



Thames
Polytechnic

Incorporating Avery Hill College

School of Computing and Information Technology

M E C H S .

A CAL Program.

M E C H S .

A CAL Program.

By : S J Jabbar.

Class : FDYC4.

ACKNOWLEDGEMENTS

I would like to thank the following for their assistance with this project:

Liz Bacon for her project supervision.

Frank Martin for some valuable references.

Thames Polytechnic Computer Services Centre
for the facilities and services provided.

School of Information Technology Micro Lab Technicians
for the facilities and services.

Thames Polytechnic Library.

Birmingham City Public Lending Library.

Damien, Roger and Ron
for their invaluable help during testing.

John and Sue for their support.

Pauline for her support.

John for his support.

ACKNOWLEDGEMENTS.

I would like to thank the following for their assistance
with this project:

Liz Bacon : for her project supervision.

Frank Martin : for some valuable references.

Thames Polytechnic Computer Services Centre
: for the facilities and services provided.

School of Information Technology Micro lab Technicians
: for the facilities and services.

Thames Polytechnic Library.
Birmingham City Public Lending Library.
Dennis, Roger and Ann
for their invaluable help during testing.

SUMMARY.

CONTENTS.

The use of computers in teaching is almost thirty years old.

ACKNOWLEDGEMENTS

If this present generation was not educated by them then the next will certainly be educated through Computer Aided Learning.

CONTENTS

What exactly CAL is and its development up to the present is briefly discussed in the introduction. The variety of CALs in use, and how infact they are used follows this.

1.2 DEVELOPMENT OF CAL.

After the introduction, the objectives of this project as stated at its conception are laid out. The the next section reports on the implementation details. This includes hardware and software used and the way the software was developed.

1.4.3 Simulations.

Testing of MECHS, the name of the program, was an on going process during development and the results were used to tailor the program to suite user needs. The nature of these tests and the major changes initiated by the test results are outlined in the report.

2. OBJECTIVES.

The conclusions to be drawn, and the recommendations are the final parts of the report.

3.2 SOFTWARE USED.

3.3 SOFTWARE DEVELOPMENT.

3.3.1 Functional specification.

3.3.2 Data Flow Diagrams.

CONTENTS.	3.3.3	Structure charts.	13
4.	TESTING.		17
	ACKNOWLEDGEMENTS		Page.
5.	CONCLUSION.		18
	SUMMARY		
	RECOMMENDATION.		18
	CONTENTS		
	REFERENCES		19
1.	INTRODUCTION		2
	1.1	WHAT IS CAL ?	2
	1.2	DEVELOPMENT OF CAL.	3
	OPENINGS.		3
	1.3	USING A CAL.	4
	1.4	VARIETIES OF CAL.	5
	1.4.1	Drill and test.	5
	1.4.2	Educational games.	5
	1.4.3	Simulations.	6
	1.4.5	Multi-choice tests.	6
	1.5	CAL FOR PHYSICS.	7
	1.6	PASCAL AS A MEDIUM FOR CAL.	7
2.	OBJECTIVES.		9
3.	IMPLEMENTATION DETAILS.		10
	3.1	HARDWARE.	10
	3.2	SOFTWARE USED.	10
	3.3	SOFTWARE DEVELOPMENT.	10
	3.3.1	Functional specification.	11
	3.3.2	Data Flow Diagrams.	11

3.3.3	Structure charts.	11
4.	TESTING.	12
5.	CONCLUSION.	16
	RECOMMENDATION.	18
	REFERENCES	19
	BIBLIOGRAPHY	20
	APPENDIX.	21
1.	USER GUIDE. THE REPORT.	22
2.	EXAMPLE RUN SESSION.	26
3.	FLOW CHARTS.	27
	3.1 Data Flow Diagrams.	28
	3.2 Structure Charts.	34
4.	LISTINGS.	36

T H E R E P O R T .

1. INTRODUCTION.

1.1 WHAT IS CAL ?

CAL is an acronym, short for Computer Assisted (or Aided) Learning.¹ A similar term CAI or Computer Assisted Instruction is also used but, for the most part, only in the United States. Other related terms are CBL, CBE and CBT - (Computer Based Learning/Education and Training). For the purposes of this report the term CAL shall be used.

CAL is based on the formal method of teaching. This approach was first discovered by Socrates, but not for another two thousand years were the possibilities of this method investigated further. In the 1950s the researches of B. F. Skinner at the Harvard university yielded methodical teaching again in the form of Programmed Learning (PL).

