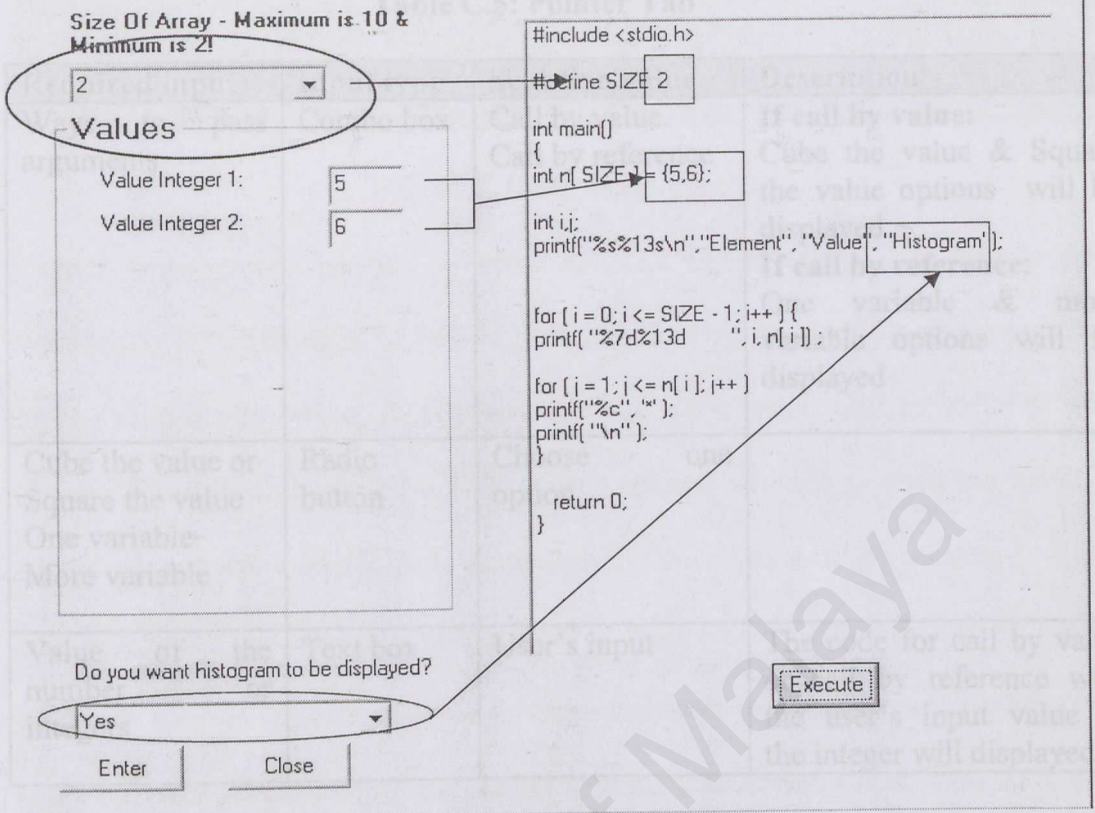


Figure C.15: Array tutorial page

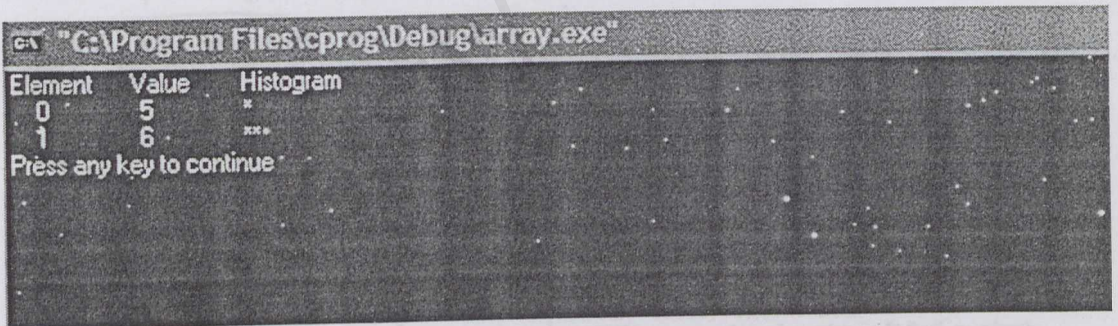


Figure C.16: Array.exe

The array topic is not that difficult as we thought after going through the lectures and do the tutorial session. This topic requires the user to select the array size, key in the values for the variables and choose to display histogram or not. Based on the selection and input the system will display the source code and when being executed the output is as at Figure C.16.

