Matematik Asas For Higher Education

By
YOUNG CHOON HOONG
(WEK 990081)

Supervisor: Cik Nor Aniza Abduliah

Moderator: Encik Mohammad Nizam

A thesis presented to the Faculty of Computer Science and Information Technology of University of Malaya in fulfillment of the requirement for the Degree of

BACHELOR OF COMPUTER SCIENCE (NETWORKING)

UNIVERSITY OF MALAYA

2001/2002

TABLE OF CONTENT

LIST OF TABLE

LIST OF FIGURES

ABSTRACT

This report was based on the final year project, as a partial requirement for the degree of Computer Science. The title of this project is "Matematik Asas" For Higher Education. This report contains three chapters, which are introduction of this project, literature review and methodology.

"Matematik Asas" For Higher Education is a project to create an educational package for student aged 17 and above. The primary objective of this project is to build up interesting educational software to give lesson to students especially in university on Mathematics Asas.

This project will mainly focus on multimedia interface to keep attention from students in learning this subject and ways to enhance current teaching problems. With this developed package, students could have alternative ways to learn mathematics and thus, improve their performance in the course.

This report will describe the system that will be develop, review on whatever had been found and analysis, and methodology that are using.

ACKNOWLEDGEMENT

I would like to extend a very thank you to my thesis's supervisor Cik Nor Aniza Abdullah. Thank you for her advice and guidance all this time.

I would like to express my appreciation to my moderator, Encik Omar Zakaria and Encik Mohammad Nizam, who had given me good idea and suggestion during I gave presentation in VIVA.

I would like to extend gratitude to all the students in University Malaya who had help me in completing the questionnaire because willing to share their experience and helping me in this project.

Lastly, I would like to extend appreciate to all my friends who had given me a lot of help and moral support especially Chong Yuen Wei, Murugan and my courmates.

Introduction

CHAPTER 1

INTRODUCTION

The twenty-first century is an era of information technology with computer technology being the integral component. Multimedia courseware will be a major component of learning in the information age.

The traditional instructional delivery system in today's education is unable to meet the rapid change in the education environment. The government vision is to revolutionize our country's education system by introducing information technology components in the learning environment. For example, introducing Smart School, one of the flagship applications in the Multimedia Super Corridor (MSC).

1.1 Project Definition

The "Matematik Asas" for Higher Education (MAHE) is an on-line CAL package that can fully manipulate the advantages of a multimedia presentation to support the learning process of student undergraduate in university. The aim of this software is to educate student undergraduate learn Mathematics in few alternative ways.

Multimedia technology is used as a platform to create this educational software. Multimedia was chosen because it provides an environment for interactive learning through the use of sounds, texts, graphics and animations that is able to capture the

interest of student undergraduate as well as promote effective learning. Moreover, multimedia is suitable for learning through discovery and exploration.

This software will in-directly contribute to the vision mention above, as by making student undergraduate familiar to the computer teaching material. They will better prepared for this revolution in the education system.

The benefits of computer aided-learning are numerous. Computers have the potential to reduce the learning difficulties as well as accelerate the learning process. Computer aided-learning has shown that student undergraduate have better understanding, more control and enjoyment while learning. These are the best benefits that they could learn at their own pace.

1.2 Project Objective

Basically this multimedia software will mainly emphasized on smart mathematics for higher education. The primary objective of this project is to review the current educational ways to find approaches to enhance the entire way to a more efficient system with the implementation of computer.

The second objective is to design and develop attractive user-friendly interface for learning. Thus the users will have more understanding on their studies. Besides, this will help the student undergraduate to have more interest in studying this course.

In addition, the objective of this project is to keep the teaching material with latest technologies because the software will employs Graphical User Interface (GUI) and Object Oriented technology.

The last objective of this project is to help the users to keep track in their study performance in a more systematic manner.

Summary of objectives:

i. To enhance current educational ways

ii. To develop attractive user-friendly interface for learning

iii. To keep the teaching material with latest technologies

iv. To help the users to keep track in their study performance

1.3 Project Scope

This multimedia package will focus on student undergraduate in the range of 17 years old and above. The scope of this software will consist of:

1. Creating a learning module for chapter that consists of complex number, percentage, types of numbers etc. Besides, example of question with answer, tips on study and important points will be included.

2. Tutorial and test that will evaluate the students understanding of the material studies using the computer because each question will be come along with answer and also the correct way to do.

3. Besides, the software will have an overall view to brief user on overall chapters and tutorials. This will give a preview to the user about the content, list of figures, tables, formulas and theorems.

4. In addition, an analysis of past years' questions will be added into this software. This will give overall perspective to the users on the important parts.

1.4 Project Module

1. Registration Entry

Objective: This module allows user to register before using this software.

Sub modules:

i. User's details

- e.g. name, nickname, age, education level, login identity, password etc.

ii. Login identity

- for existing users, they can login using their own password.

iii. Previous records or last performance

- show the last records for existing user.

2. Overview

Objective: This module briefs user on overall chapters and tutorials. This will give a preview to the user.

Sub modules:

i. Content

- show all the chapters and tutorials with details.

ii. List of figures and tables

- show a list of figures and tables.

iii. List of formulas or theorems

- show a list of formulas and theorems.

3. Courses

Objective: This is the main part of the system where all the chapters in subject "Calculus" will be covered in an interesting way.

Sub modules:

i. Chapters include all theorem and formula.

ii. Example of questions with answer.

iii. Tips on study.

iv. Important point.

v. Favorite ask question (FAQ)

- optional.

4. Tutorials

Objective: This module will include all tutorials; each chapter will have one tutorial.

Sub modules:

i. Questions and answer and also the correct way to do.

ii. Comment.

iii. Marks and record.

iv. No time consuming.

5. Tests

Objective: This module will have those past year paper and analysis question from which chapter. Moreover model paper will be included also.

Sub modules:

i. Few tests where each test will cover 2-4 chapters depend on how many scopes in that particular chapter.

ii. Marks and record.

iii. Time consuming.

1.5 Expectation Output

After research and review on current teaching style (will be explain in the following chapter), in conjunction to enhance the current system, this project will have the following expectation outcome.

This software output include:

- Interface that include multimedia or animation application.
- A database that store user performance.
- User could select any chapter from menu.
- Answer and explanation will be given for tests and tutorial.
- Ways to motivate students in learning the subject

1.6 Limitation and Constrain

Due to time limitation and wide range of coverage, this project will have a limitation.

- This project will only include main chapter of the "Matematik Asas" but not all chapters.

1.7 Data Flow

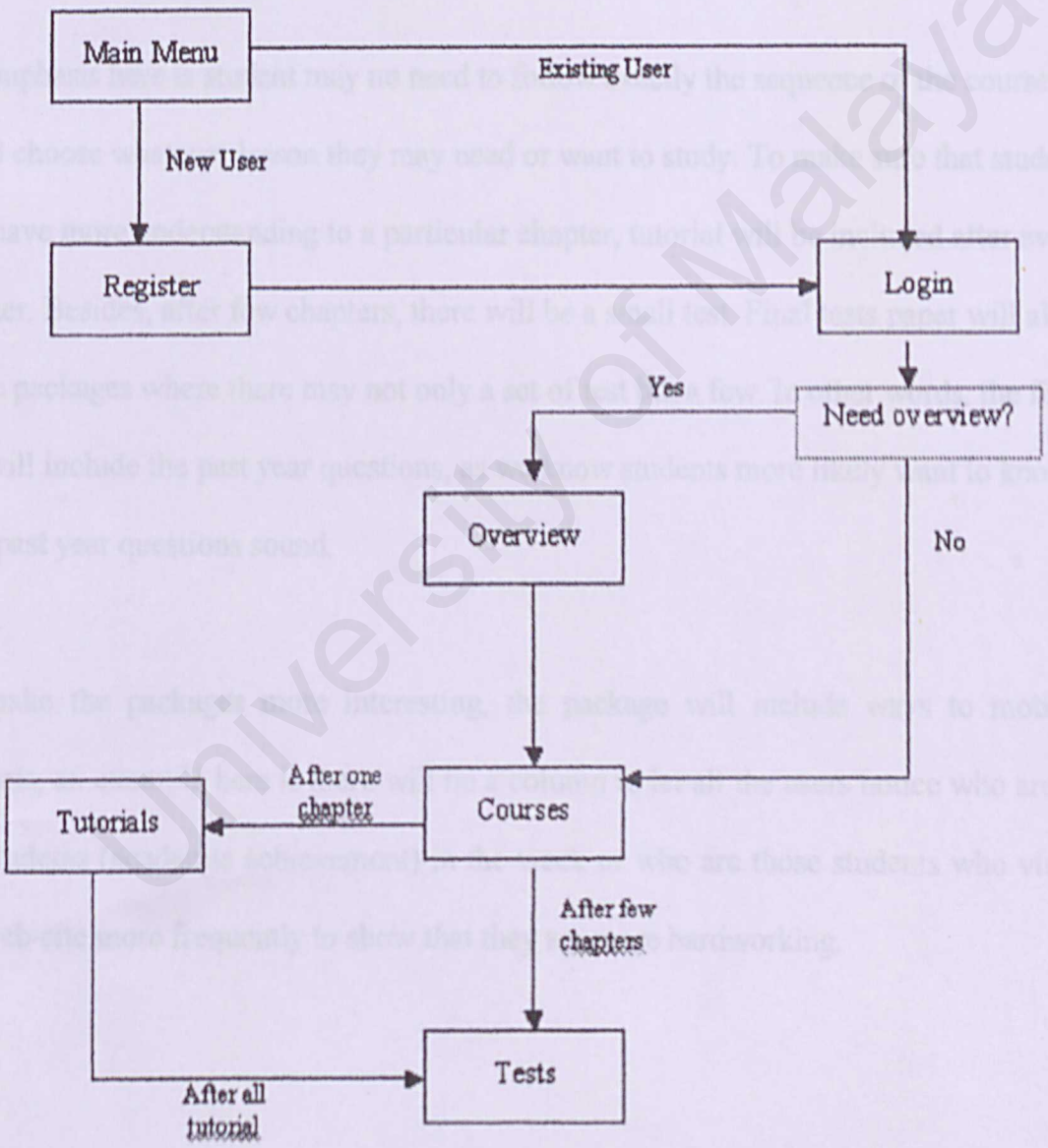


Figure 1.1 Flow chart for MAHE

When a user or here called student enters the web site, he or she may either an existing or new user. For existing user, he may already had previous information, thus he could log in as existing user and choose courses. For a new user, he or she may need to register first, where a small database will store the information and their performance. Then the flow will move to whether the user need to have overview on all the courses or only interested to pick up a lesson.

An emphasis here is student may no need to follow exactly the sequence of the course but could choose whatever lesson they may need or want to study. To make sure that students will have more understanding to a particular chapter, tutorial will be included after every chapter. Besides, after few chapters, there will be a small test. Final tests paper will also in the packages where there may not only a set of test but a few. In other words, the final test will include the past year questions, as we know students more likely want to know how past year questions sound.

To make the packages more interesting, the package will include ways to motivate students, an example here is there will be a column to let all the users notice who are the top students (academic achievement) in the week or who are those students who visited the web-site more frequently to show that they are more hardworking.

1.8 Project Scheduling

Figure 1.2 in the following page is the project schedule for the whole project included phase I and phase II. The first phase of the project was completed in approximately one and a half month.

No.	Task Name	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan
1	**Information Gathering and Survey**								
	➢ Briefing with lecturer.								
	➢ Identify problems and objectives.								
	➢ Estimate scopes.								
	➢ Searching information from various sources								
	➢ Prepare questionnaire and interview questions.								
	➢ Conduct survey in the university.								
2	**Analysis and Design**								
	➢ Analyzing interviews and surveys results.								
	➢ Determine system module.								
	➢ Design and draw the outline for the user interface.								
	➢ Design the recommended system.								
3	**Documentation 1 (Proposal)**								
	➢ **Documentation.**								
4	**Coding and Development**								
	➢ Develop system.								
5	**Testing and Implementation**								
	➢ Testing and maintaining the designed system.								
	➢ Implementation of the system.								
6	**Documentation 2 (Report)**								
	➢ **Documentation.**								

Figure 1.2 Project Schedule

Literature Review

CHAPTER 2

LITERATURE REVIEW

2.1 The Use of Computer in Teaching and Learning

2.1.1 Learning Categories

The five categories of learning are Attitude, Motor Skill, Cognitive Strategies, Verbal Information and Intellectual Skill. This package will make use of the three of these categories, which are Attitude, Cognitive Strategies and Intellectual Skill. The table below shows the computer teaching involvement in the different kinds of teaching.

Kind of Learning	"The student will" (behavioral action)	Computer's Teaching Involvement
Attitude	Choose something to do.	Indirect, rather than direct. For example, a mathematics game drill and practice program tries to make practice enjoyable and something the student would choose to do.
Motor Skill	Move finger, body parts.	Limited; For example, student learn to use the mouse and keyboard.
Cognitive Strategies	Efficiently or incisively do one of the other categories of learning.	Potential for use in creative problem solving, inventing a design or work of art or music, or devising a plan for searching a database.
Verbal Information	State, describe list, name.	Very heavily used; drill, practice and tutorials.
Intellectual Skill	Discriminate, identify,	Can do well in tutorials, drill and

	classify, demonstrate, generate.	practice, simulations and problem-solving programs.

Table 2.1 Five Categories of Learning

2.1.2 The Ways Learning Using Computer

There are a number of ways in which computers are being used in the educational process. These range from applications in which the computer supplements or replaces the teacher to those in which it is introduced into the student's learning environment as an additional resource or tool. Some of the more widely accepted uses are as follows:

1. Drill and Practice Method

   Drill and practice programs take advantage of the computer's endless patience and ability to provide immediate feedback and reinforcement, which can be tailored to a student individual needs. Drill and practice programs provide a variety of questions with varied formats. The programs are structured to in various ways to provide as much as possible, a non-threatening learning situation for students.

   Several levels of difficulty can be available within the same drill and practice program. Positive and negative feedback as well as reinforcement can be included. It is also possible to automatically record student's performances and to make this data available to the student. This package will make use of practice and drills.

2. Tutorial Method

   In the tutorial role, the computer acts as the teacher. All interactions are between the computer and the lecturer. In this method, information is presented in small units

followed by a question. The computer (compared with responses plugged in by the author) analyzes the student's response and appropriate feedback is given. A complicated network of pathways or branch can be programmed. The more alternative available to the computer, the more adaptive the tutorial can be to individual difference. Thus the computer can achieve teaching flexibility by providing immediate, real-time instructional decision based on the student's last response or upon an evaluation of his or her past response.

3. Instructional Gaming Method

Instructional Gaming Method is classified as a game because it employs a definite set of rules and strategy for winning. Instructional games appeal to a student sense of play in meeting educational goals. They address the area of basic skill but also provide situation in which problem-solving skills must be utilized to win. The power of the computer allows most games to be more complex than non-computer games, resulting in the teaching of multiple skills and the concepts in a way that the learner enjoys.

4. Simulation Method

Simulation approach is deductive in the sense that the student is involved in investigating the model of situation by systematically altering values of the parameters. In this method, the learner confronts an approximation of a real-life situation. It allows realistic practice without the expense or risk otherwise involved or might be impossible to provide in the classroom. The learner is expected to use

their own previous knowledge of the topic to enable them to plan a series of investigations on the simulated environment and to respond and draw conclusions from output.

5. Discovery Method

Discovery is a general term to describe activities using an inductive approach to learning; that is, presenting problems that the student solves through trial and error or systematic approaches. Using the Discovery method in Computer-Based Instruction Learning lets the learner employs an information retrieval strategy to get information from a database.

6. Problem Solving Method

In the case of problem solving, the student is obliged to interpret a problem and express a solution as an algorithm in the form of a computer program, the purpose of which is to test the validity of the algorithm. The point of this activity is for the student to develop problem-solving skills such as:

- Starting with a simple specific case
- Exploring the nature of errors
- Attempting to generalize
- Effects to gain further insight into the problem
- Learning that there are frequently a number
- Of algorithms which characterize a given problem and its solution

At the present time computers are probably most widely used in support of this problem solving activity.

Table 2.2: Utilization of various CAL methods shows a summary of all the methods of learning, with their description as well as the roles of the computer and the students in these methods.

Methods	Description	Role of Computer	Role of Student
Drill and Practice	Content already taught. Reviews basic facts and terminology. Variety of questions in varied formats.	Ask questions "evaluate" student response. Provide immediate feedback. Records student progress.	Practice content already taught. Responds to questions.
Tutorial	Presentation of new information. Teaches concepts and principles. Provides remedial instruction.	Presents information. Ask questions. Monitor responses. Provide remedial feedback. Keep records	Interacts with computer. See result. Answer questions Ask questions.
Gaming	Competitive. Drill-and-practice in a motivational format. Individual or small group.	Acts as competitor, judge and scorekeeper.	Learn facts, strategies and skill. Evaluate choices. Competes with computer.
Simulation	Approximate real-life situations. Based upon realistic models.	Play roles. Deliver results of decisions. Maintains the model.	Practice decision-making. Makes choices. Evaluates decisions.

Discovery	Inquiry into database. Inductive approach. Trial and error. Tests hypotheses.	Presents student with source of information. Stores data. Permits search procedures.	Makes hypotheses. Test guesses. Develop principles or rules.
Problem Solving	Define problems. State hypothesis. Examine data. Generate solution.	Present problems. Manipulate data. Maintains database. Provides feedback.	Defines the problem. Set up the solution. Manipulate variables. Trial and error.

Table 2.2 Utilization of Various CAL Methods

2.1.3 Rationale Using Computer in Education

1. Individualization of Instructions

   Many educators believe the more individualized the teaching, the greater the learning. Based on the premise that better learning occurs if student is taught individually to achieve to the extent of his ability at his own pace, the tailored program to his interest, abilities and learning rate. Through learning options and a continuing process of self-assessment the student is encourage to become self-directed and self-initiating learner.

2. Contribution of Learning Mastery

   The learner may test himself after each learning activity. In competition only with himself, the learner can evaluate his performance, and if not satisfactory may return to the learning stage and explore another set of materials. A per-test, administered before the program begun, will help determine the entering level of knowledge. The

computer can use a student's history of learning successes and failures as a basis for selecting new problems for leaning. As the computer selects alternative learning paths of varying difficulty, the progress of the student is monitored.