Although books were published, methods were devised and even machines built, based on the work of Skinner, by the mid '60s these had largely disappeared. PL had not died, however, it had undergone a metamorphosis and became computerised, it was now CAL.

using computers. Much of the development of CAL began in IBM's garage.

1.2 DEVELOPMENT OF CAL.

In 1954, Carl Beck and Howard Thomas developed the first CAL program, called Project. Specialized CAL computers were

found. In 1958, Rath and Anderson, Two IBM personnel wrote the first CAL program. Using an IBM 650 mainframe, the program taught binary arithmetic. It interacted with the student through the console typewriter. Don Bitzer of the University of Illinois made the next important contribution by using what is now called an authoring language. This is a language that allows the CAL to fill in screens of information (known as frames) and not have to be a computer expert. However, advances with the launch of the IBM PC. In 1982, a new era in CAL program development emerged.

The next major developments were produced by two private companies as a result of work sponsored by a grant from the US government. Control Data Corporation (CDC) produced a system called PLATO (Programmed Logic for Automatic Teaching Operations) and Mitrone Corporation (MC) produced one called TICCIT (Timeshared, Interactive Computer-Controlled Information Television). PLATO used plasma screens which are transparent and allow colour slides to be superimposed upon computer generated graphics. TICCIT used normal television sets. PLATO had an authoring language called TUTOR which allowed the teacher to write material. MC decided to employ a team of writers instead.

During this time IBM were developing their own CAL systems.

The earlier ones were far too expensive to be developed beyond the research stage but other CAL systems and authoring languages

were spawned such as Coursewriter-III and IIS. Other manufacturers who worked on CAL were Bell and Howell, Texas Instruments and Hewlett-Packard. Specialist CAL companies were founded such as Computer Curriculum Corporation (CCC).

1.4 DEVELOPMENT OF CAL

These developments have for the most part been on large mainframes and minis, which meant that they were out of the reach of most schools and colleges never mind an individuals. But since the early 1980s, with the development of the home computer market in the late '70s, with the widespread use of home computers such as Commodore 64, Sinclair ZX Spectrum, Amstrad CPC, BBC Micro, Apple, Tandy, Commodore etc, CAL systems have been implemented on these, much more inexpensive, machines. With the launch of the IBM PC, in 1982, a new era in CAL programs development emerged. Hundreds, perhaps even thousands of CAL systems exist now for drill and test. Some of the more noteworthy systems are MICROTEXT, PASS, MCSD, MUMEDALA and CYCLOPS. This means that CAL is now within reach of most people with home-computers, and these days most are well within the reach of average households. ~~normally they are not recommended for individual use.~~

1.3 USING A CAL PROGRAM

These sorts of programs are designed for young children and have been designed to be used with a television screen or a VDU (visual display unit). The usual way by which the student communicates with the CAL program is via a normal typewriter style keyboard. The student is presented with educational material in a structured way and the computer allows him to respond individually, actively, and at his own pace.

Finally the computer is instructed to give accurate and immediate feedback to the student assigned as a game.

1.4.3 Simulations.

1.4 VARIETIES OF CAL.

There are two sorts of simulations in a short-running computer programme within a housed CAL or one can be. There are a great variety of educational programs which are not easily classifiable as drill and test programs, or simulations and games. These rather loosely described as CAL. Among these are drill and test programs, educational games, simulations, multi-choice tests (with or without accompanying text) and others which can not easily be placed in any particular category.

...peripherals or control an industrial process.

1.4.1 Drill and test.

...In such tape programs, events are condensed. This sort of program is suited for infants and juniors, so that is for groups of children to have fun together in drills of various sorts. But generally they are not recommended for individual use.

1.4.2 Educational games.

These sorts of programs are designed for young children and have similarities to arcade games. Games programs can be used in two ways :

...Firstly, full they can be used as rewarding interludes between periods of more formal learning.

1.5 CAL FOR PHYSICS Secondly, the whole educational program can be designed as a game.

Programmes in business subjects for encoding into a CAL see section 1.4.3 Simulations. have not one answer only. Most selected subjects are naturally suited for CAL.

There are two sorts of simulations - a short-running simulation, eg. the animated example within a normal CAL or they can be an insertion/written as complete simulation packages. The latter ones of course can be run in real-time or fast time. e.g. *MANUFACTURING* microcomputer. The programs were written in VEDO pascal with the aid of the BASIC subroutines. In real-time you can learn how to fly an aeroplane or control an industrial process.