Table C.5: Pointer Tab

Required input	Input type	Selection value	Description
Ways to pass arguments	Combo box	Call by value Call by reference	If call by value: Cube the value & Square the value options will be displayed If call by reference: One variable & more variable options will be displayed
Cube the value or Square the value One variable More variable	Radio button	Choose one option.	
Value of the number or integers	Text box	User's input	The code for call by value or call by reference with the user's input value of the integer will displayed.

Arithmetic
Control Structure
Functions
Arrays
Pointers
File Processing

Ways to pass arguments:

Call by value

2 programs on call by value using pointer are available. Choose one of the option below and key in the variable value to be computed!

Options

☒ Cube the number

☐ Square the number

Value of the number

```

#include <stdio.h>

int cubebyvalue(int);

main(){
    int number = 4;

    printf ("The original value of number is %d\n", number);
    number = cubebyvalue(number);
    printf("The new value of number is %d\n", number);
    return 0;
}

int cubebyvalue(int n)
{
    return n*n*n;
}

```

Execute

Enter Close

Figure C.17: Pointer tutorial page

The original value of number is 4
 The new value of the number is 64
 Press any key to continue

Figure C.18: Pointer.exe

Pointer topic covers the call by value and call by reference as a way to pass arguments. The difference of the way to code the two types of passing arguments can be seen clearly when all the selection has given input. Call by value is very straight forward where based on the chosen options whether to cube the value or square the value the source code is displayed. When value of the number is 4 and chosen to cube the value, the output is as at Figure C.18. Whereas for the call by reference there is concept of array incorporated, if it more than one variable array made the code simple and shorter. Other than that, the concept is as same as call by the value but the source code will be different in terms of way of coding.

- File processing

Table C.6: File processing Tab

Required input	Input type	Selection value	Description
Select a process	Combo box	Create a Sequential file Writing to a Random Access file Reading a Random Access file	
Variable type	Combo box	int, double, char	
Variable value	Text box	User's input	
Name of file to be opened	Text box	User's input	Program code based on the selected criteria will be displayed

File processing types:

Create a Sequential File

Variable to be written into the file

Variable type:

int

Variable name:

A

File name to be opened?

create

Enter

Close

```

#include <stdio.h>
int main(){
    int A;
    FILE *cfPtr;
    if((cfPtr = fopen("create.txt","w"))==NULL )
        printf( "File could not be opened\n" );
    else{
        printf("Enter the A\n");
        printf("Enter EOF to end input.\n");
        printf("Enter your data here:");
        scanf("%d",&A);
        while(!feof(stdin)){
            fprintf(cfPtr,"%d\n",A);
            printf("? ");
            scanf("%d",&A);
        }

        fclose( cfPtr );
    }

    return 0;
}

```

Figure C.19: File processing tutorial page

The topic file processing actually covers how to write the code to create a sequential file and writing to a random access file based on the variable type, variable name and name of the file to be opened. The source code will be displayed based on the user input. As in the Figure C.19, if the file processing type chosen is create a sequential file, variable type is int, variable name is A and the file name is create, the source is as in the right panel. Other than that, when the option to read a random is selected then a simple source code on how to read a random access file, sequentially updates the data that already written to the file and create or delete new data to the file will be displayed for the users viewing.