3. Computers Provide more Quality Material and is Available to more People

A wide range of materials to accommodate different styles, motivations, sensory preferences and capabilities can be made available to large number of learner. The Internet places extensive libraries of instructional programs at the fingertips of users.

4. Computers use Emphasis on Analysis

The use of computer sensitizes the learner to the need for self-assessment, providing him with information about his progress and failure. He becomes more self-aware, more constructively self-conscious and able to analyze his need.

5. Computers Provide a Mean of Educational Reforms

The traditional method of instruction threats every student in the classroom; regardless of educational level is exposed to the same educational experience. The use of computers provides an alternative structure of education. An educational environment in which students are free to follow their natural curiosity and to be independent learners.

2.1.4 The Benefits of Using Computers

Specific advantages of computer usage in Computer-Based Instruction Learning are:

- Simply allowing students to learn at their own pace produces significant timesaving over conventional classroom instructions. Computer-Based Instruction Learning allows students some control over the rate and sequence of their learning.

- High-speed personalized response to learner actions yields a high rate of reinforcement.

- The patient, personalized manner that can be programmed provides a more positive afflictive climate, especially for slower learner. Mistakes, which are inevitable are not exposed to peers and therefore are not embarrassing.

- Computer-Based Instruction Learning is effective as special needs can be accommodated and instruction proceeds at an appropriate pace.

- Color, music and animated graphics can add realism and appeal.

- The record keeping ability of the computer makes individualized instruction feasible. Individual prescriptions can be prepared for all students and their progress can be monitored.

- Computers can provide coverage of a growing knowledge base. They can manage information of all types such a graphic, text, audio and video. The learning experience can utilize a variety of instructional methods. More information is put easily at the instructor's disposal.

- The computer provides reliable and consistent instruction from learner to learner regardless of time of day and the location.

- Computer-Based Instruction Learning can improve efficiency and effectiveness. Effectiveness refers to improved learner achievement whereas efficiency means achievement objectives in less time or at lower cost.

- Computer forces the learner to communicate with it in an orderly and logical way.

- Computer users learn using the muse or typing skills.

In summary, the advantages of computer-based learning programs is that it provides immediate feedback, simpler individualization of instructions, increases social-problem solving skill and increases the time spent on learning task.

In addition, they can enhance preschooler's self-esteem, motivate these children, as well as allow the preschooler's parents or educators to evaluate their progress. Further more, it gives the children the opportunity to control their own environment. There is also a

substantial saving of learning time with the use of computer-based learning programs compared to the traditional method of instructions.

2.2 Multimedia Usage in Creating the Application Package

The term "MULTIMEDIA" is a combination of two words. "MULTI" which means much, many or more than one. "MEDIA" is digital data, sound, video, graphic and image. "MULTIMEDIA" means the combination of digital data, sound, video, graphic and image to present information on a computer. Multimedia usage is simply using a variety of media, might they be visual or auditory with the intent of communication. Multimedia production has become more accessible today with the help of inexpensive, powerful personal computers and may easy-to-use authoring languages.

Smart Mathematics for High Education falls in the category of interactive multimedia applications. This area of interactive application provides he application to a specific user as a function of response was developed using interactive, graphical application.

Multimedia is any combination of text, graphic art, sound, animation and video with links and tools that let the user navigate, interact, crate and communicate. When a person weave together the sensual elements of multimedia such as dazzling pictures and animation, engaging sounds and compelling video clips and raw textual information, it can electrify the thought and action centers of people's mind. Multimedia excites the eyes, ears, fingertips and most important the head.

2.3 Multimedia Building Block

✓ Sound

Sound combined in multimedia software provides information and enhances the other media being presented. There are three sound file formats that are widely used: wave (.WAV), sound (.SND), and midi (.MID) files. The first two file formats are formats used to record voice and sound effects. The third audio type is midi (music instrument digital interface) and is used to create digital sound form midi-compatible instruments.

✓ Image

Most often used to enhance the appearance of a multimedia presentation or to add important information. Images come in a variety of formats: compressed or uncompressed, bitmapped or vector, color of gray-scale. In general multimedia application work with bitmapped images as opposed to vector images. Bitmapped images are pixel-by-pixel representation of image, whereas vector images are composed of may, independently drawn objects composing an image. The most common image file types are Windows bitmaps (.BMP), device independent bitmap (.DIB), PC Paintbrush (.PCX), TrueVision (.TGA), CompuserveGIF (.GIF), Joint Expert Photograph Group (.JPG) and Tagged Image File Format (.TIF).

✓ Animation

There are two main types of animation used in multimedia applications. The first type of animation is the simple movement of objects on the screen, generally through linear translation. The creation is time consuming, very technical and artistry is involved. The third-party software used to create sophisticated animation is often expensive and used as exclusive tools without relationship to multimedia. The most common animation file formats are AutoDesk Studio and Animator (.FLI, .FLC) and Macromedia Director (.MMM) files.

✓ Video

The embedding of video in multimedia applications is a powerful way to convey information, which can be, incorporated a personal element, which other media lack. Compression techniques are used with digital video and a result resolution is compromised. Storage of video files requires a comparatively large amount of hard disk space. High-speed processors can play full-screen video where as slower computers, the play back in a window about one-quarter the size of screen.

✓ Text

Multimedia uses text, as it is an effective way to communicate ideas and provide instruction to users. Text is basically used for tittles and headlines, menus, navigation and to show content.

2.4 Computer System Design

2.4.1 Introduction

Designing a computer system is very different from designing an algorithm:

- The external interface (that is, the requirement) is less precisely defined, more complex, and more subject to change.
- The system has much more internal structure, and hence many internal interfaces.
- The measure of success is much less clear.

The designer usually finds himself floundering in a sea of possibilities, unclear about how one choice will limit his freedom to make other choices, or affect the size and performance of the entire system. There probably isn't a 'best' way to build the system, or even any major part of it; much more important is to avoid choosing a terrible way, and to have clear division of responsibilities among the parts.

Hints for computer system design are illustrated by a number of examples. They range from hardware such as the Ethernet local area network and the Alto and Dorado personal computers, through operating systems such as the sds 940 and the Alto operating system and programming systems such as Lisp and Mesa, to application programs such as the Bravo editor and the Star office system and network servers such as the Dover printer and the Grapevine mail system. There are references for nearly all the specific examples but for only a few of the ideas; many of these are part of the folklore, and it would take a lot of work to track down their multiple sources.

2.4.2 Functionality

The most important hints, and the vaguest, have to do with obtaining the right functionality from a system, that is, with getting it to do the things you want it to do. Most of these hints depend on the notion of an interface that separates an implementation of some abstraction from the clients who use the abstraction. The interface between two programs consists of the set of assumptions that each programmer needs to make about the other program in order to demonstrate the correctness of his program. Defining interfaces is the most important part of system design. Usually it is also the most difficult, since the interface design must satisfy three conflicting requirements: an interface should be simple, it should be complete, and it should admit a sufficiently small and fast implementation. Alas, all too often the assumptions embodied in an interface turn out to be misconceptions instead.

The main reason interfaces are difficult to design is that each interface is a small programming language: it defines a set of objects and the operations that can be used to manipulate the objects. Concrete syntax is not an issue, but every other aspect of programming language design is present.

2.4.3 System Speed

This section describes hints for making systems faster, forgoing any further discussion of why this is important.

- *Split resources* in a fixed way if in doubt, rather than sharing them. It is usually faster to allocate dedicated resources, it is often faster to access them, and the

behavior of the allocator is more predictable. The obvious disadvantage is that more total resources are needed, ignoring multiplexing overheads, than if all come from a common pool. In many cases, however, the cost of the extra resources is small, or the overhead is larger than the fragmentation, or both.

- *Use static analysis* if you can; this is a generalization of the last slogan. Static analysis discovers properties of the program that can usually be used to improve its performance. The hooker is "if you can"; when a good static analysis is not possible, don't delude yourself with a bad one, but fall back on a dynamic scheme.

- *Dynamic translation* from a convenient (compact, easily modified or easily displayed) representation to one that can be quickly interpreted is an important variation on the old idea of compiling. Translating a bit at a time is the idea behind separate compilation, which goes back at least to Fortran 2. Incremental compilers do it automatically when a statement, procedure or whatever is changed. A simpler version of his scheme is to always do the translation on demand and cache the result; then only one interpreter is required, and no decisions are needed except for cache replacement.

- *Cache answers* to expensive computations, rather than doing them over. By storing the triple [f, x, f(x)] in an associative store with f and x as keys, we can retrieve f(x) with a lookup. This is faster if f(x) is needed again before it gets

replaced in the cache, which presumably has limited capacity. How much faster depends on how expensive it is to compute f(x). A serious problem is that when f is not functional (can give different results with the same arguments), we need a way to invalidate or update a cache entry if the value of f(x) changes. Updating depends on an equation of the form f(x + D) = g(x, D , f(x)) in which g is much cheaper to compute than f. For example, x might be an array of 1000 numbers, f the sum of the array elements, and D a new value for one of them, that is, a pair [i, v]. Then g(x, [i, v], sum) is sum - xi + v.

- *Use hints* to speed up normal execution. A hint, like a cache entry, is the saved result of some computation. It is different in two ways: it may be wrong, and it is not necessarily reached by an associative lookup. Because a hint may be wrong, there must be a way to check its correctness before taking any unrecoverable action.

- *When in doubt, use brute force.* Especially as the cost of hardware declines, a straightforward, easily analyzed solution that requires a lot of special-purpose computing cycles is better than a complex, poorly characterized one that may work well if certain assumptions are satisfied.

- *Compute in background* when possible. In an interactive or real-time system, it is good to do as little work as possible before responding to a request. The reason is twofold: first, a rapid response is better for the users, and second, the load usually

varies a great deal, so there is likely to be idle processor time later in which to do background work.

- *Use batch processing* if possible. Doing things incrementally almost always costs more, even aside from the fact that disks and tapes work much better when accessed sequentially. Also, batch processing permits much simpler error recovery.

- *Safety first*. In allocating resources, strive to avoid disaster rather than to attain an optimum. Many years of experience with virtual memory, networks, disk allocation, database layout, and other resource allocation problems has made it clear that a general-purpose system cannot optimize the use of resources. On the other hand, it is easy enough to overload a system and drastically degrade the service.

- *Shed load* to control demand, rather than allowing the system to become overloaded. This is a corollary of the previous rule.

2.5 Learning Environments Enhanced by Emerging Technologies

Technology certainly has the power to make learning dramatically faster, better, more effective. But the methods of instruction now dominating the educational software market look more like traditional classrooms than immersive learning environments. Text predominates; lecturing, reading, and test-taking still prevail. All are necessary elements but also very boring.

Recent advances in learning sciences dictate that students move toward learning by doing. They know that the best methods involve creating perceptual environments for problem-based learning. This means immersing the learner in an experience requiring rapid-fire decision-making, fast acquisition of new information, and reliance on the expertise of colleagues. The ability to create and store useful knowledge allows individuals to innovate.

Yet there are few shining examples of actual implementation of these ideas. Industry solutions still wrap development efforts around familiar learning environments. Economics are partly to blame. But what really keeps schools, government, and industry from building next-generation learning environments is the lack of verifiable methods for integrating new learning-science techniques into today's software and hardware solutions.

Let's return for a moment to those immersive environments that have so enchanted the students. They offer visually rich, fast-paced, action-oriented, problem-based

environments that require skills like navigation, spatial recognition, strategic thinking, and relationship building. These sounds a lot like business in the digital economy.

What the software needs is rich and deep simulations for mathematics, built so that anyone can succeed based on competency instead of age, gender, race, or previous degree. We shouldn't call them games, but visual learning environments. And these new creations could actually track human performance. This is better than checking off bubbles on a piece of paper to determine human competency.

This educational software is not revolutionizing education right now because it doesn't build shareable visual objects that transfer from one environment to another with their inherent, functional properties intact. The software has done this with text, but barely even considered standards for visual objects. This software is a Internet-based learning world where participants are enabled by rich simulation tool-sets and create their own learning objects, enjoy embedded assessment for management of their own educations, and establish the profound people-to-people connections which allow for instruction, experimentation, and overall facilitation of the learning process.

We live in an era of innovators, quick thinkers, and technologically arrogant end-users whose demand more. It's going to be hard to answer this demand if the software focus only on teaching objectives, teacher preparation, state assessment tests, quality of education, and bridging the digital divide. These are important issues, but they tend to redefine the problem instead of leading to a solution.

Students have become enchanted by the high-end multiplayer games, interactive television, and wireless devices in which communication is rapid, visual stimulus is motivating, hundreds of activities and experiments can be conducted in the same environment, and reputations and human networking evolve at blinding speeds. In looking at the next generation of technology for education, the software have to consider creating similar environments that make technology ubiquitous and invisible, and at the same time enable human communication and interaction to accelerate the pace.

2.6 The Role of the E-moderator in E-learning

The key role of the online teacher or trainer is gradually evolving from one of ensuring the accurate transmission of known information to one of enabling exploration, and generating new and relevant knowledge for the use of individuals, groups, businesses, and not-for-profit organizations. Because of this shift, and the complexities involved in fulfilling the new role, e-moderator seems the most appropriate designation for it. Challenges to traditional approaches to learning are rife but they focus far too much on the role of the technology. E-moderator know that new modes of teaching and learning both in Higher Education and at work, together with the need for rapid updating of skills and knowledge, are emerging.

The advent of fourth generation learning delivery tools for use on campus, in the corporation and at a distance, offers the potential for enhancing education. There are increased opportunities for students: student interaction, student-centeredness, and

collaboration. Some argue that this can result in a situation where "the real learning 'space' among students is closer," whether on or off campus. The critical issue is that students most appreciate learning from others, with the support of a credible facilitator, and this is independent of the technological platform.

Online teaching and learning requires different skills, and changes what actually do with students. Currently, most online teachers do not have enough training to make the online experience truly successful and productive for learners. Where training is provided it often concentrates on the use of the technology rather than the role of the online teacher.

In professional and management education, and often where shared corporate knowledge is desired, the e-moderator is the person responding to and building on the contributions to an online conference. E-moderators need to be able to engage in reflective practice themselves, and be very democratic and open about their roles. The challenge is to enable managers to recognize the narrowness of their own experience and be open to other evidence. The e-moderator should prompt, encourage and enable such openness, while acknowledging the personal experience.

Managers can add value to online networking in a variety of ways. Firstly, the contributor needs to be acknowledged. Secondly, the contribution should be recorded and available for others to read online so it becomes a form of inventory. The e-moderator's role is to enable it to be retrieved, viewed, and responded to by others. In a collective conference, personal reflections may be responded to in various ways and in different time frames,

depending on individual thought processes. It is important that the e-moderator avoids the temptation to discount the experience in anyway or to counter it and enter into argument. Instead he or she can draw on the evidence that is presented to try and explore overall conclusions. Thirdly, the e-moderator should comment, at an appropriate moment, on the sufficiency of the data being presented and the quality of the argument around it.

Implementing these practices ensure that the experience, while it is valued, is not necessarily considered complete on its own. It also enables the e-moderator to exemplify ways of exploring and developing arguments. When this is done well, the result is a rich discussion with high levels of participation.

2.7 Math on the Brain

Human beings seem to have an inborn mathematical ability. Research has shown that even tiny babies seem to have a built-in awareness of numbers. But this is not the only way our brains process mathematics.

Researchers in France and America have recently confirmed that there are two distinct ways in which human deal with numbers and mathematical relationships: an "intuitive" mode, that may involve a visual component, and a "linguistic" mode developed later in life.

Human use their intuitive ability to make rough estimates, or recognize obvious errors (such as 16 + 8 = 168) without specifically calculating, and to perceive relationships between quantities. Human use their linguistic mode to perform more precise calculations and give them shortcuts (for example, knowing their times tables - which human learn verbally - allows them to avoid laboriously calculating products in their heads as a sequence of sums).

Earlier work with brain-injured patients had shown that some patients were able to subtract (a quantity-based operation) but not multiply (a verbally based operation), and others were able to multiply but not subtract. This suggested that different parts of the brain were used for the two activities. It seemed that learning a multiplication table was like memorizing a shopping list, whereas learning how numbers relate to each other might be tied to visual intuitions about space.

The more recent research has confirmed this two-mode theory and shown where the two different abilities are located in the brain. It seems that the verbal mathematical ability, the "linguistic" mode, is located in the left frontal lobe, which is the part of the brain known to be responsible for making connections between words.

However, mathematical estimation, the "intuitive" mode, was found to involve both the left and right parietal lobes, which are responsible for visual and spatial representation and also for controlling the fingers.

Interestingly, it has long been known that patients with damage to the parietal lobes suffer from "acalculia", a difficulty with basic number skills (for example, being no longer able to count). The recent research, locating mathematical estimation to the parietal lobes, helps to provide an explanation for this phenomenon.

2.8 Current System Problems

When teaching any subject, it is always best to let teachers know their students well- their likes, dislikes, interests, hobbies, favorite things, common and unique experiences, etc. The more you know about them, the more adeptly and securely you will drive their mathematical hooks.

That is the teacher's goal- to get the student to "own" the information- to help them learn it well enough to teach it, remember it for all time, and to use it meaningfully, quickly, easily, appropriately, and profitably. Anyhow, this consider very difficult in our current teaching environment since in a small lecture hall or tutorial rooms there may consist of hundred and above students.

The language of mathematics is used to communicate ideas, properties, relationships, and behavior concerning defined actions, objects, symbols, sets or collections, classifications, organization, quantity, size, shape, order, space, time, form, velocity, speed, distance, magnitude, etc. Each math idea has three components: (1.) linguistic, (2.) conceptual, and (3.) procedural or skill. Each component requires deliberate specific and differentiated

instructional attention. Math learning should be interactive. A concept should be introduced using concrete objects, and with logical language emphasis that is tied to everyday quantitative language expressions. (Sharma 1990, 23) In the actual situation or in Malaysia educational environment, test and examination is much more important than learning and interesting, thus teachers do not have enough time to teach details like history of a theorem or other interesting information. Students and teachers were rushing out of time to catch up the examination either in school or government examination.