1.6 PASCAL AS A MED In fast time programs, events are condensed into short time periods for the learner, eg. *THE CHANCELLOR* you can be Chancellor of the Exchequer and run CAL. Instead trying to control balance of payment deficits during a few minor ones over a long period of years. CAL programs as number of factors have to be taken into consideration when

about 1.4.5 Multi-choice tests. etc. These include a program availability, related score, utilization rate, ease of learning, productivity. This project is an example of a CAL in this category.

It presents text after which the pupil is given a set of **BASIC** questions with a choice of answers. Multi-choice programs are useful for revision purposes. and again availability. Its inherent unstructured nature made it preferable for this program.

1.5 CAL FOR PHYSICS.

It is not difficult to find a language for a simple program and the C language is well and freely used for many programs although Physics is a suitable subject for encoding into a CAL as any questions set can usually have but one answer only. Most science subjects are similarly suited for CAL.

On the Open University course S271, together with the author, we have chosen UCSD pascal as the language for the project.

On the Open University course 'S271 - Discovering Physics' an important element of the summer school is the use of programs. One of these is a CAL program and runs on a TERAK 8510:a microcomputer. The programs were written in UCSD pascal with the aid of the OASIS authoring system.

The author has been working on microcomputers since 1983.

He has used several languages in his work:

- PASCAL is the language of implementation for this project.
- 1.6 PASCAL AS A MEDIUM FOR CAL.**

As we have seen, due to the emphasis on structured programming, PASCAL is a good language for CAL. It is also reliable and has the advantage of availability and the authors familiarity with the language. There is no one particular language that is ideally suited for CAL. Indeed nearly every major language in use today, and quite a few minor ones, have been used to write CAL programs. A number of factors have to be taken into consideration when choosing a language to work with. These include : price, availability, related tools, utilization rate, ease of learning, productivity and special features.

- BASIC is the first choice for many who are involved in CAL but that is as a result of its low cost and wide availability. Its inherent unstructured nature made it unsuitable for this project.

- C would be a better language for a serious program and the C kernel is quite small and fairly easy to master. Although there are extensive libraries in C that make CAL possible, the lack of standardised libraries makes portability to other access to a unix machine is vital. The lack of access at the start of this project was, together with the authors' lack of experience in C ruled this language out.

Those who have used the various high-level languages

including C will appreciate that student would be able to use

- NATAL is the only high-level portable language specially designed for CAL. Unfortunately it is difficult to obtain a commercial version of this language despite micro-NATAL having been around for microcomputers since 1983.

followed by questions on the last subject. Should the student

fail to answer a question correctly he or she can frame of text

- PASCAL is the language of implementation for this project. will be presented, especially the reasons for this choice. This is due to its emphasis on structured programming, its availability and the authors familiarity with the language. and a comparison on the overall performance of the students

will be made with the other languages.

The program will be tested during its development by

students and the results of these tests will be used to refine

the program accordingly. The students will be both 'physically

literates' as well as 'soft-physicists' users.

2. OBJECTIVES.

This project aims to produce a Multi-choice CAL program with text that serves as a revision aid for classical mechanics. It is based upon a current O-level syllabus but should be used by those who have been on a physics course upto O-level standard. However an intelligent non-physics student should be able to use it aswell. how to be made up with the students.

The program consists of six 'chapters' each with a tutorial followed by questions on the text presented. Should the student fail to answer a question correctly a reinforcement frame of text will be presented, explaining the correct answer. At the end of a session the program will give marks for the chapters attempted and also a mark on the overall performance of the student. Program development was Sheffield based.

The program will be tested during its development by students and the results of these tests will be used to tailor the program accordingly. The students will be both 'physics literate' as well as non-physics users.

The software was developed using structured techniques.

3. IMPLEMENTATION DETAILS.

3.1.1 Functional specification.

3.1 HARDWARE.

To design a physics revision aid the program that will do the following:

The program MECHS was written on the PRIME 750 minicomputer. The terminal used in developing the program (and consequently the one that it has to be run on) was the GT100.

- (a) test the student on the test presented.
- (b) receive and analyse student responses.
- (c) provide reinforcement material if required.

3.2 SOFTWARE USED.

The subject matter of the CBI program used in MECHS was written using the Sheffield screen editor SED under the CP/M operating system. The language for program development was Sheffield Pascal.

3.3.1 Data Flow Diagrams.

SEE APPENDIX 3.1

3.3 SOFTWARE DEVELOPMENT.

3.3.3 Structure Charts.

The software was developed using structured techniques.