Quiz Session:

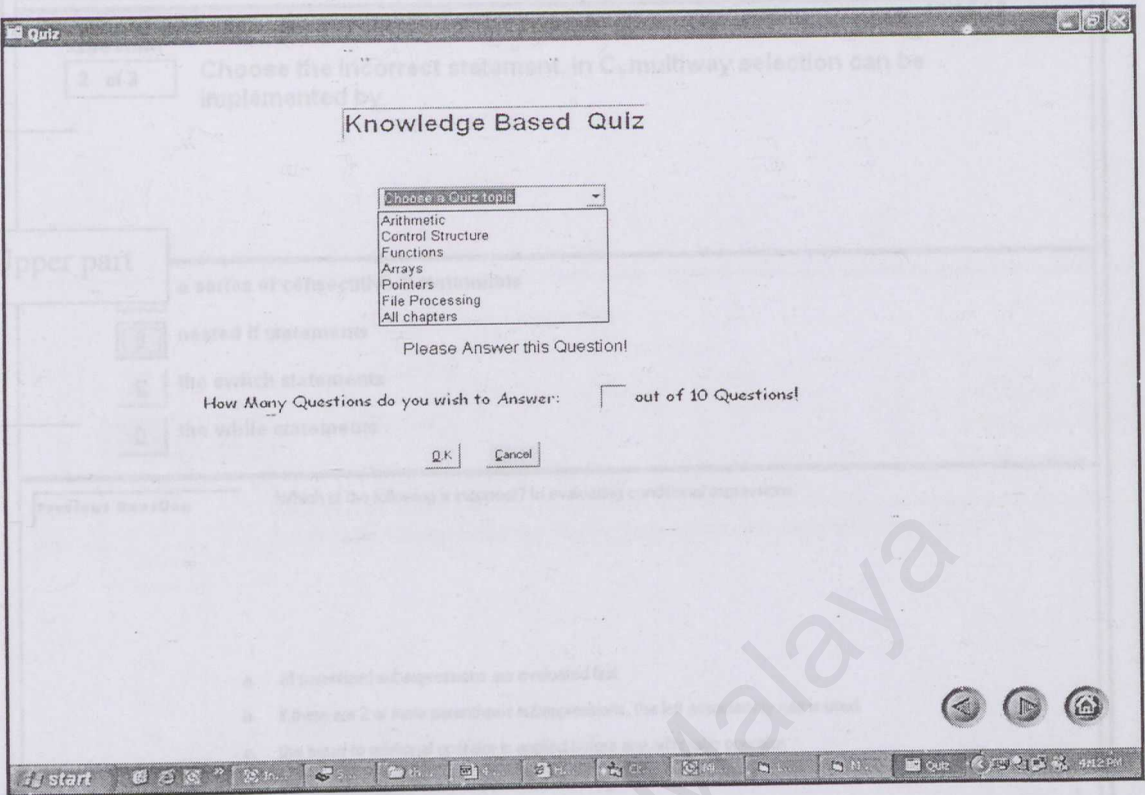


Figure C.20: Quiz session page

The Figure C.20 shows how the initial Quiz page looks like. This is the input form to generate quiz questions in the later page. Based on the user's input the system will generate the questions.

1. Select one of the topics in the combo box which displays 'Choose a Quiz topic'.
2. Key in the number of questions to answer in the text box.
3. Press [Ok] to start the quiz session.
4. Press [Cancel] to end the session and go back to menu page.