Gender differences in math skills are due more to social forces than to gender-specific brain construction and function. Gender differences can be eliminated by equalizing the activities and experiences of both boys and girls at every level of development. The social forces that direct a child's experiences and choice of activities lead to the differences in the neurological sophistication of boys and girls. (Sharma 1989)

For example, most studies show that girls do better than boys in math until the age of 12. Then boys dominate the subject. This difference can be explained by analyzing the gender-specific development of math prerequisite, spatial orientation skills. The main reason for this is the methodology of teaching in pre-school and elementary grades, where focus is on fine-motor skill development. (Sharma 1989)

To achieve high-quality educational evaluations, the evaluator must be an expert in the field of inquiry, and should possess the following competencies:

1.) ability to describe the situation being evaluated;

2.) ability to describe the evaluation's context;

3.) ability to conceptualize the purpose and appropriate framework for the evaluation;

4.) ability to select and identify relevant information needs, sources, and questions;

5.) ability to identify, select, and apply effective procedures and techniques for data collection, processing and analysis;

6.) ability to determine the value of objects being evaluated;

7.) ability to communicate plans and results effectively;

8.) ability to successfully manage the evaluation;

9.) ability to maintain ethical standards;

10.) ability to make adjustments in external factors that influence an evaluation; and

11.) ability to critique, revise, utilize, and learn from the evaluation experience. (Worthen, Sanders, and Fitzpatrick 1997, 511-513).

According to the survey results, many students felt that teachers which were teaching higher mathematics may have possibility not enough knowledge on that particular subject but the current problems facing was less man power as teachers.

"I feel so stunned that after all these years! For so long I have been 'vague and scatterbrained,' driving myself and everyone around me crazy with my 'absent-mindedness.' Sometimes I am dismissed by people who assume that I'm 'thick' because I can't do the most basic calculations, or get myself from point A to B without getting lost." A student during interview said this.

Many students felt hesitate to ask questions in class due to they though this may end up with laughter in class. Another possible reason was student may think their opinion may be too childish or may cause teacher's angry because they had not paying enough attention in class. Thus, most of the students kept quite when they had questions or when they would like to express their point of view.

2.9 Recommended way to teach and learn mathematics

There is no better way to do this than to begin presenting (teaching) and integrating mathematical concepts, matter-of-factly, from birth.

Nine Prerequisite Math Skills

The student must be able to:

i. follow sequential directions
ii. understand and apply classification systems
iii. order, organize, and sequence
iv. have command of spatial orientation and spatial organization
v. understand and employ estimation
vi. visually cluster objects
vii. recognize and extend patterns
viii. visualize
ix. think deductively
x. think inductively. (Sharma 1990, 24)

Every math concept should be considered using both types of thinking. Quantitative approaches use standard deductive reasoning, sequential, procedural and algebraic algorithms. Qualitative approaches use "visual, spatial, inductive, and pattern recognition strategies.

		Sharma's Recommended Teaching Sequence for Math Concepts
1	Inductive Approach for Qualitative Learners	a. Explain the linguistic aspects of the concept. b. Introduce the general principle, truth, or law that other truths hinge upon. c. Let the students use investigations with concrete materials to discover proofs of these truths. d. Give many specific examples of these truths using the concrete materials. e. Have students talk of their discoveries about how the concept works. f. Then show how these individual experiences can be integrated into a general principal or rule that pertains equally to each example.
2.	Deductive Approach for Quantitative Learners	Next use the typical deductive approach. • Re-emphasize the general law, rule, principle, or truth that other mathematical truths hinge upon. • Then show how several specific examples obey the general rule. • Have students state the rule and offer specific examples that obey it. • Have students explain the linguistic elements of the concept.

Figure 2.1 Teaching sequence for Math concepts

CHAPTER 3

SYSTEM ANALYSIS AND REQIUREMENT

3.1 Information Gather Methodology

In the phase of gather information, five different methods were used for capturing requirements for this project. These techniques include questionnaires, existing system and educational packages, observation, library research, and Internet research.

3.1.1 Questionnaires

A set of questionnaire had been give away to 100 students in University Malaya and the questionnaire had enclosed as Appendix A. Sampling 100 students had being conducts as 25 students from Faculty Computer Science and Information Technology, 35 students from Faculty Science which were majoring in mathematics, 25 students from Faculty Engineering randomly and another 15 more students were came from Faculty Educational. The questionnaires were collected on the spot and each student did not used more than 5 minutes to fill up the questionnaire.

3.1.2 Existing System and Educational Packages

An analysis on the current system and educational packages was done to get an overall perspective on the mechanism and workflow of the system.

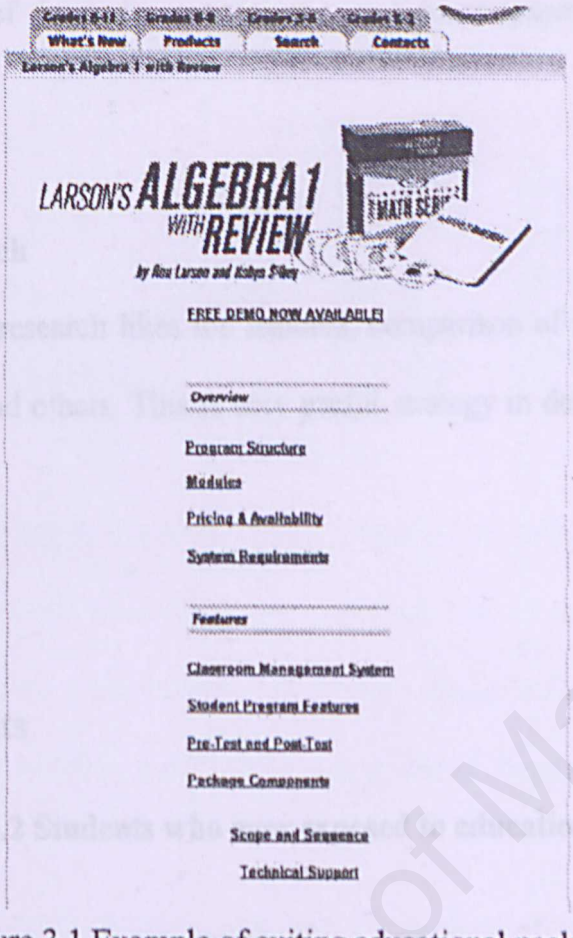


Figure 3.1 Example of exiting educational package

3.1.3 Observation

Through this method, the current study environment and problems could be understood. This method was done during attending lesson "Matematik Asas" in Faculty Science.

3.1.4 Library Research

This traditional approach to research was conducted to study the trend and evaluation of the procurement process, from the history of teaching materials, styles, and ways to learn mathematics to electronic procurement to further enhance the project. This was done by

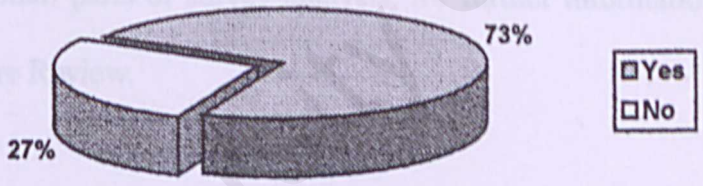
referred to variety of journals, magazines, conference papers and other relevant documents.

3.1.5 Internet Research

This modern tool for research likes the features, comparison of software and hardware, system requirement and others. This is very useful strategy in determining improvement to the current system.

3.2 Analysis Results

Figure 3.2 Students who ever exposed to educational software



In analysis survey result, 73 students among 100 sampling agree that they had expose to educational software before while the others 27 students saying that they had no experience interact with such software. Anyway this result reflected that the students commonly had tried these types of software either privately or in school.

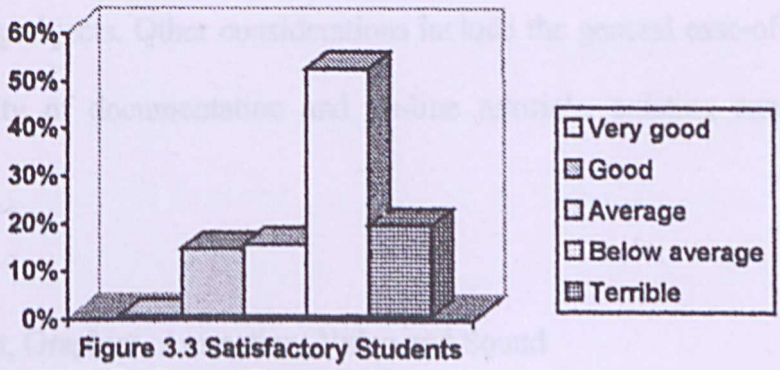


Figure 3.3 Satisfactory Students On Current Teaching Ways

From the chart, is obvious to found that most students that were 52% of the sampling students feeling the current teaching styles in lecture hall or tutorial rooms were not effective to let them learn mathematics. The reasons students felt that current teaching systems were found not suitable, was explained in precious chapter.

There were only small parts of survey analysis, for further information please refers to chapter 2: Literature Review.

3.3 Development Tools Evaluation Criteria

There are several critical areas in which consideration was given when the development tools was selected for "Smart Maths for Higher Education". The following areas were considered before deciding on the authoring package:

1. Authoring Environment

   The authoring environment should have graphically-based tools and scripting tools. It should include capabilities such as clicking and touching, time-outs, recording the

number of tries, text input, key-press detection, click-able objects, push buttons, scroll bars and moving objects. Other considerations include the general ease-of-use of the tools, the quality of documentation and on-line tutorials, existing examples and vendor support.

2. Support for Text, Graphics, Animation, Video and Sound

   The authoring tool should be able to support and handle rich text, graphics, animations, video and sound. Another consideration was, the development tool should be able to handle file formats of the media being used in the application.

3. Specific Support for Learning Applications

   The ease with which standard templates could be created and modified was considered.

3.4 Comparison of Development Tools

This section examines the merits and demerits of some authoring tools. It is not possible to be completely objective about a software application and although a reviewer can try to use objective assessment inevitably personal experience and bias will color the true outcome or even be wrong.

3.4.1 Asymetrix ToolBook

A popular development tool, which uses a book and page metaphor, akin to HyperCard's stack and card idea. Toolbook has its own programming language called OpenScript. It is a general-purpose development tool that can be used to create courseware and also applications such as databases. It is suited for text handling and is ideal for application with large amount of text.

The developer is required to 'draw' objects such as buttons and fields on a 'page' or 'background', the behavior of which is controlled by 'scripts' either attached to the object or placed above in the 'object hierarchy'. This approach is very flexible and allows the developer a large scope but makes the software more complicated to learn and use.

Asymetrix of use of this software depends on the developer's previous experience. Asymetrix provides a ToolBook CBT Edition which, according to Asymetrix, includes all the power and features of Multimedia ToolBool 3.0 plus an expanded feature set tailored to the specific needs of courseware authors.

3.4.2 Visual Basic

Visual Basic is a programming system for Windows that is often used to organize and present multimedia elements. It is make up of controls (objects) that reside on forms. Visual Basic uses language code syntactically similar to GW-BASIC. The program is event-driven. Code is attached to objects and remains idle until called to respond to user or system-initiated events, such as a mouse click or system time-out. Controls are used to

create the user interface of an application, including command button, picture box, timer, menu bar and others.

Visual Basic provides flexible response to mouse and keyboard events. It can also be coded to show and hide the objects during the runtime. There are also powerful debugging commands to help isolate and correct code errors. Completed Visual Basic projects can be converted into executable files to run on stand-alone Windows application. Beside, the developer can utilize the application setup wizard to create a setup program on CD-ROM that allows the user to install the application.

3.5 The Chosen Development Tool

3.5.1 Macromedia Director 8

Macromedia Director 8 is a complete environment for the creation of multimedia. Think of it as an artist's canvas. Or, to use the metaphor that Director follows, a stage.

Users can fill this stage with their own production. Any element in the production is called a cast member or simply a member. The computer screen where the action takes place is a window call Stage.

The rest of the elements in Director also follow a theater or film metaphor, although some element names follow it better than others.

3.5.2 New Features in Director 8

Director 8 contains a wide array of new features, generated mostly from user feedback. Director users will also see performance increase in just about every aspect of the program. The following sections include a summary of those new features.

1. Authoring Improvements

   Director 8 introduces several major features that make multimedia authoring much easier, all of which are covered here:

   - The Property Inspector – instead of having dozens of different property dialog boxes throughout Director, the Property Inspector is a single window that changes format according to the selection. It can display information about members, sprites, interface elements and even the Stage itself.

   - Zoomable Stage – Director users can now zoom the Stage windows in or out to see greater or lesser detail. In addition, the Stage window can be extended beyond the bounds of the actual Stage so that user can work with elements that are beyond the Stage rectangle.

   - Cast Window List View – Director users can now choose between the standard "cell" view of a cast or a list view. The latter offers sorting by various properties.

- Lockable Sprites – Director programmers can now lock sprites in place. Although they can be easily unlocked, this prevents accidental changes to sprites, as well as preventing changes from multimedia authors working on the movie.

- Guides – Like page-layout programs, Director users can set horizontal and vertical guides on the Stage to help align objects.

- Asset Management Fields – Each cast member has a creator, a creation date, a modified date and a comments field. These features can use to organize and maintain the Cast.

2. Linked Scripts

Director 8 users can have external text files that are used as Lingo scripts. Movies can share scripts and easily use external text editors to edit them.

3. General Lingo Improvements

Overall, Lingo performance has been optimized. In addition, here are some specific new features:

- Timeouts – this new object type enables users to set timers that can call handlers at specific time intervals.

- *flushInputEvents* – users can clear mouse clicks and key presses that might have built up while users were waiting for a movie to load or for a transition to complete.

- *isOKToAttach* – Behaviors can be set to attach to only certain types of sprites.

- *scriptList* – the entire script list of a sprite can be checked and set with Lingo and the Score will reflect any changes.

- Bit Operations – the *bitAnd*, *bitNot*, *bitOr* and *bitXOr* functions enable users to perform bit operations on variables.

- Application awareness – the *activateApplication* and *deactivateApplication* event handlers enable users to detect when a projector is being sent to the background or brought to the foreground.

- Handler Detection – the *hander* and *handlers* functions enable users to test for the existence of a handler in a script.

- the *markerList* – an alternative to the *labelList* which returns a proper Lingo list of markers.

4. New Imaging Lingo

   For the first time, Director developers have direct control over every pixel in bitmap members. New commands, such as *copyPixel*, *setPixel*, *draw* and *fill*, enable users to create and alter bitmaps.

5. New Sound Lingo

   The Lingo sound engine has been totally rebuilt for Director 8. Not only has performance been significantly improved, but also new commands, such as *queue*, enable users to queues sounds in a Sound channel, complete with loop definitions, as well as pan and pitch shifting.

6. Publishing

   Director 8 introduces a whole new way to "save" users finished project. The Publish command, found in the File menu, instantly saves Shockwave-compressed movie, taking into account an entire set of options. Users now have many types of sample HTML files that can be generated, plus a variety of premade loader movies and options.

7. Scalable Shockwave

   The main improvement to Shockwave is that movies can scale like Flash movies. This means users can set their Shockwave movie to fill the user's window and it will stretch to do so, no matter what the size of the window.

8. Others New Features

(a) Multicurve Vectors – vector shapes can have more the one curve in them.

(b) Flash Communication – users can get and set variables in Flash movies imported or linked as cast members. Users can also call actions in them.

(c) Editable Text Input – inline IME (Input Method Editor) allows multibyte characters, such as ones used in non-English languages, to be entered into editable text members. Standard editing commands, such as copy, cut and paste, can also used in editable text members.

(d) New Library Behaviors – The Director 8 behavior library includes all Director 7 favorites, plus many new ones. There is an entire set of behaviors that uses the new image Lingo to create individual sprite transitions; another set exists to create a drawing canvas; and there's also a set to create a multi user chat room.

(e) Image-Compression Settings – users have the capability to use JPEG compression on bitmaps when saving the movie as a Shockwave movie. Users can select a different compression setting for each bitmap, plus default setting for the entire movie.

3.6 Development Approach

3.6.1 Systems Development Life Cycle

A Systems Development Life Cycle is a process by which systems analysts, software engineers, programmers, and end-users build information systems and computer applications. It is a project management tool used to plan, execute, and control systems development projects. There are eight basic principles:

1. Get the user of the systems involved.

2. Use a problem solving approach.
   - Identify the problem, opportunity or directive.
   - Understand the problem's environment and the problem's causes and effects.
   - Define the requirements of a suitable solution.
   - Identify alternative solutions.
   - Select the "best" solution.
   - Design and implement the solution.
   - Observe and evaluate the solution's impact. Refine the solution accordingly.

3. Establish phases and activities.
   - Systems planning.
   - Systems analysis.
   - Systems design.

- Systems implementation.
- Systems support.

4. Establish standards for consistent development and documentation.
   - Activities.
   - Responsibilities.
   - Documentation guidelines or requirements.
   - Quality checks.

5. Justify systems as capital investments.
   - Look at several alternative solutions.
   - Evaluate each solution for feasibility, especially cost-effectiveness.

6. Don't be afraid to cancel or revise scope.
   - Cancel project if no longer feasible.
   - Reevaluate costs and schedule if project scope is to be increased.
   - Reduce scope if project budget and schedule are frozen, but not sufficient to cover all project objectives.

7. Divide and conquer.

8. Design systems for growth and change.

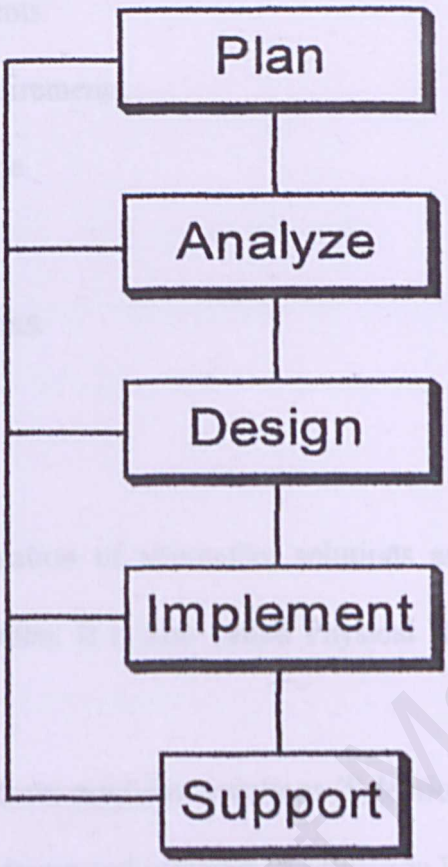


Figure 3.4 System Development Life Cycle

3.6.1.1 Systems Planning

The Systems Planning function of the life cycle seeks to identify and prioritize those technologies and applications that will return the most value to the organization. Systems Planning involves three basic phases:

i. Study the organization's missions.

ii. Define an information architecture.

- Data architecture.
- Network architecture.
- Activities or applications architecture.
- People architecture (i.e., Information Systems organization structure).

- Identify requirements.
- Model system requirements.
- Discovery prototype.
- Prioritization.
- Review requirements.

3.6.1.3 Systems Design

Systems Design is the evaluation of alternative solutions and the specification of a detailed computer-based solution. It is also called Physical Design. Systems Design is composed of three phases:

i. Select a design target from candidate solutions (Selection Phase).

ii. Acquire necessary hardware and software (Acquisition Phase).

iii. Design and integrate the new system (Design and Integration Phase).

- General design.
- Detailed design.

3.6.1.4 Systems Implementation

Systems Implementation is the construction of the new system and delivery of that system into "production" (meaning "day-to-day" operation). Systems Implementation involves four phases:

i. Build and test networks and databases (if necessary).

ii. Build and test the program.

iii. Install and test the new system.

iv. Deliver the new system into operation.

3.6.1.5 Systems Support

Systems Support is the ongoing maintenance of a system after it has been placed into operation. This includes program maintenance and system improvements. Systems Support is composed of four activities, in no particular order, but rather as problems arise:

i. Correct errors.
   - Software bugs.
   - Documentation errors or omissions.

ii. Recover the system.

iii. Assist the users of the system.

iv. Adapt the system to new requirements.

3.6.1.6 Reasons for Initiating Systems Development

Below is a bulleted list of some of the reasons that a system might consider systems development. Each of these list items contributes to the system's success by allowing or promoting growth and competitive advantage. Systems development might use one or more of several means, some of which are listed to the right of the image.

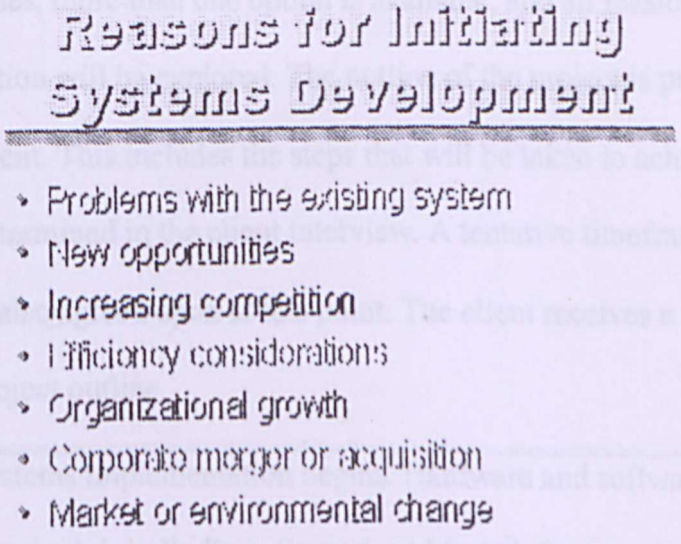


Figure 3.5 Reasons for initiating systems development

Table 3.1 Summarizes some of the ways in which the SDLC is used today.

Planning or Conceptual Design Phase	A client interview takes place. During this interview, the objectives of the situation are determined, and the fact-finding process begins. It links the business objectives of the organization and the proposed systems solution. The project plan outlines project scope, policies, philosophies, responsibilities, guidelines, major milestones, assumptions, and constraints. The resulting plan provides the blueprint to implement a successful, cost effective system.
Analysis	A more detailed systems survey and analysis, which is designed to determine the existence and nature of unmet or poorly met needs.
Design	A Manager and a focus team is assigned to the Project. The team designs a system that best meets the needs and the budget of the client. In most

	cases, more than one option is available, and all feasible courses of action will be explored. The outline of the project is presented to the client. This includes the steps that will be taken to achieve the objectives determined in the client interview. A tentative timeframe for completion is also agreed upon at this point. The client receives a written copy of the project outline.
Implementation	Systems implementation begins. Hardware and software systems are acquired, installed, configured, and tested. Systems implementation is usually the longest phase of the SDLC, and may require many iterations and adjustments in order to meet the project goals. The results of the project are tested against industry benchmarks and documented.
Maintenance	This phase includes continuously monitoring, maintaining, and modifying the system to ensure the system performs as expected, and continues to meet the user's dynamic needs. Continuous evaluation helps identify and prevent potential problems, pinpoint where maintenance costs can be minimized, and determine when modification or replacement activities should begin

3.6.2 Waterfall Approach

The waterfall model has been used to prescribe software development activities in a variety of contexts. Associated with each process activities were milestones and deliverables, so that the project managers could use the model to gauge how close the project was to completion at a given point in time.

The waterfall model can be very useful n helping developers lay out what they need to do and it is simple to explain to customers. Many other, more complex models are really just embellishments of waterfall.

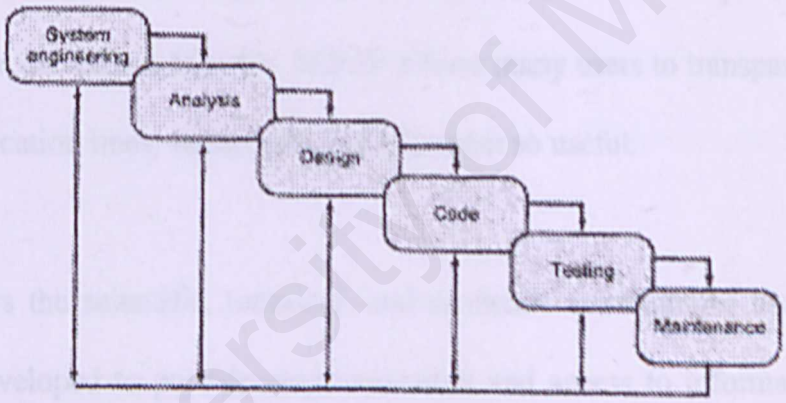


Figure 3.6 Waterfall model

3.7 Internet World Wide Web

3.7.1 Description

The Internet is a collection of interconnected computer networks communicating using a number of established standards and protocols. It started out as a Department of Defense network called ARPAnet which was designed with the goal of reliably sending message over an unreliable network. The Internet Protocol (IP) was established as a method of sending small packets (1500 bytes) of information with address wrappers through various paths to their destination. Transmission Control Protocol (TCP) was established to insure that the packets of information comprising a message all arrived intact at their destination, and that the message could be reassembled from the packets even if the packets where received out of order. TCP/IP allows many users to transparently share the same communication lines, which makes the Internet so useful.

For many years the scientific, technical, and academic communities used the Internet. Tools were developed to provide communication and access to information on remote computers attached to the Internet. Tools such as TELNET (remote login), FTP (file transfers protocol), and mail formed the basis for Internet use. Then a group of physicists several years ago designed a tool for linking documents together using what is referred to as hypertext (a method of embedding references to other documents in the text of a document). These references where in the form of URL's (Uniform Resource Locators) which contained resource/document type, location, and specific file name. The collections of documents connected via these links are referred to as the World Wide Web (WWW). Graphical browsers (e.g. Netscape, Mosaic, etc.) that interpret the

hypertext instructions were developed to assist in viewing and accessing these documents. Hypertext Markup Language (HTML), the interpretive language of the browsers, has become the standard language for World Wide Web documents. The ability to graphically browse interconnected documents residing on computers all over the world has resulted in an explosion of Internet use.

3.7.2 Usage

Many businesses, schools, individuals, and government agencies at all levels have rushed to establish a presence on the Internet. The attraction is the ability to quickly and inexpensively provide information to a large number of people. Most web pages are being developed as stand alone information sources for individual businesses or agencies. Few coordinated efforts exist to present a collection of web pages organized along State or municipal lines.

3.7.3 State of the Technology

The Internet and it's associated standards and protocols can be considered a mature technology. However it is undergoing rapid growth and expansion in many areas. It is estimated that the number of non-educational users currently at 13.5 million is doubling each year. Similar expansion exists in the number of commercial Internet service providers (ISPs), servers, domains, and web pages. HTML, the language for creating web pages, is firmly entrenched as a standard. The Netscape browser offers some enhancements to HTML which may or may not be adopted as part of the standard.

In this few years the Internet and its user base will be much larger. Operating systems (e.g. Microsoft Windows 98, OS/2 Warp, etc.) included browsers, and the TCP/IP software to support easy Internet access. Encryption software had widely available to protect information such as financial transactions over the Internet. As high-speed modems become cheaper most users will be accessing the Internet from home.

3.7.4 Strengths

The primary strength of the Internet\WWW approach to service and information delivery is how easy it is to quickly become accessible to a large number of users. For basic static information delivery the costs are minimal. The small investment required setting up web pages and paying an ISP yield huge returns in increased citizen access to government. For a slightly larger investment (server, router, communication lines) a dedicated presence on the Internet can be maintained which would provide for e-mail between agencies and the public. The emerging MIME (Multi-purpose Internet Mail Extensions) standard will enable the exchange of information beyond traditional textual messages.

3.8 Hardware and Software Requirements

The software and hardware that are required to develop this system include:

Hardware Requirement

- Computer with Pentium 133MHz or higher
- Sound card
- Modem
- Speaker
- Microphone

Software Requirement

- Windows 2000 as platform
- Macromedia Director 8.0 for create interface and processing
- Lingo scripts as developer's language
- Paint and PhotoShop 5.0 for creating and editing graphics
- Microsoft Access as database

System Design

CHAPTER 4

SYSTEM DESIGN

4.1 Introduction

System design includes hardware and software design. System design is transformation of the specification of requirements, first into a detailed logical (conceptual) specification and secondly into a detailed physical (technical) specification. Design is the creative process of transforming the problem into solution. Conceptual design tells the user exactly what the system will do. Technical design allows system builders to understand the actual hardware and software needed to solve the user's problem.

Thus, this system design will focus on technical or implementation concerns of the system.

All information and requirement that were gathered from previous chapter had lead to system design of this educational package. In this mathematics package, area of design included:

I) Overall design and Flow

II) Interface Design

III) Functional Design

4.2 Overall Design and Flow

The overall structure of educational software design is shown as below:

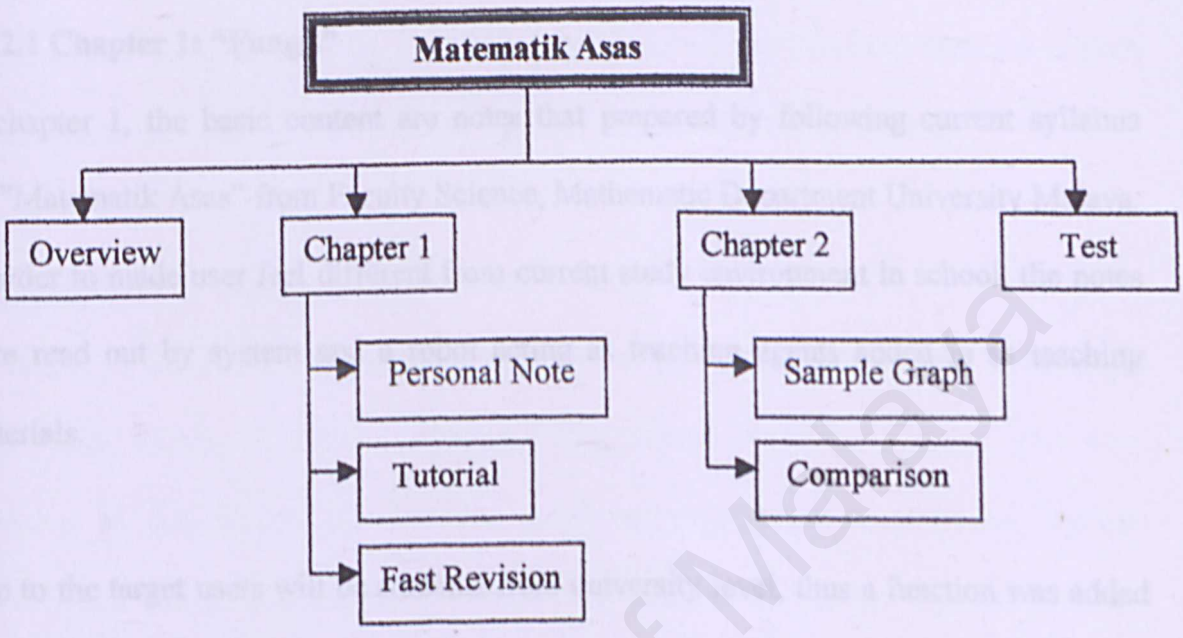


Figure 4.1 Overall Design Structure

4.2.1 Overview

This module briefs user on overall chapters and tutorials. This will give a preview to the user. User could select desired chapter from here. This package had use drop down menu to made user had deeper understanding on particular chapter's content. Besides, in order user need to exit system, the software prepared exit function button on each interface.

In addition, the package had added in voice instruction to give user instruction without needed them to read written instructions.

4.2.2 Learning Module

There are total up two main chapters included in the learning module.

4.2.2.1 Chapter 1: "Fungsi"

In chapter 1, the basic content are notes that prepared by following current syllabus for "Matematik Asas" from Faculty Science, Mathematic Department University Malaya. In order to made user feel different from current study environment in school, the notes were read out by system and a robot acting as teaching agents added in as teaching materials.

Due to the target users will be students from university level, thus a function was added in this module. According to the research from previous chapters, current students prefer to take notes as they were doing revision as well as attending lecture. Therefore, this package had prepared a small function to let user take note either by typing in or using "cut and paste" method. Besides, the note that users type in the prepared column will appear again in "Fast Revision" section. Through this, user could only view "Fast Revision" section when he or she doing revision.

Tutorial was one of the most important material to made students had deeper understanding on that particular chapter. From the information gathered through interview and questionnaires, students now prefer tutorial that act like games base. Thus, the tutorial in this chapter not only consider on user understanding but also consider how fast and accurate when user answering questions.

At first place, voice instruction will be read out in order to let user understand how to do the tutorial. In the tutorial, 1000 marks will be given at first for each question. Then, 1 mark will be keep on deducting as time goes. On the other hand, when user had selected a wrong answer for that particular question, 500 marks will be deducted from the current marks. When user had selected a correct answer, marks will stop deducting.

4.2.2.2 Chapter 2: "Geometri Koordinat Cartesan"

In chapter 2, the notes that prepared were following current syllabus for "Matematik Asas" from Faculty Science, Mathematic Department University Malaya. The main objectives in this chapter was learning graph and recognizing the different between graphs. Therefore, in this chapter user could have simplest ways to learn graphs.

Students could select any type of graph and the particular graph will appear at middle of the interface. This is not enough to understand the graph if no explanation is available. So, user could select explanation to have more understanding.

Another problem that students meet when study this chapter was they felt difficult to differentia among graphs. Thus, comparison was well prepared here to solve students' problems. Example using was comparison between graphs "Hyperbola" and "Ellipse".

4.2.3 Test Module

In this module, students had multiple choices to answer questions. Developer had used checkbox as selection. In some situation, user may had idea on previous question after answering the questions behind, thus the system had this facilities that could let user back to previous questions and change answer. When user confirming satisfied with their questions. He or she may select finish, then user will have their result.

4.2.4 Flow Control Diagram

Figure below shows overall flow control diagram for this package.

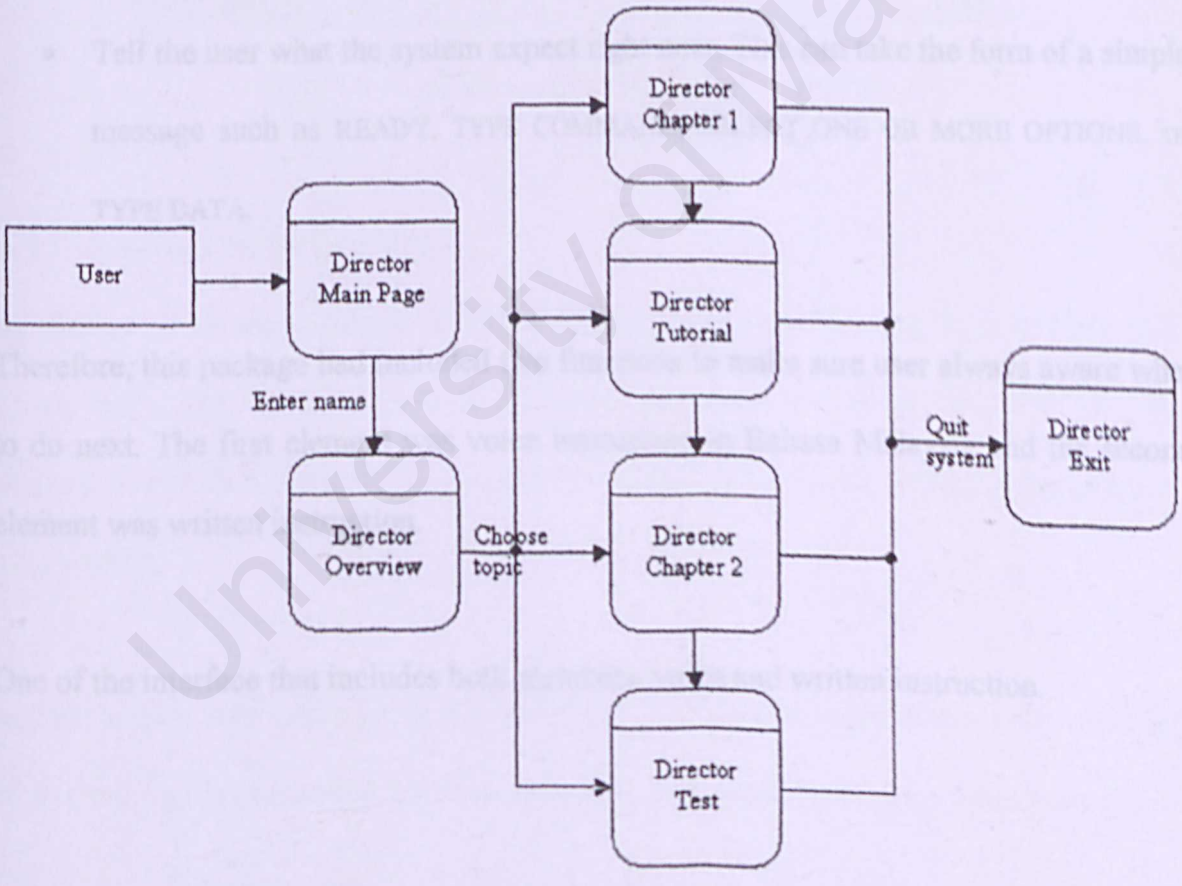


Figure 4.2 Flow Control Diagram

4.3 Interface Design

In an educational software package, interface design play very important role as it could determine how effective will they be. A number of important human engineering factors should be incorporated into the design.