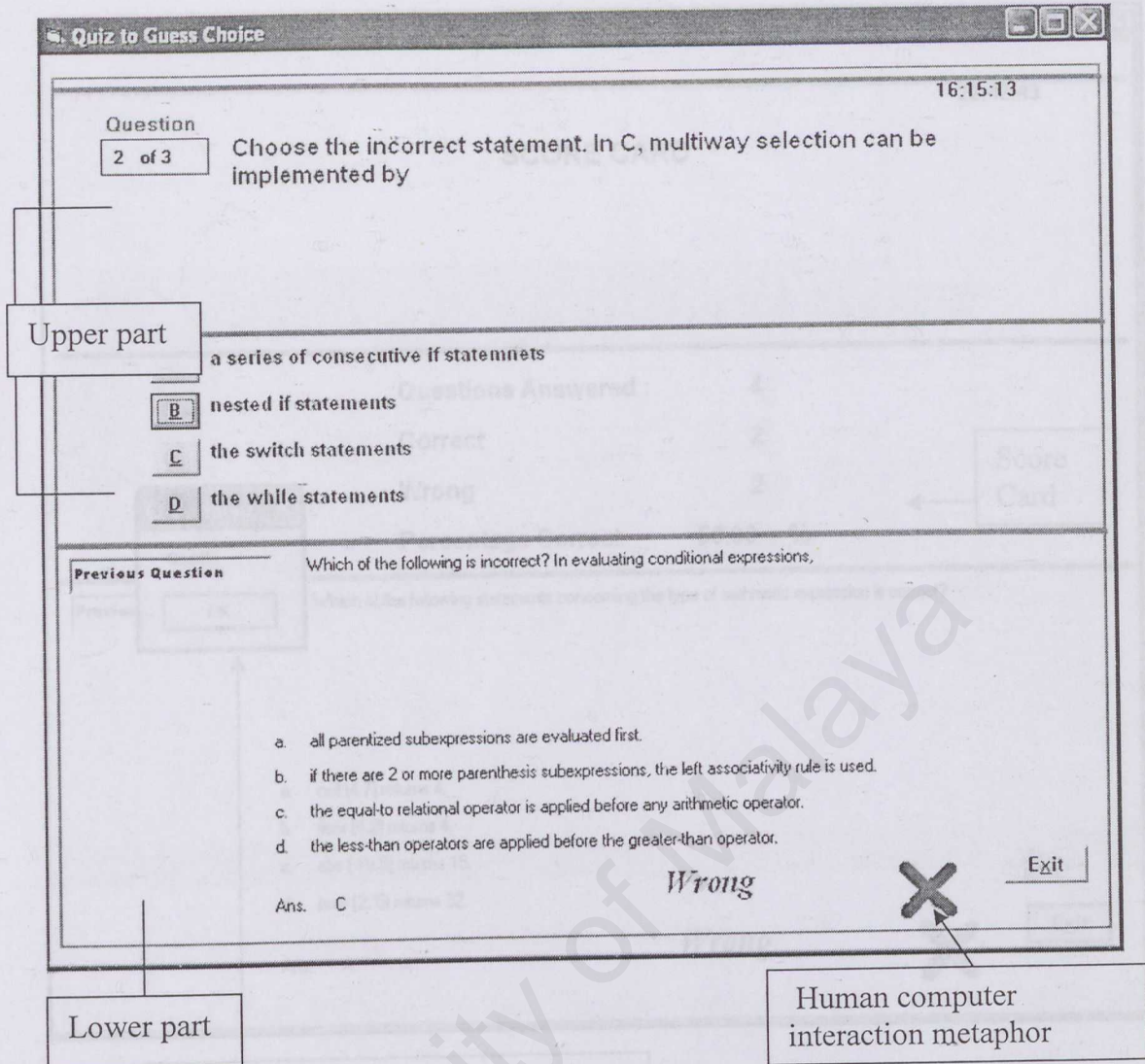