4.3.1 Clear Instruction

The system user should always be aware of what to do next. The system should always provide instructions on how to proceed, back up, exit, and the like. Several situations require some type of feedback:

- Tell the user what the system expect right now. This can take the form of a simple message such as READY, TYPE COMMAND, SELECT ONE OR MORE OPTIONS, or TYPE DATA.

Therefore, this package had included two functions to make sure user always aware what to do next. The first element was voice instruction in Bahasa Malaysia and the second element was written instruction.

One of the interface that includes both elements: voice and written instruction.

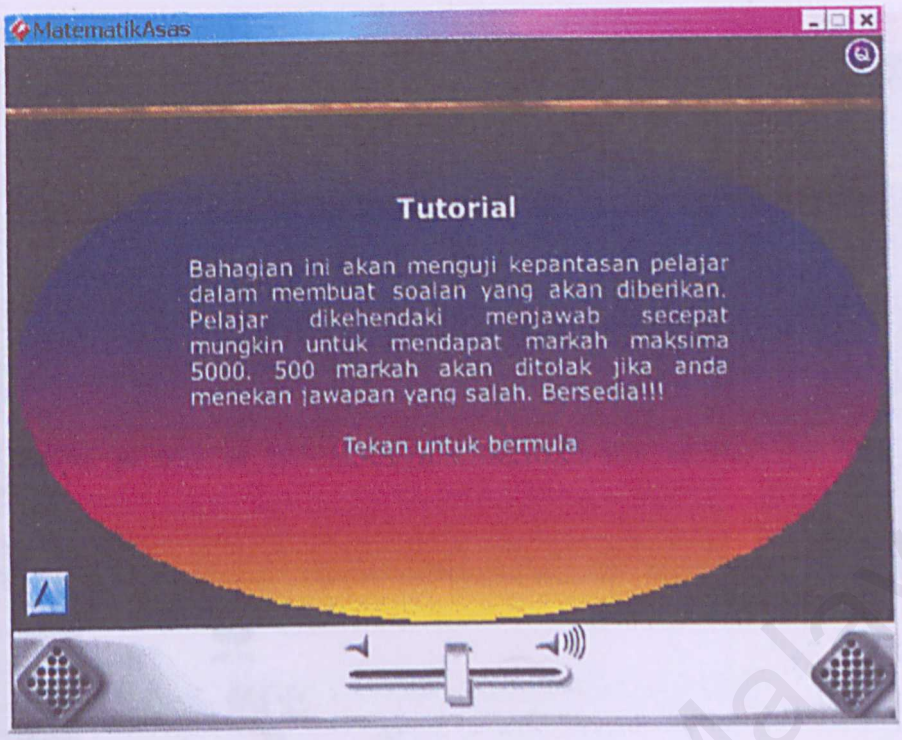


Figure 4.3 Interface at the beginning of tutorial

4.3.2 Display attributes sparingly

Use display attributes sparingly. Display attributes, such as blinking, highlighting, and reverse video, can be distracting if overused. Judicious use allow developer to call attention to something important – for example, the next field to be entered, a message, or an instruction.

In order to help user when using this software, tools tips for each button had added in. For example in the following interface tools tips "previous" will show whenever mouse near.

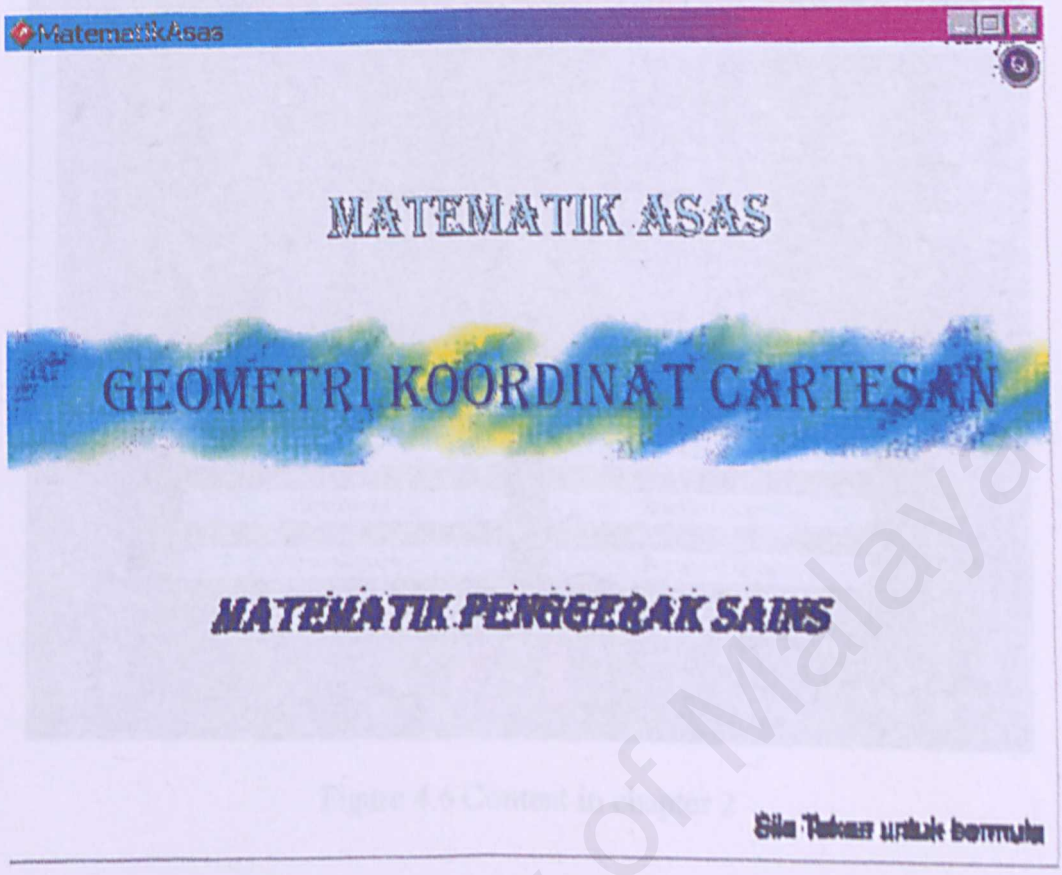


Figure 4.5 Interface at beginning of chapter 2

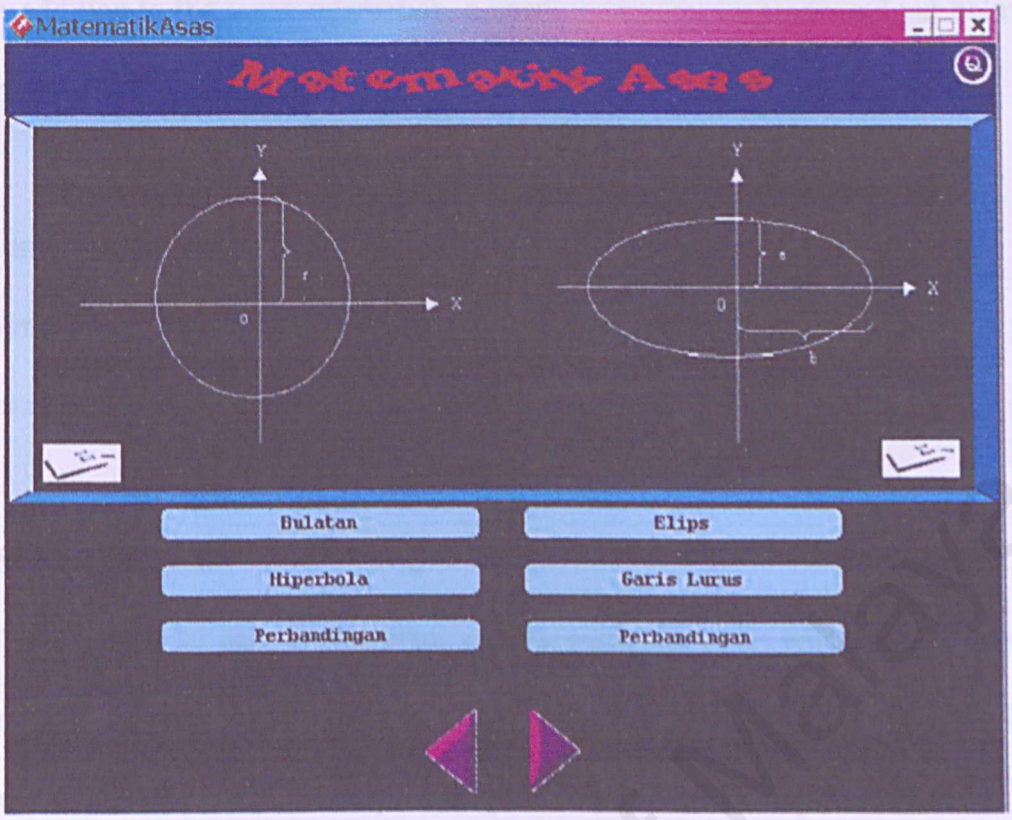


Figure 4.6 Content in chapter 2

4.4 Functional Design

In order to make this educational software different from current software, few functions had added in the software module.

I) Voice instruction

Voice instruction that added in the software created to make user understand what to do next. Besides, using human voice to give lecture although similar in schooling but more formal and intones will made software more interesting.

In addition to the voice instruction, a small voice control mode had put below every interface to control the volume of voice.

II) Simple calculation

Simple calculation such as total up how many correct answer does user provide, marks that earn, time taken to answer questions and etc.

III) Summarize notes

This function let user to take notes and the particular notes will be paste at one of the interface named Fast Revision. This function is useful when user needed express revision. User needs not to read through whole section for express revision.

Implementation

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

From the previous phases, a lot of information had gained to help in understanding users' problem and to devise a high level solution for it. Now, in this phase, focus will put on implementing the solution as software. This indicates coding or program must be written to implement the design.

5.2 Implementation Tools

Clearly there are many ways to implement a design, and many languages and tools are available. For implementing this educational software, development environment that involve are:

1. Microsoft Window 2000 Professional
   - Serves as development platform or operating system

2. Macromedia Director 8.0
   - Serves as development tools and platform for system and interface design.

3. Lingo Script
   - Language used to develop advanced application and functional features.

4. Sound Force 4.5
   - Serves as tools to develop sound effects and voice instruction.
5. Microsoft Art Gallery
   - Helps in developing interface and animated graphical
6. Adobe Photoshop
   - Tools for graphical design.
7. Swish 2.0
   - Help in developing multimedia application like animation effects.

5.3 Programming Standard and Procedures

Before begin to write code, there are standard ways to follow, so that the codes and associated documentation are easy to everyone who reads them.

In the phase of implement system design, there are two type of standard used to followed:

5.3.1 Developer's Standard

Standards and procedures help to organize thoughts and avoid mistakes. Standardize documentation helps in locating faults and in making changes, because it clarifies which section of program perform which function.

Standards and procedures also help in translating designs to code. Consequently, modifications to code that result from changing in hardware or interface specification are straightforward, and the possibility of error is minimized. This is very important in developing this system because the software may used at different platform.

5.3.2 Standards For Others

Once the code is complete, others likely to use it in a variety of ways. For example, for students that wish to do future enhancement on the system or similar project could do reference on the code.

Therefore, coding must followed standard for others also to make sure that maintenance programmer could easily find component that needed to change or refer. Automated tools are available that can analyze the code to determine which procedures are called by the component and which procedures invoke it.

5.4 Algorithms

Algorithms help to make sure system run smoothly as what mention in system design. This is due to coding normally performing whatever planned in algorithms. In this package, there are few algorithms that determine the flow of whole software.

5.4.1 Algorithms for First Chapter

The main ideas in this part are delivery courses and enable user to take note. There are few components need to run at same time, there are:

1. Robot as teaching agent moves
2. Content of chapter 1 deliver by using typing animation.
3. Flash animation at the banner on top of the interface.

Timing is one of the most important factors to consider in developing multimedia software. After these three components had run, users are allowed to take note at prepared space.

Below is the summary of algorithm for chapter 1.

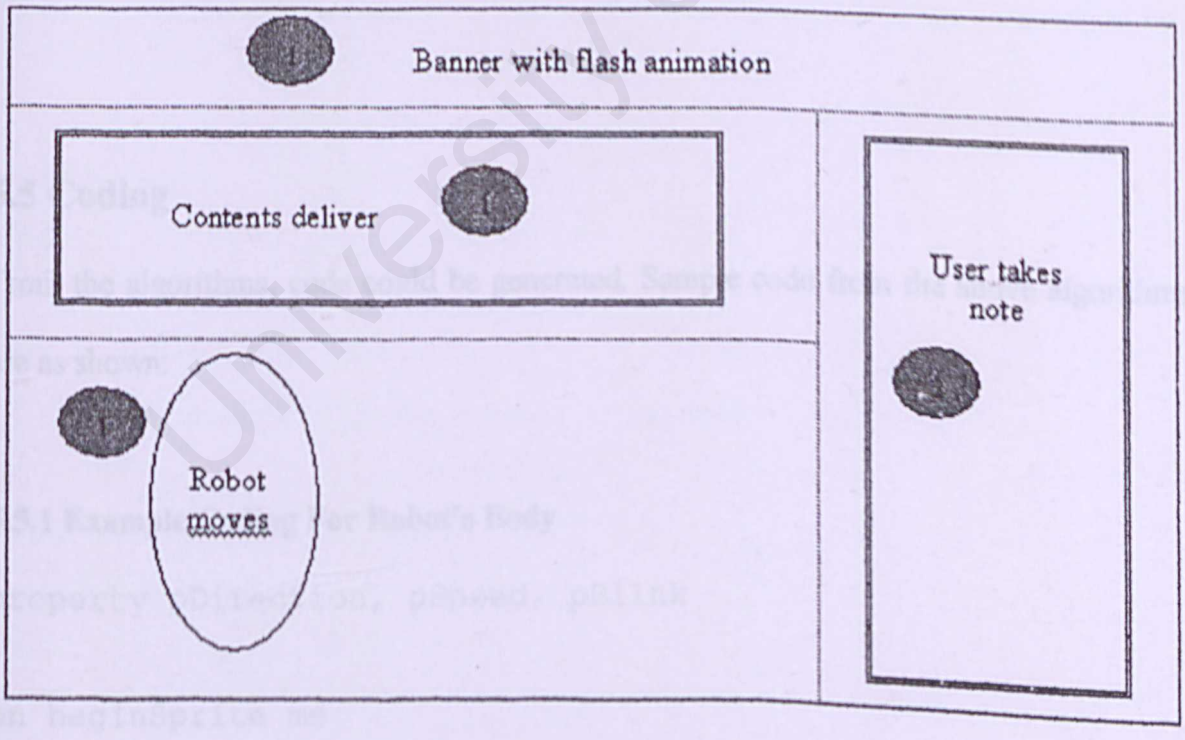


Figure 5.1 Summary of algorithm chapter 1

5.4.2 Algorithms For Tutorial

The aim in this part not only to let user had more understanding on the content, moreover to train users' speed. Therefore the algorithms sound this way:

1. Voice instruction given out to user
2. 1000 marks gave to user at first place
3. 1 marks will be deducted for every 1 second past
4. If answer correct
   4.1 Then mark down the marks left
   4.2 Else mark down the marks left + deduct another 500 marks for wrong answer
5. Total all marks
6. Exit section

5.5 Coding

From the algorithms, code could be generated. Sample code from the above algorithms are as shown:

5.5.1 Example Coding For Robot's Body

```
property pDirection, pSpeed, pBlink

on beginSprite me
  pDirection = 1
  pSpeed = 10
```

```
  pBlink = FALSE
end

on moveRobot me
  if pDirection = 0 then
    exit
  sprite(me.spriteNum).member = member("robot"&&pDirection)
  sprite(me.spriteNum).locH = sprite(me.spriteNum).locH +
    pDirection*pSpeed
  sendAllSprites(#robotPartMove,pDirection*pSpeed)
  if pDirection = 1 and sprite(me.spriteNum).locH > 420
  then
    pDirection = -1
  else if pDirection = -1 and sprite(me.spriteNum).locH <
          80 and soundBusy(1) then
            pDirection = 1
  else if pDirection = -1 and sprite(me.spriteNum).locH < -
          80 and not soundBusy(1) then
            pDirection = 0
  end if
end

on stopMoveRobot me
  sprite(me.spriteNum).member = member("robot")

  if pDirection = 0 then
    pDirection = 1
end

on blink me
  sprite(me.spriteNum).member = member("robot blink")
```

```
  pBlink = TRUE
end

on exitFrame me
  if pBlink then
    sprite(me.spriteNum).member = member("robot")
  pBlink = FALSE
  if random(50) = 1 then
    blink(me)
end
```

5.5.2 Example Coding For Robot's Arms

```
property pDestAngle, pCurAngle, pMinAngle, pMaxAngle,
pAngleChangeSpeed, pMapPoint, pMapToPrevSprite

on getPropertyDescriptionList me
  list = [:]
  addProp list, #pMinAngle, [#comment: "Min Angle:",
    #format: #integer, #default: -30]
  addProp list, #pMaxAngle, [#comment: "Max Angle:",
    #format: #integer, #default: 30]
  addProp list, #pAngleChangeSpeed, [#comment: "Angle
    Change Speed:", #format: #integer, #range: [#min: 1,
    #max: 10], #default: 3]
  addProp list, #pMapToPrevSprite, [#comment: "Map to Point
    Sprite?", #format: #boolean, #default: FALSE]
  addProp list, #pMapPoint, [#comment: "Map to Point on
    Previous Sprite:", #format: #point, #default:
    point(0,0)]
  return list
```

```
end

on beginSprite me
  -- set starting angle as in the Score
  pCurAngle = sprite(me.spriteNum).rotation

  -- select a new angle at random
  newDestAngle(me)
end

on newDestAngle me
  -- pick an angle between the max and min
  pDestAngle = random(pMaxAngle-pMinAngle)+pMinAngle
end

on talking me
  -- add or subtract angle to get to destination
  if pDestAngle < pCurAngle then
    pCurAngle = pCurAngle - pAngleChangeSpeed
    if pCurAngle <= pDestAngle then
      newDestAngle(me)
  else
    pCurAngle = pCurAngle + pAngleChangeSpeed
    if pCurAngle >= pDestAngle then
      newDestAngle(me)

  end if

  -- set the rotation of the sprite
  sprite(me.spriteNum).rotation = pCurAngle
```

```
  -- if needed, position sprite to point in previous sprite
     (joint)
  if pMapToPrevSprite then
    sprite(me.spriteNum).loc =
      mapMemberToStage(sprite(me.spriteNum-1), pMapPoint)
  end if
end

on robotPartMove me, x
  sprite(me.spriteNum).locH = sprite(me.spriteNum).locH + x
end
```

5.5.3 Example Coding For Typing Text

```
property pText, pTextMem, pChar, pDrawColor, pFinalColor,
  pPause

on getPropertyDescriptionList me
  list = [:]
  addProp list, #pTextMem, [#comment: "Member With Text",
    #format: #member, #default: VOID]
  addProp list, #pFinalColor, [#comment: "Final Color of
    Text", #format: #string, #default: "00AA00"]
  addProp list, #pDrawColor, [#comment: "Drawing Color of
    Text", #format: #string, #default: "66FF66"]
  return list
end

on beginSprite me
  pText = pTextMem.text
  pChar = 0
```

```
  sprite(me.spriteNum).member.text = ""
  pPause = 0
end

on unPause me
  pPause = pPause + 1
end

on exitFrame me
  if pPause = 0 then
    exit
  end if

  -- if at the end of the text, set whole field to final
     color
  -- then dispose of this behavior
  if pChar >= pText.length then
    sprite(me.spriteNum).member.color = rgb(pFinalColor)
    sprite(me.spriteNum).scriptInstanceList = []
    exit
  end if

  -- add two characters at a time, because text is too slow
     to draw on most computers
  repeat with i = 1 to 2
    -- next character
    pChar = pChar + 1
    newChar = pText.char[pChar]

    -- add character to member
    sprite(me.spriteNum).member.setContentsAfter(newChar)
```

```
    if newChar = "." and not ("0123456789" contains
      pText.char[pChar-1]) then
        pPause = pPause - 1
    end if
  end repeat

  -- set all text, except the new character, to the final
    color
  if pChar > 1 then
    sprite(me.spriteNum).member.char[1..pChar-1].color =
      rgb(pFinalColor)
  end if

  -- set new character to the draw color
  sprite(me.spriteNum).member.char[pChar].color =
    rgb(pDrawColor)
end
```

5.5.4 Example Coding For Tool Tips

```
property pState

on beginSprite me
  pState = sprite(me.spriteNum).member.name
end

on mouseEnter me
  member("State Name").text = pState
end
```

```
on mouseLeave me
  member("State Name").text = " "
end
```

5.5.5 Example Coding For Volume Slider

```
property pPressed, pBoundingSprite, pLeftLimit, pRightLimit

on getPropertyDescriptionList me
  list = [:]
  addprop list, #pBoundingSprite, [#default: 0,
    #format:#integer, #comment: "Bounding Sprite Number:"]
  return list
end

on getBehaviorDescription me
  return "Volume Slider Handler"
end

on beginSprite me
  limitRect = sprite(pBoundingSprite).rect
  pLeftLimit = sprite(pBoundingSprite).rect.left
  pRightLimit = sprite(pBoundingSprite).rect.right
end

on mouseDown me
  pPressed = TRUE
end

on mouseUp me
  pPressed = FALSE
```

```
end

on mouseUpOutSide me
  pPressed = FALSE
end

-- position slider according to mouse
on exitFrame me
  if pPressed then
    hPos = the mouseH
    if hPos = 0 then
      exit
    hPos = max(pLeftLimit,hPos)
    hPos = min(pRightLimit,hPos)
    sprite(me.spriteNum).locH = hPos
    percent = float(hPos-pLeftLimit) / float(pRightLimit -
      pLeftLimit)
    vol = integer(percent*255)
    sound(1).volume = vol -- set the sounds volume
    sound(2).volume = vol
  end if
end
```

5.5.6 Example Coding For Tutorial

```
global gQuestionNum, gPossiblePoints, gScore,
  gCorrectAnswer

on startGame
  gQuestionNum = 1
  gScore = 0
```

```
  showScore
  askQuestion
  go to frame "Play"
end

on askQuestion
  text = member("Data").text.line[gQuestionNum]

  the itemDelimiter = ";"
  question = text.item[1]
  answers = text.item[2]
  gCorrectAnswer = value(text.item[3])

  member("Question").text = question
  the itemDelimiter = ","
  repeat with i = 1 to 4
    member("Answer"&&i).text = answers.item[i]
  end repeat

  gPossiblePoints = 1000
  showPossiblePoints
end

on gameTimer
  gPossiblePoints = gPossiblePoints - 1
  showPossiblePoints
end

on showPossiblePoints
  member("Possible Points").text =
    "Points:"&&gPossiblePoints
```

```
end

on showScore
  member("Score").text = "Score:"&&gScore
end

on clickAnswer n
  if n = gCorrectAnswer then
    gScore = gScore + gPossiblePoints
    showScore
    nextQuestion
  else
    gPossiblePoints = gPossiblePoints - 500
    showPossiblePoints
  end if
end

on nextQuestion
  gQuestionNum = gQuestionNum + 1
  if gQuestionNum > member("Data").text.line.count then
    go to frame "done"
  else
    askQuestion
  end if
end
```

5.5.7 Example Coding For Radio Button

```
property pOnMember, pOffMember, pState, pGroupList

on getPropertyDescriptionList me
```

```
  list = [:]
  addProp list, #pOnMember, [#comment: "On Member",\
    #format: #member, #default: ""]
  addProp list, #pOffMember, [#comment: "Off Member",\
    #format: #member, #default: ""]
  addProp list, #pState, [#comment: "Initial State",\
    #format: #boolean, #default: FALSE]
  addProp list, #pGroupList, [#comment: "Group List",\
    #format: #list, #default: []]
  return list
end

on beginSprite me
  if pState then
    turnMeOn(me)
end

on mouseUp me
  turnMeOn(me)
end

on turnMeOn me
  pState = TRUE
  sprite(me.spriteNum).member = pOnMember
  repeat with i in pGroupList
    if i <> me.spriteNum then
      sendSprite(sprite i,#turnMeOff)
    end if
  end repeat
end
```

```
on turnMeOff me
  pState = FALSE
  sprite(me.spriteNum).member = pOffMember
end

on selected me
  repeat with i in pGroupList
    if sprite(i).pState = TRUE then
      return i
  end repeat
end
```

5.6 Interface

In phase of implementation, creating icon and common interface is important. The following show icon and some graphical items used in the software.

robot

button

take note

volume control

welcome

flash banner

Figure 5.2 Icon and common graphical items used

5.7 Documentation

Program documentation is considered to be the set of written descriptions that explain to a reader what the programs do and how they do it. Internal documentation is descriptive material written directly within the code all other documentation is external documentation.

5.7.1 Internal Documentation

The internal documentation contains information directed to someone who will be reading the source code of the program. Thus, summary information is provided to identify the program and describe its data structures, algorithms and control flow.

Usually, this information is placed at the beginning of each component in a set of comments called the header comment block. The following information must include in header comment block.

1. what component is called
2. who wrote the component
3. when the component was written and revised
4. why the components exists
5. how the component uses its data structures, algorithm and control

For example:

```
-- Young Choon Hoong 17/12/01
on mouseUp me
  turnMeOn(me)
end
```

Besides, other internal documentation that used in this package includes other program comments, meaningful variable names and statement labels and formatting to enhance understanding.

5.7.2 External Documentation

External documentation is intended to be read also by those who may never look at the actual code. External documentation is part of the overall system documentation. In this project's external documentation, information that include are: describing the problem, describing the algorithms and describing the data.

The main part of external documentation of this package is chapter 7 in this report, which is system evaluation, problems encounter and future enhancement.

Testing

CHAPTER 6

TESTING

6.1 Introduction

There are many types of testing technique. In order to judge the degree of success for this package, three types of testing technique had been carried out.

1. Unit Testing
2. Integration Testing
3. System Testing

6.2 Unit Testing

In unit testing, firstly, codes are being read through, trying to spot algorithms, data and syntax errors. Code review, is similar to the requirements and design reviews discussed in earlier chapters.

The example shows below was a unit testing on moving robot and button testing. The desired result was the robot will move accordingly to the contents and whenever mouse pointer comes near the button, comment on the button will automatically comes out. The result of the testing was satisfied.

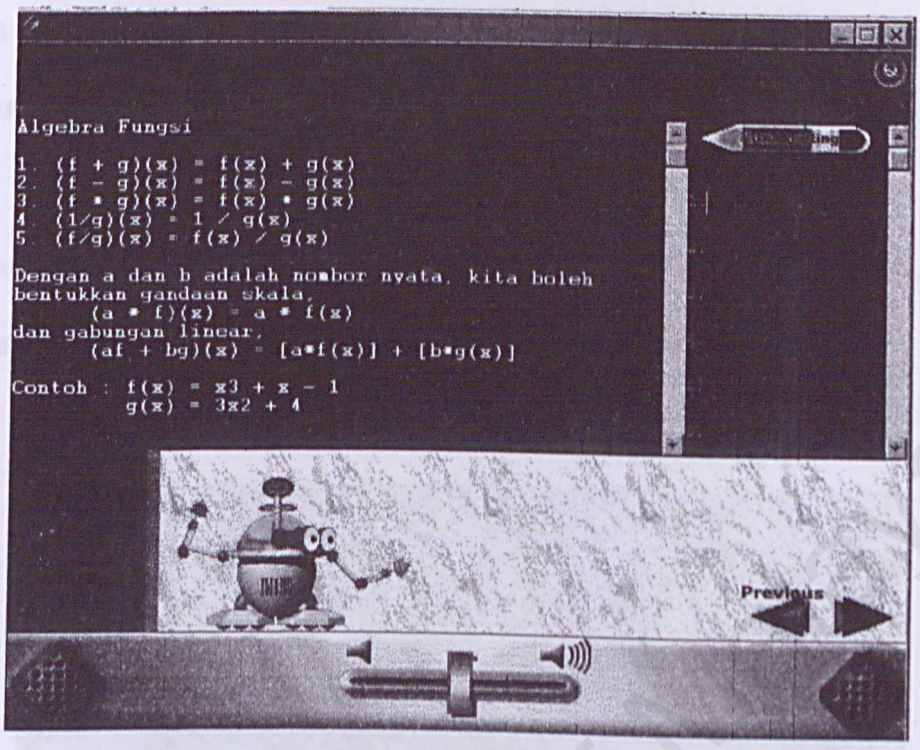


Figure 6.1 Example of unit testing

6.3 Integration Testing

When individual components are working correctly and meet objectives, they are now combining into a working system. This integration is planed and coordinated so that when a failure occurs, developer will have idea of what caused it. In addition, the order in which components are tested affects choice of test cases and tools.

The system is again viewed as a hierarchy of components, where each component belongs to a layer of the design. The integration testing used in this project was bottom-up integration. When this method is used, each component at the lowest level in the

system hierarchy is tested individually first, Then, the next components to be tested are those that call the previously tested ones. The approach is followed repeatedly until all components are included in the testing

Example of integration testing here is, when an user take note in the specific column, the note will automatically appear on another page; "Revisi Cepat".

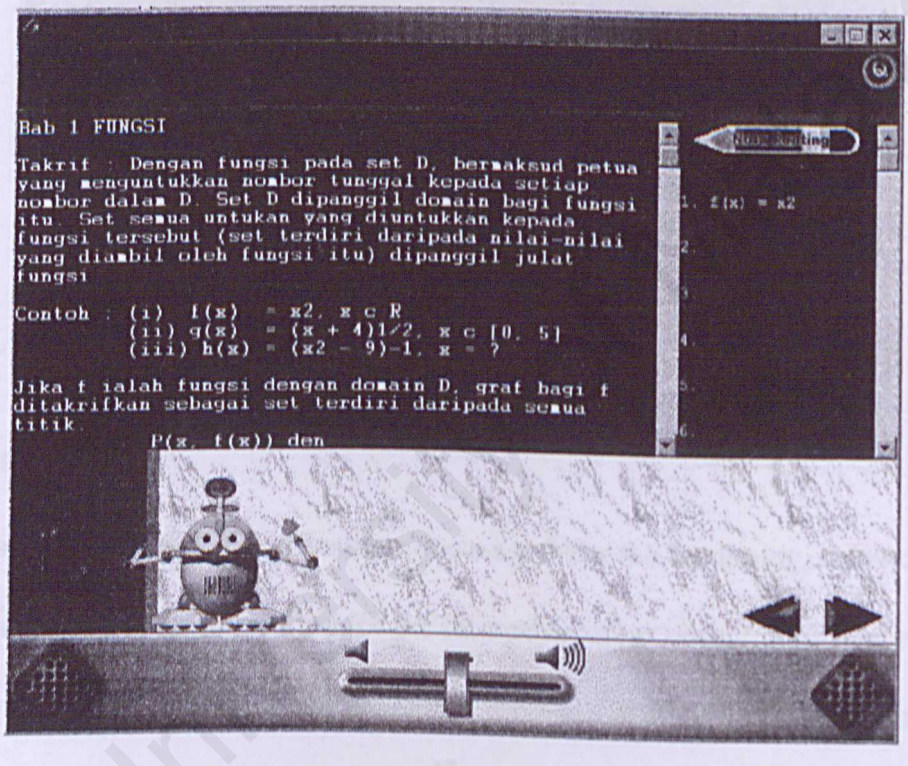


Figure 6.2 Example of integration testing

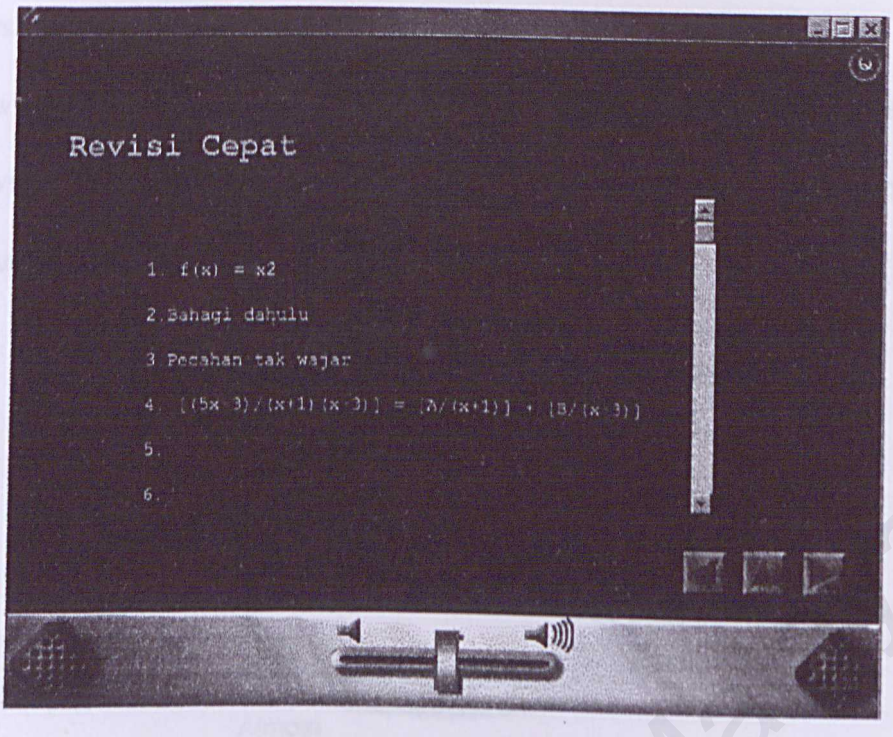


Figure 6.3 Example of integration testing (cont)

6.4 System Testing

As the entire system involved in testing processes from the beginning, upon completion of the system it only required testing for the desired result or output. It did not require any algorithms or syntax testing here. It would be very difficult to track through any algorithm or syntax errors at this stage.

Usually, for the system testing purpose both the developer and a few external reviewers are involved. In this project, the external reviewers – users from University Malaya had helped in system testing. The reviewers' feedback and comment was analysis and result are as followed.

6.4.1 Users' Testing Result

The following chart shows users respond on question how they feel similarity of this package with the original course. There are total up three users agree and one user disagree out of ten users.

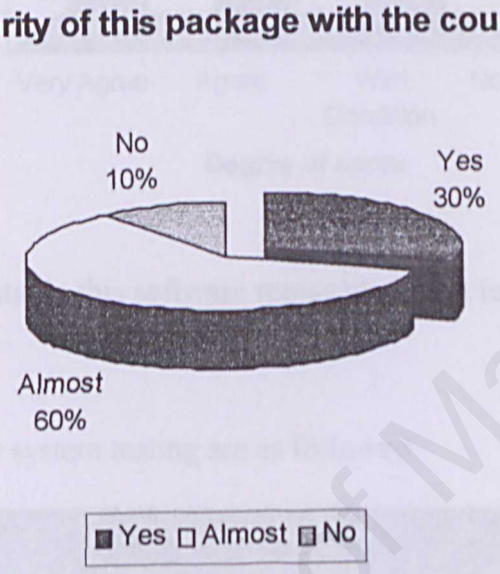


Figure 6.4 Pie Chart on System Testing Result

Below is analysis result on how suitable did users feel this package replace current teaching system. In conclusion, only one student very agrees with this statement and only one student disagree. Whereas students who agree and agree with condition had four in total respectively.