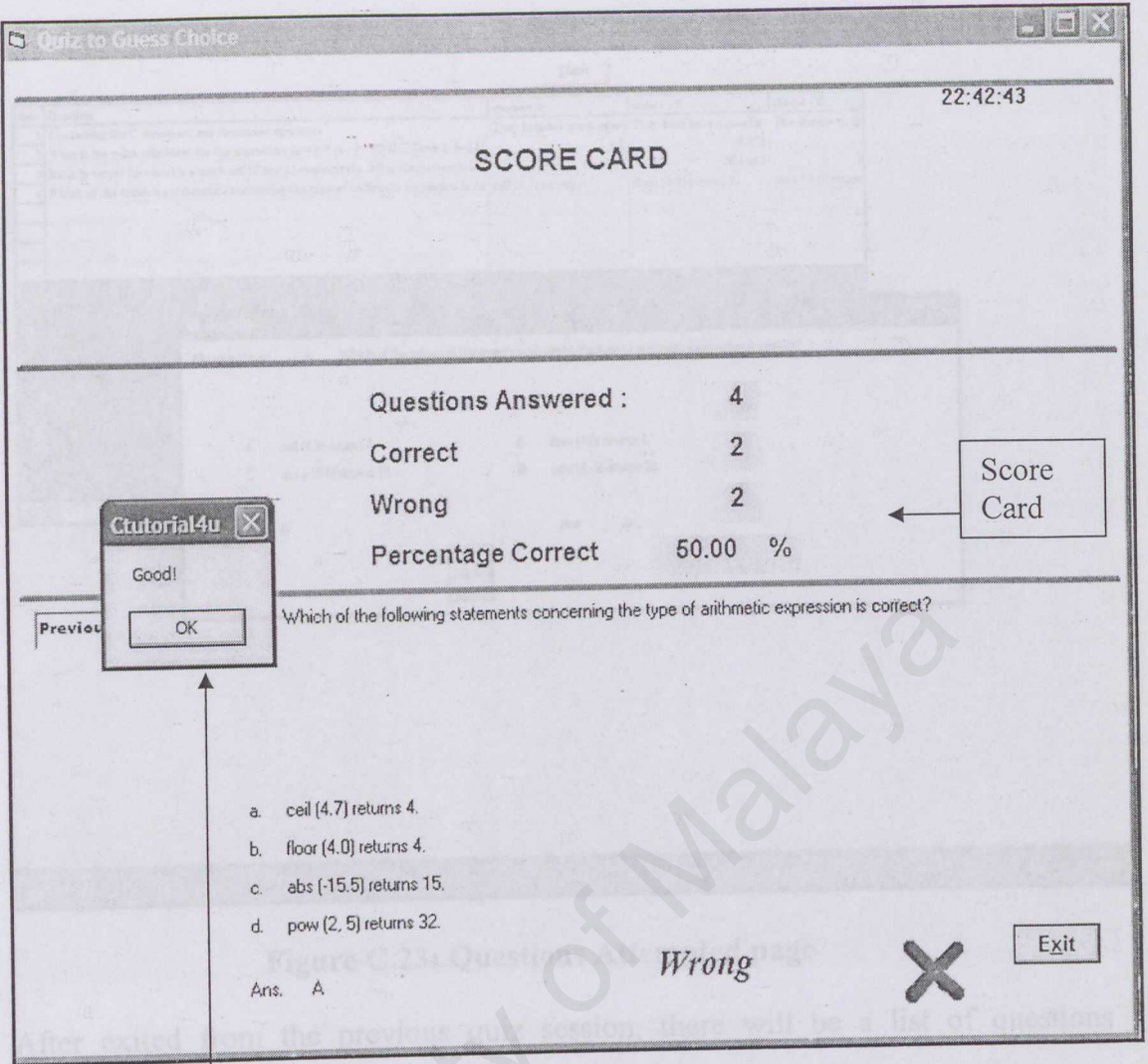