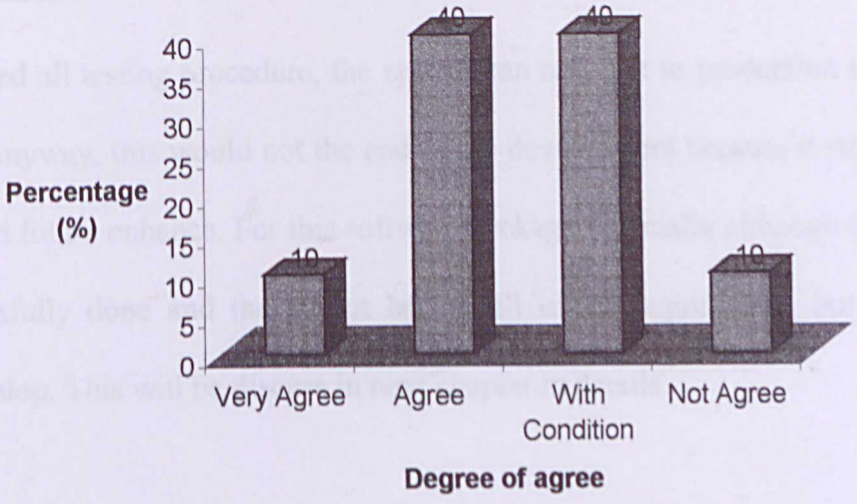


Figure 6.5 How suitable this software replaced current teaching material

Some function that used for system testing are as followed.

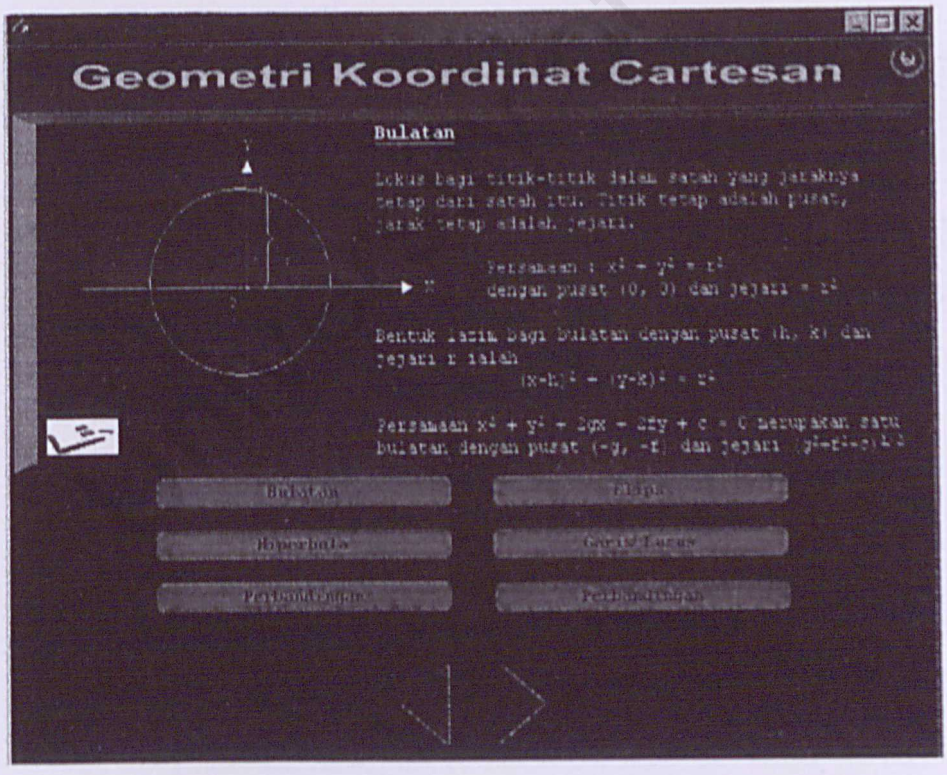


Figure 6.6 Interface used in system testing

6.5 Conclusion

After proceed all testing procedure, the system can now put to production after changes had done. Anyway, this would not the end of the development because it still needed for evaluate and future enhance. For this software package especially although all the testing was successfully done and the output had fulfill users' requirement, but it still need further develop. This will be discuss in next chapter in details.

Evaluation and Conclusion

CHAPTER 7

EVALUATION AND CONCLUSION

7.1 Introduction

System evaluation is done to judge the degree of how this system fulfill users' requirement. Besides, evaluation process also aims to clarify system strength and weakness, list out problem encounter and suggestion for future enhancement. Thus evaluation process was done on the "Matematik Asas" software package.

These areas will be delved into further details in the following sections.

7.2 System Strengths

7.2.1 keep the teaching material with latest technologies

This software was developed by using combination of tools Macromedia Director 8.0, Lingo Script, Swish 2.0 and others latest technology. Thus, it kept teaching material with latest technology.

7.2.2 User Friendliness

This package employs Graphical User Interface (GUI) to ensure the interface is attractive and easy to used. Thus the users will have more understanding on their studies. Besides, this will help the student undergraduate to have more interest in studying this course.

7.2.3 Voice Reading

In order to keep new technology in teaching material a lot of new tools and application included in the software. This may cause the software gave user impression that it is not formal. In order to make the software look formal, voice reading had add in to the package.

Besides, voice reading also used for purposed to give instruction for user. As result, users need not to read instruction before using the any application or function. This idea was created to fulfill users' requirement from previous interview.

7.2.4 Provide Easy Tools

Another strength of the package is providing easy tools for user. In order to enable user taking notes, the package prepared tools in helping users to do fast revision and taking notes.

Moreover, easy tool also available when deliver content of the course. For example, in chapter 2, after viewing the graph that prepared, users may need to understand the different between graphs. Therefore, this package prepared such function to help user identify differential between these graphs.

7.2.5 Idea of Examination Scope

By involving in test and tutorial section, users would have idea on real examination scope and types of questions.

7.2.6 Educational Package in BM

Many educational packages in market nowadays were using English as delivery language. Anyway this is not suitable to use for subject in University Malaya, because the content of the course are normally in Malay. Thus, by using Bahasa Malaysia as delivery language is a strength of this software.

7.3 System Limitations

7.3.1 Support Single User Environment

This system could only support single user environment. That means it is not suitable for multi user to access this application at a time.

7.3.2 Store additional notes

This package could allow users store their notes. Any way this function had its own limitation, that is the space for addition notes is limited. When the space finished, users could not store any more notes.

7.3.3 Not Keeping User's Record

Another limitation for this software is this package was not able to kept users' previous record or performance. Although this considers an important point in developing educational language, the software did not support this.

7.3.4 Not Include All Topics

This project had only included two chapters of the "Matematik Asas" but not all chapters. Moreover, not all details are included in the two chapters. In other words, in chapter 1 and chapter 2, not all contents included and questions also not completely added in.

7.4 Problem Encountered and Solution

7.4.1 Size of File

In general, developing an educational software package needed to include as much graphical items and multimedia effects. Anyway, this may cause problem o developer who develops at own place or without CD writer.

Although most of the graphic, icon and pictures are store in JPEG format, the size of the file was still too big to save in few diskettes.

The only solution is getting a CD writer.

7.4.2 Coding Problem

New development language and unfamiliar development software need time to learn. Learning need time, moreover, to fully make use of its features need more time. Thus, this process spends some time during development period. In overall effect, it slows down development process.

In addition, inflexible tools in Lingo Script and Macromedia Director cause problems during development. Example of problem faced is Macromedia Director 8.0 only allowed undo once. This burden development phase cause developer has to check every step to make sure only one step cause error.

The solution was do referred to materials such as books, articles, internet and always get advice from expert and supervisor. Besides, some web site offer suggestion and idea for developer such as http://www.experts-exchange.com , http://macromedia.com .

7.4.3 Target User

As earlier defined, this educational package was developed for user in university. Different from students from kindergarten or primary school, student undergraduate in university had more mature thinking and different taste.

The package should not full of colorful pictures and funny sound effects. Moreover, more application need to add in to the package as user may required more functions than color interface.

In order to solve this problem, research and interview target user had to be done. Moreover, get suggestion from supervisor also helps to undergo this problem.

7.4.4 Complexity of Content

There are total up seven chapters in real "Matematik Asas" syllabus. In this package, developer only includes two chapters. This is due to the complexity of the syllabus. Most of the content needs writing whereas calculation actually takes small parts. Thus, is not suitable to conduct tutorial and test by using multiple-choice questions.

The solution of this problem was get advice from lecturer in-charge of this course.

7.5 Future Enhancement

A useful system would not complete develop for lifetime. Below are some suggestions from users and developer to further enhance the package.

7.5.1 Cover More Features

This package could include more advanced features like error detection features. The feature should be able to detect error that cause by user. In other words, if user had wrong selection or not following instructions, system should able to detect and further more tells the user to proceed in actual manner.

Besides, enable to answer students' questions. This could be done in the way such as help file in other application where user can search content to look for information that desired.

7.5.2 Record on Students' Performance

Previously, this functionality was considered very important and useful for users. Thus, in future enhancement, this function should put higher priority. This function should include saving student's previous record and manipulate it to make it useful for student like statistics.

Comparison also could be done for students. In other words, higher marks students in test and tutorial could put their name at top performance list.

7.5.3 Enhancement on Tutorial and Test

In current software, only multiple-choice questions included in tutorial and test. In future enhancement, tutorial and test should involve more variety of questions. Especially questions that need idea of students and proven of theory.

This may look more familiar to students and these type of questions more likely same with the real examinations.

7.5.4 Voice Delivery Course

In the package, voice and sound only serve as background support and giving instructions. In future, content of the course should deliver by using voice.

7.5.5 High-light Pen

In this software, developer had added in tools to enable user taking notes for fast revision. Anyway, this could be enhancing to way like highlight pen. When user use the tools and passed through the notes, the notes should be high lighted with color. Moreover, user could select different color.

7.6 Conclusion

The "Matematik Asas" educational software package was developed to fulfill requirement as teaching material for "Matematik Asas". The product should be able to deliver the course in attractive and efficient ways that involve multimedia application.

This package has completed its mission in enhancing the current educational system to computerized system. Many new and advanced features had added in this package to implement new concept in delivery lecture and apply on studying.

Bibliography

1. Heinich, Robert (1993), Instructional Media and the New Tecnologies of Instruction, Macimilian Publishing Co., New York.

2. Paul G. Geisert and Mynga K. Futrell (1995), Teachers, Computers and Curriculum: Microcomputer in the Classroom.

3. Rupe, Vickie S (1986), A Study of Computer-assisted Instruction, its uses, Effects, Advantages and Limitations. – Journal

4. Fred A. Teague, Douglas W. Rogers & Rogre N. Tipling (1994), Technology and Media Instructional Application, Kendall/Hunt Publishing.

5. Use of Copmputer in Teaching and Learning (1974).

6. Pressman, R.S. (1992), Software Engineering: A Practitioner's Approach, McGraw Hill.

7. Tay Vaughan (1996), Multimedia: Make It Work, 3rd Ed. McGraw Hill.

8. Shari Lawrence Pfleeger (1998), Software Engineering: Theory and Practice, Prentice Hall International.

9. Jeeffery Hay (1998), Edutainment – Seminar by Maria Clawe, Professor at the University of British Colombia.

10. http://www.cs.yovku.ca/courses95.96/4361/edutainment.htm#Edutainment

11. Kenneth E. Kendall and Julie E. Kendall (1995), System Analysis and Design, 3rd Ed., Prentice Hall Inc.

12. Eileen Tan, On Designing Multimedia Learning system and Instructional Strategies.

13. Price, Robert V. (1991), Computer-aided Instructions: a Guide for Authors, Pacific Grave California: Brooks.

14. Using Multimedia in Education, (1993). http://fps.uwaterloo.ca/projects/eng04.html#inter

15. Gousalya Siva (1998), Multimedia with Director.

16. Suzprowic, Bohdan O. (1994), Multimedia Technology: Combining Sound, Text, Computing Graphics and Video, 2nd Ed. Charleston: Computer Technology Research Corporation.

17. Levin, Carol (1993), Multimedia Tool for Teaching, PC Magazine.

18. Jack A. Chamber and Jeery W Sprecher (1983), Compter-Assisted Instruction: Its use in the Classroom, Prentice Hall Inc.

19. Barker, P. (1989). Multi-media computer assisted learning.

20. Ballantyne, R. & Packer, J. (1995). Making Connections Using Student Journals as a Teaching/Learning Aid. Gold Guide No. 2.

21. Brown, S. & Knight, P. (1994). Assessing Learners in Higher Education.

22. Chalmers, D. & Fuller, R. (1995). Teaching for Learning at University (Theory and Practice).

23. Renee M. Newman, M.S., (1998) Effective and Responsible Teaching. http://www.dyscalculia.org/Edu505.html

QUESTIONNAIRE

Name : _____

Age : _____

Gender: Male / Female

Faculty: _____

1. Do you surf internet or access internet?
   - ☐ Yes
   - ☐ No

2. How much time do you spend on computer every week?
   - ☐ Less than 2 hour
   - ☐ Between 2 to 4 hours
   - ☐ Between 4 to 6 hours
   - ☐ More than 6 hours

3. Have you ever expose to educational software in the market?
   - ☐ Yes
   - ☐ No

4. What are your commend about the existing educational software?
   (You may choose more than one option)
   - ☐ Good
   - ☐ No commend
   - ☐ Not interesting at all

5. Which type of educational software you prefer?
   - ☐ on-line educational packages
   - ☐ buying the original software

6. What is your opinion regarding the teaching style of mathematics in school?

- ☐ Very good
- ☐ Good
- ☐ Average
- ☐ Below average
- ☐ Terrible

7. Do you interested to learn mathematics via computer software?

- ☐ Yes
- ☐ No

8. Which language do you prefer for higher-level mathematics educational software?

- ☐ Malay
- ☐ English
- ☐ Mandarin
- ☐ Tamil
- ☐ Other: _____

9. Which type of educational software do you think is more suitable for higher mathematics?

- ☐ Games base
- ☐ Formal way with more explanations and exercises
- ☐ Interactive base (more on question and answer section)
- ☐ More graphic less word
- ☐ Other: _____

10. What kind of learning do you think is more suitable for higher mathematics?

- ☐ Attitude learning (indirect e.g. games)
- ☐ Motor skill (movement e.g. mouse click)

- ☐ Cognitive strategies (problem solving)
- ☐ Verbal information (state, explain, tutorial)
- ☐ Intellectual skills (identify, demonstrate problem and solving)

11. What skill do you emphasize on educational software?
(You may choose more than one option)

- ☐ Listening
- ☐ Reading
- ☐ Speaking
- ☐ Writing

12. Suggest some features or section to higher-level mathematics software.

_____

_____

13. Do you think that using computer as method of teaching will be more interesting compare to manpower? Why?

_____

_____

User Testing Feedback

Name : _____

Age : _____

Gender: Male / Female

Faculty: _____

1. Did you attend course "Matematik Asas" before?
   - ☐ Yes
   - ☐ No

2. Have you ever expose to educational software from web site?
   - ☐ Yes
   - ☐ No

Question on testing system

3. Are the instruction given easy to understand?
   - ☐ Yes
   - ☐ No

4. You can understand the contents of the software. How far do you agree this statement?
   - ☐ Very agree
   - ☐ Agree
   - ☐ Fair
   - ☐ Need further explanation
   - ☐ Not agree

5. What is your opinion on the interface?

- ☐ Nice
- ☐ Fair
- ☐ Childish

6. In the tutorial, how many points did you earn? What is your comment on the tutorial?

- ☐ Less than 2000
- ☐ 2000 - 3000
- ☐ 3000 - 4000
- ☐ Above 4000
- ☐ Comment:

_____

7. Do you agree this software is similar to the subject "Matematik Asas" in original courses?

- ☐ Yes
- ☐ Almost
- ☐ No

8. What is your comment on the system functionality especially function to take notes and voice instruction?

- ☐ Very good
- ☐ Good
- ☐ Fair
- ☐ Not good

Suggestion:

_____

9. What are your commend about the this educational software?

- ☐ Good
- ☐ Fair
- ☐ Not interesting at all

10. Do you think this software can be used as teaching material replacing current teaching style in university?

- ☐ Very agree
- ☐ Agree
- ☐ With certain condition
- ☐ Not agree

13. Do you any good idea in order to enhance this software?

_____

_____

User Manual

Introduction

User manual is important and useful for various users to use the system and in this case using this software. For this educational package, this user manual consists of guidelines for all types of users. By end of this user manual, user will able to fully use the software.

Hardware and Software Required

- Microsoft Window 98 and above;
- Pentium II 400 MHz processor;
- 256 MB RAM;
- Hard disk with more than 15MB free space

Getting Started

As user startup the software by installing the CD into system, user could see interface as shown, then user could click on the "GO" button to proceed to overview.

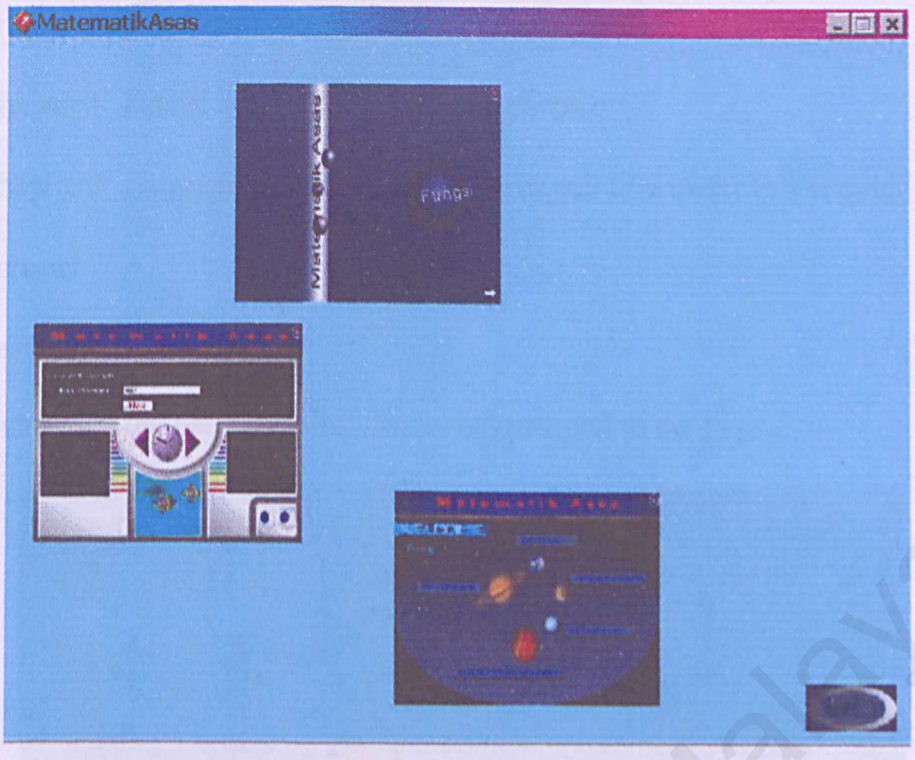


Figure 1 Introduction page

After this user could log in to the system by entering name.