Figure C.21: Quiz to Choose answer page

The Figure C.21 depicts the quiz session which is divided into two parts. The upper part is the current question for the user to answer whereas the lower part is the previous question which already answered by the user. The correct answer is given for the previous question. At the same time, it is shown as wrong sign if the answer is wrong and correct sign if the answer is correct for purpose to give a clear notification whether their answer is correct or wrong.



Message box

Try again: if result <50

Good: if 50= \leq result <90

Excellent: if result \geq 90

Figure C.22: Score Card page

The Figure C.22 shows the score card which contains the total number of questions answered, number of correct answers, number of wrong answers and percentage or marks of the quiz. If the percentage scored is more than 50% then there will be a message prompted indicating [Good]. Click [ok] to proceed. Press [Exit] button to exit from the quiz session.

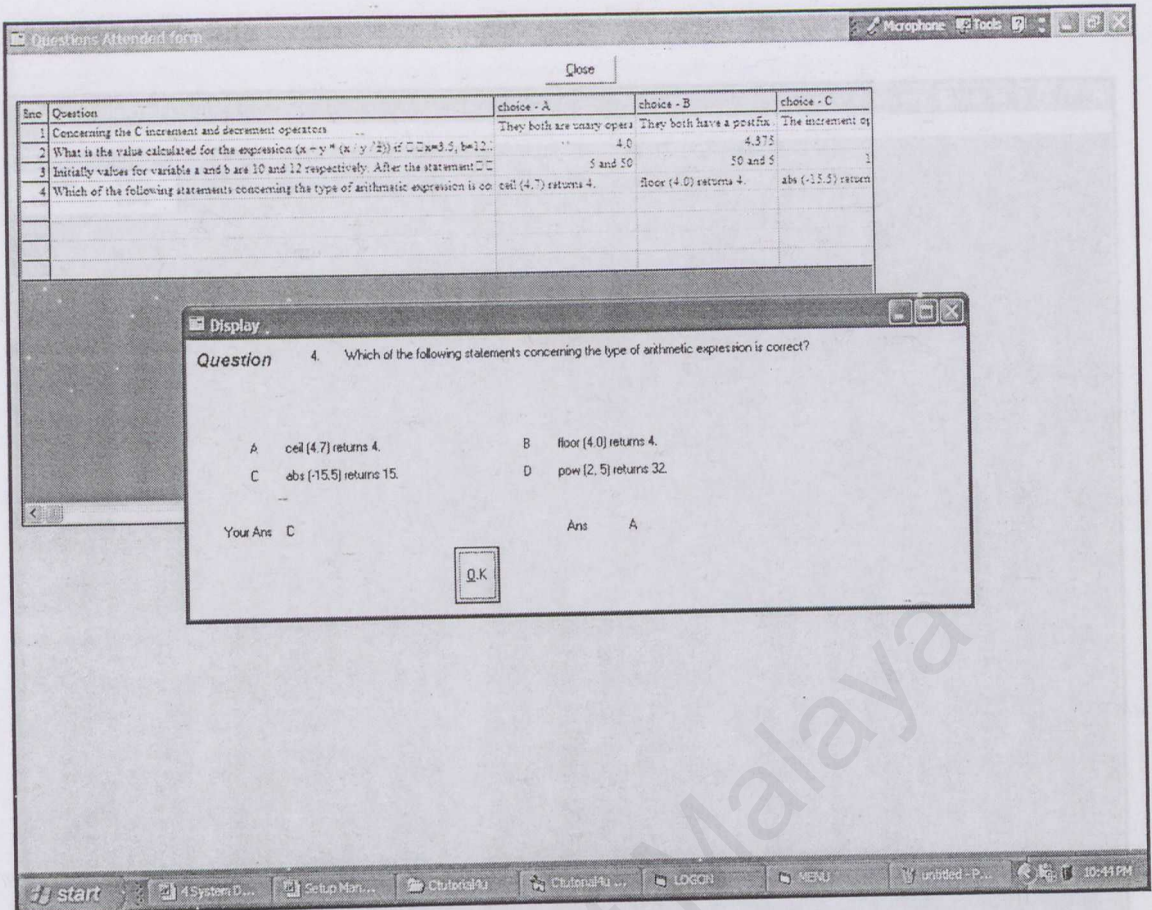


Figure C.23: Questions Attempted page

After exited from the previous quiz session, there will be a list of questions attempted in the previous session displayed in the questions attempted form as in the Figure C.23. Click on the question then details of the question will be displayed in the Display form. Click [OK]. Then click [Close] to close the form and go back to the menu form.