Figure 2 Log in interface

From overview, user had few options. User could proceed to:

- Chapter 1: Fungsi or sub title under this chapter
- Chapter 2: Geometri Koordinat Cartesan or sub title under this chapter
- Tutorial
- Test

By just clicking on the option desire, user could proceed to the page.

Figure 3 Overview

For first selection, if user had chosen to start from chapter 1, voice instruction will be given. A robot as teaching agent will moves together with contents in chapter 1. After the content finish typed out, user can insert notes in the space that prepared. User could type in the notes or use method copy and paste from the notes provided.

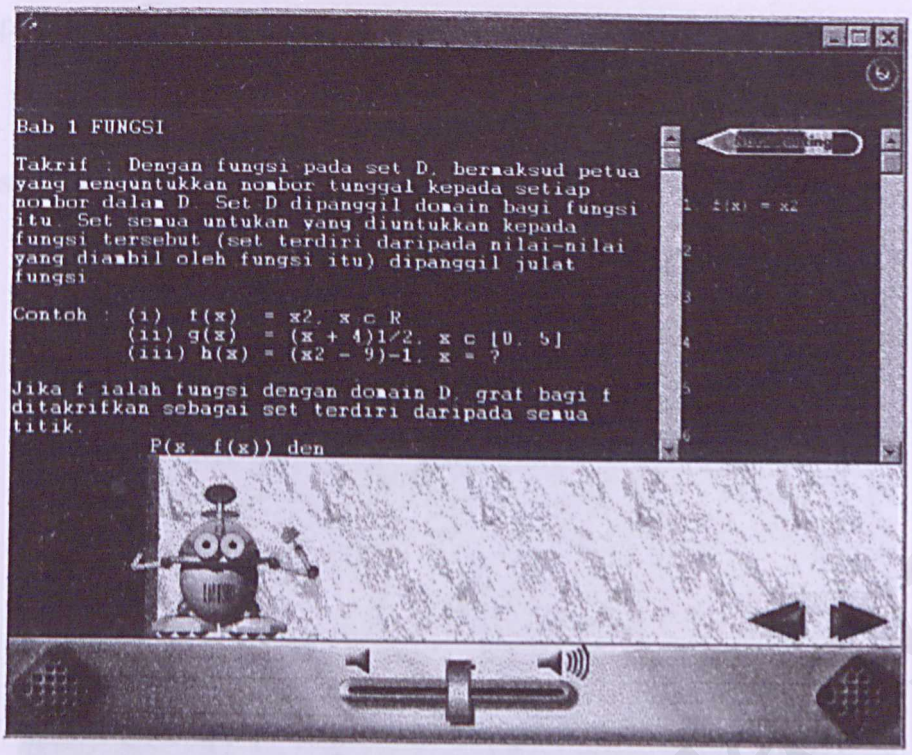


Figure 4 Chapter 1: Fungsi

The notes will save in another interface so that user could have fast revision during examination period.

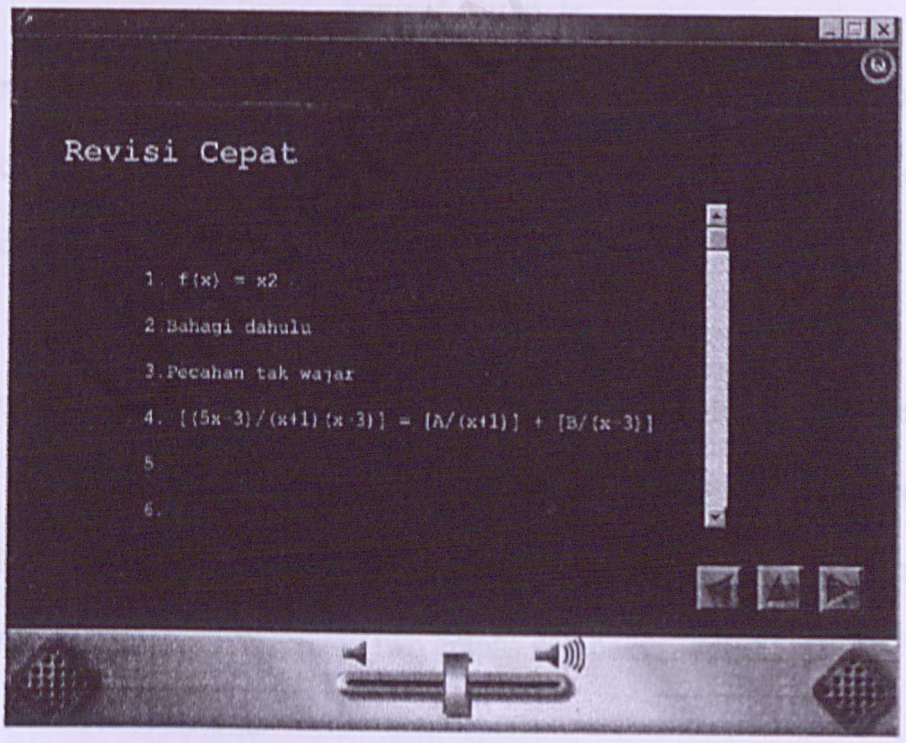


Figure5 Express Revision

From chapter 1, user also could proceed to tutorial. In this section, voice instruction will be given. User just needs to follow the proper way to finish the tutorial. This tutorial could consider a game based tutorial and is full of fun.

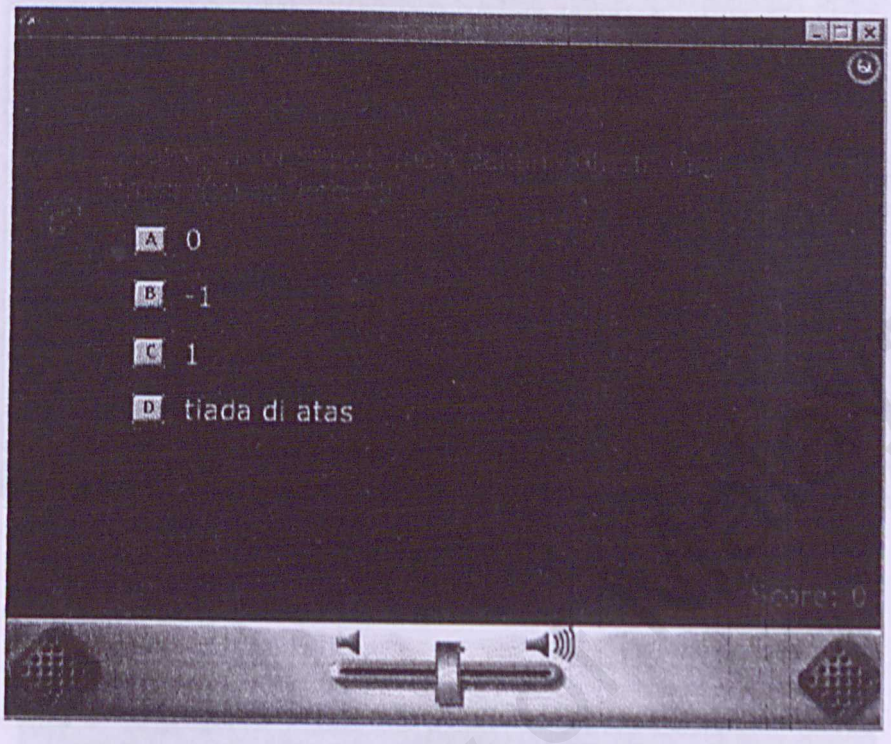


Figure 6 Tutorials

For chapter 2: Geometri Koordinat Cartesan, this chapter will focus on graphs. User could select graph that need to learn and the sample graph will appear in the middle of the interface.

Further more, user could click on the button next to the graph for further explanation. User may feel confuse on so many different types of graphs or user may need to understand the different between graphs. In order to compare graphs, user could select comparison button. Different between two graphs will be explained.

The following interface shows all these:

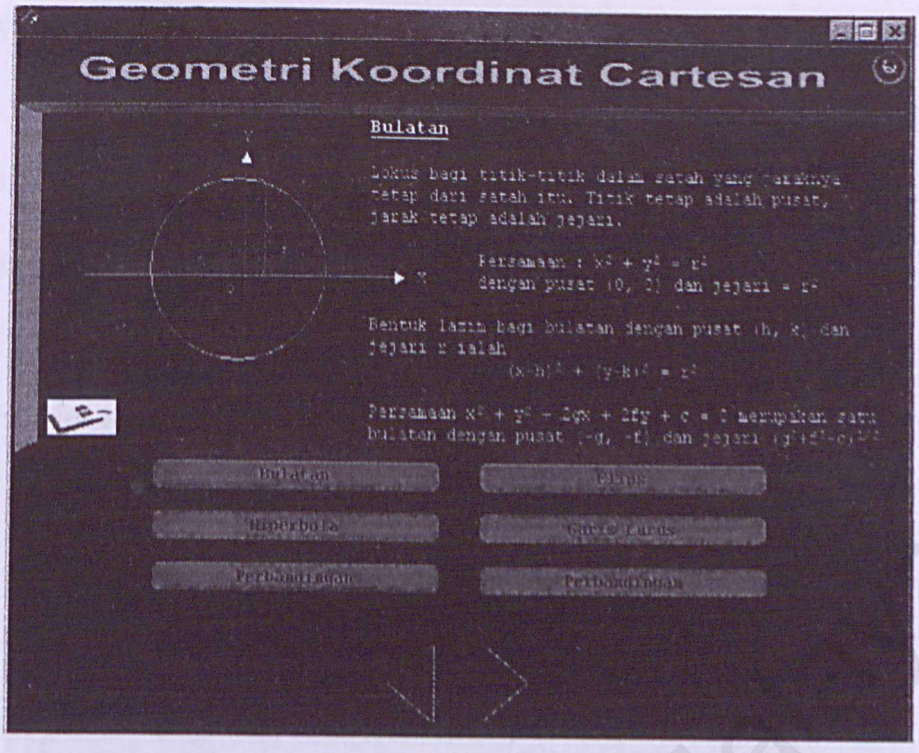


Figure 7 Chapter 2: Geometri Koordinat Cartesan

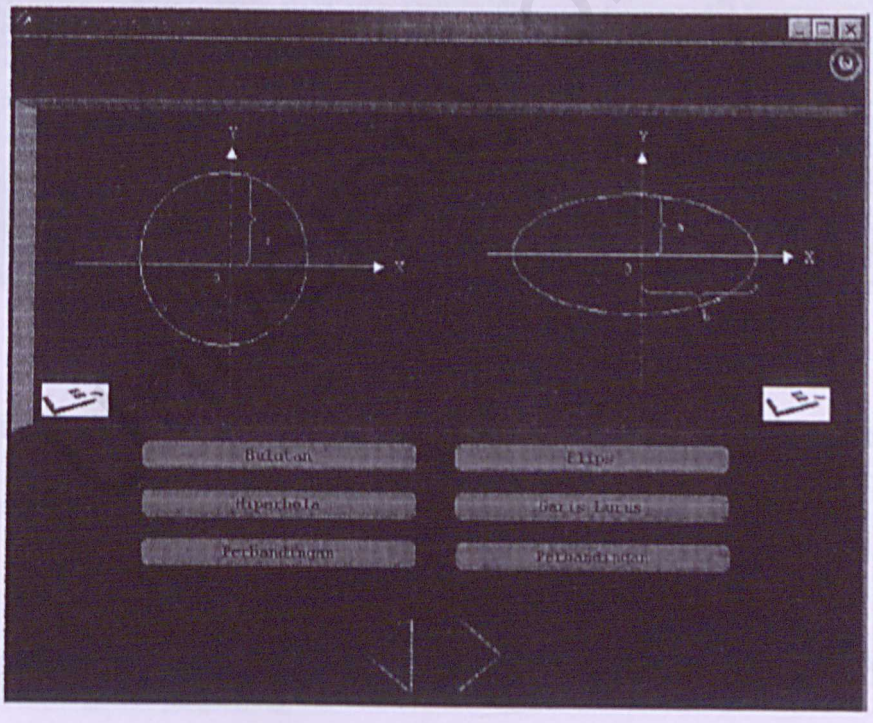


Figure 8 Comparison on graphs

Finally, when user tries to answer questions from test section, user had multiple-choice questions waiting to answer. Sometimes, user may need to change previous answer after reading question at the back. Thus, the software prepared these facilities to let user back to previous question to do changes.

The interface for test looks this way.

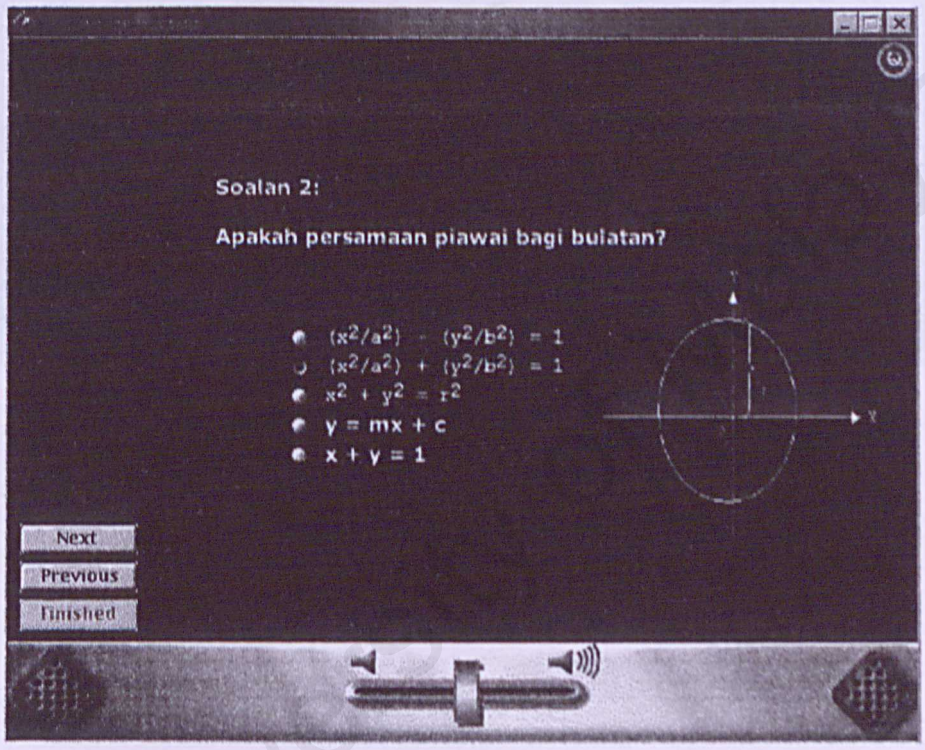


Figure 9 Test

From the interface user could select buttons "Next", "Previous" and "Finished", this is to let user determine whenever felt satisfied with answer only quit this section.

After proceeding all application or interface, user could exit the system and the following interface will show.

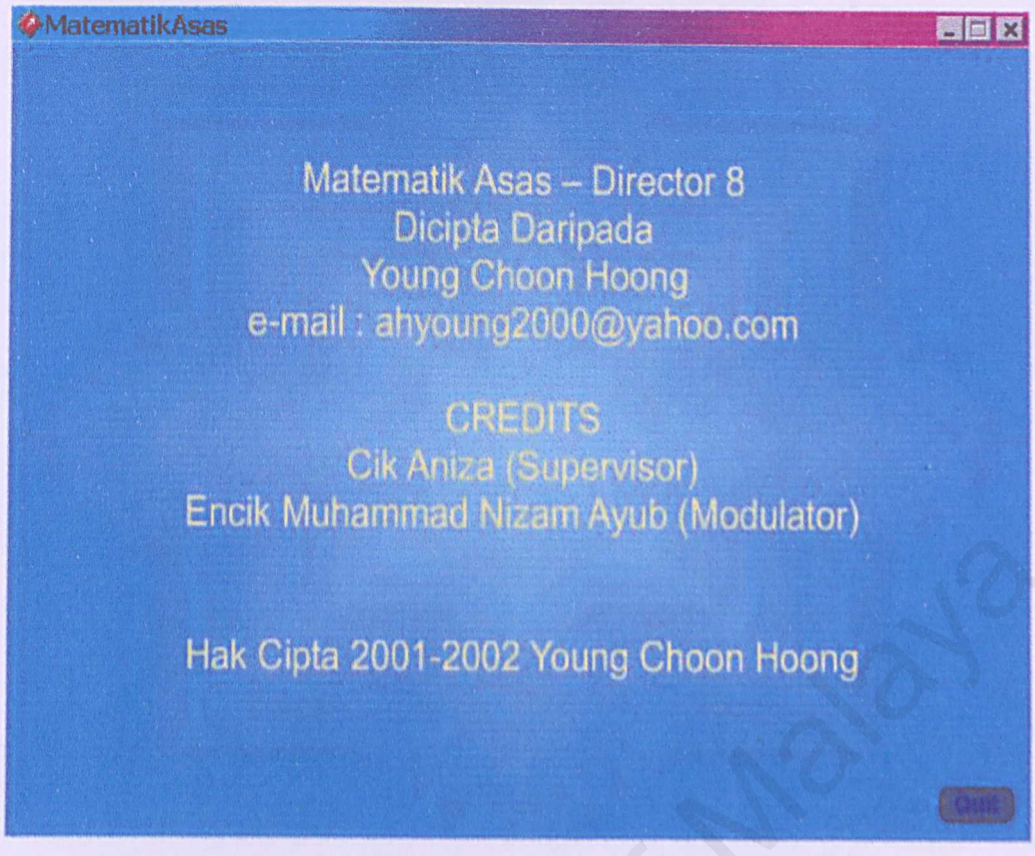


Figure 10 The ending interface for this software