b) Admin Module

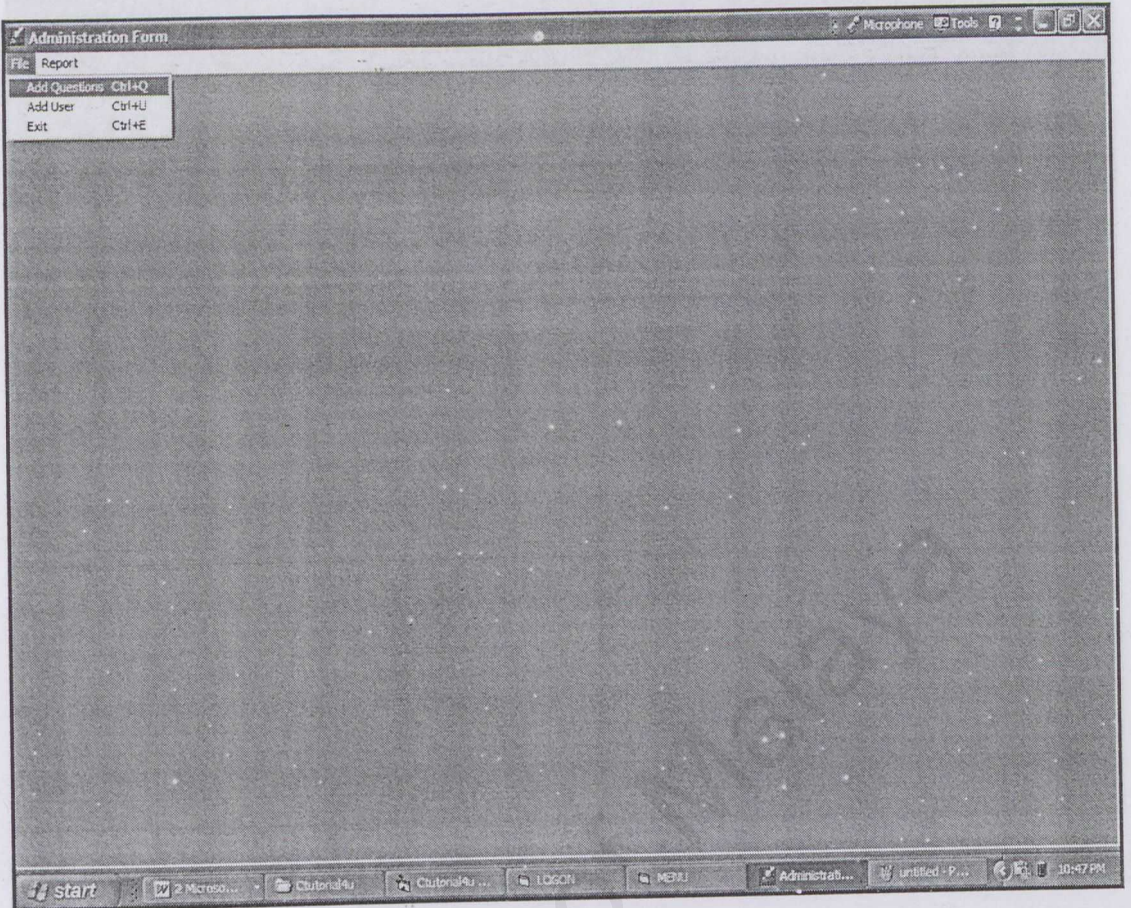


Figure C.24: Administration page

Admin page:

1. If the administrator wishes to add, edit and delete questions, click on Add question menu.
2. If the administrator wishes to add, edit, delete user then click on the Add user menu.

Administration Form - [Questions]

File Report

Username

Add questions to chapter?

Question ID

Question

A

B

C

D

Correct Choice :

Add Edit Delete Cancel OK Exit

<< < > >>

start 2 Microsoft... Customized Customized... LOGON MENU Administrati... untitled - P... 10:59 PM

Figure C.26: Question page

The admin id that user input in the previous input box will be captured in the username text box of this page..

Add Question

1. Firstly, choose the chapter to add the question in the [Add questions to chapter?] combo box.
2. Just key in all the fields as displayed in the figure above.
3. Click [Add] command button to add the question.
4. A message will prompt to indicate the question successfully added.

Edit Question

1. Firstly, choose the chapter to edit a question in the [Edit questions to chapter?] combo box.
2. Key in the question ID to select the question to be edited.
3. Once the question details have been displayed, then edit the question.
4. Click [OK] command button to save the edited data.

Delete Question

1. Firstly, choose the chapter to delete a question in the [Delete questions to chapter?] combo box.
2. Key in the question ID to select the question to be deleted.
3. Once the question details have been displayed, then edit the question accordingly.
4. Click [OK] command button to save the edited data.

Administration Form - [User Addition]

File Report

Username: Asrul

Password: ****

User Type: Student

Add User Clear Close

start 2 Micro... Tutorial... LOGON MENU Administrat... untitled - P... 11:00 PM

Figure C.27: User page

The Figure C.27 shows the user management module.

1. Key in the username, password and user type either student or administrator type.
2. Then click [Add User] command button.
3. Click [Clear] button to clear the form.
4. Click [Close] button to close the page.
5. Once the details of the user successfully added then there will be a message prompt [User created successfully].