SEE APPENDIX 3.2

~~3.3.1 Functional specification.~~

3.3.1 Functional specification. ~~It will be the responsibility of the student to decide how best to go about this.~~

Produce a physics revision aid CAL program that will do the following: carried out after the program has loaded in skeletal form (without page numbering or text justification) and without (i) a) present text via frames,
b) test the student on the text presented,
c) receive and analyse student responses,
~~text analysis~~ d) provide reinforcement material if required.

Some computing science student is to be found in each class. The subject matter of this CAL program should be suitable for students who are taking physics at O level and classical mechanics and should be aimed mostly at students with ordinary level physics knowledge.

~~3.3.2 Data Flow Diagrams.~~ ~~more and not entirely relevant to~~

~~the project~~ after discussion with the teacher.

SEE APPENDIX 3.1

The data flow diagram was developed and was updated to include the changes that have been made since it was first drawn.

~~3.3.3 Structure Charts.~~

~~more and not entirely relevant to~~

SEE APPENDIX 3.2

4. TESTING. ~~Surprisingly~~ I found, immediately after having had a go at the program and the message about pressing any key to start it.

The best method for testing a CAL is to give it to a potential user and see how he/she gets on with it.

- The test was conducted so as not to start a major bug.

The tests were carried out after the program was written in skeletal form i.e without page numbering or text justification and without the full range of chapters.

TEST No. 9; (by ROBIN - first year engineering student.)

TEST No. 11; (by DENNIS - 2nd year computing science student).

His observations were:

Being a computing science student he found no difficulty in getting to the system and running MECHS. His observations were:

- Seeing that all page numbering of text was omitted.

"The introduction is too short and not entirely relevant to what happens after." - some lines are also omitted in the introduction and three parts are for the page number.

- The introduction was lengthened and was updated to include information about the changes that had been made since it was first written.

"It's confusing to read, especially when text is split across two pages and the message about pressing any key is in between." It is difficult not to get a rough idea of what had been intended, from the initiated message. For example, one - The text was rewritten so as not to start a major subsection on the bottom of a screen.

There should be some diagrams along with the text.
This you incorporated in your recommendations.
TEST No. 2; (by ROGER - a first year engineering student.)

Again there were no major difficulties in getting to MECHS. The questions are a bit wordy and some of the multiple His observations were :
"The answers are a bit silly."

- "It would have been nice if the pages were numbered and the questions that proved too wordy were shortened and the screen was split up at the top and bottom."
- Some ridiculous answers were replaced by more probable ones.
- Coding that allowed numbering of both the frames of text and the question frames was added. Using the GT100s alternative character set, a routine was also included to split the screen off into three parts, one for the page number, one for screen commands, the text and one for the command line at the bottom.
 - Coding to put the title on the screen was added. The title appears on the same line as the page number.

TEST No. 3; (by ANN - a first year biology student.)

Instead of the chapters being called chapter 1, chapter 2, etc., the Login proved to be difficult but after a rough user guide was given, the student was able to type in the title of the chapter she had been supplied, MECHS was initiated easily. Her observations were:

"There should be some diagrams along with the text." This was impossible to do. See RECOMMENDATIONS. There were no problems with these.

"The questions are a bit wordy and some of the multiple choice answers are a bit silly."

- The questions that proved too wordy were shortened and the silly multiple choice answers were replaced by more probable ones.

"The title of the chapter that's being done should be on the screen somewhere."

- Coding to put the title on the screen was added. The title appears on the same line as the page number.

9-00 "The diagnostics part could be more meaningful."

- Instead of the chapters being called chapter 1, chapter 2 etc, The routine to produce the title of the chapter was used to make the diagnostics section more meaningful.

Tested as a memory test. The results of PEGO were very good. This exercise highlighted the insincerity of a testing program's claimed reliability. It may well offend thought if

The tests highlighted other problems but these were mostly minor and trivial from the user's point of view. In detail. The various range, type and size errors were

encountered by all the test subjects so these were noted and the program can now detect and deal with these.

It will be interesting to see how far another pascal programmer goes with this.

There maybe a case for some form of data compression on the test files.

Pascal has been a good medium for writing a CAL, allowing a working version of the program to have been produced at an early stage in the development. Because of pascal's structured nature and through the use of structured development techniques work on MCCHS proved to be wholly compatible with the prototype.

Ordinary level physics in general, and classical mechanics in particular, is a good subject for CAL. An understanding of the subject matter to a depth greater than needed for O level was required. However, the insight into 're-learning' thus gathered, as well as the process by which this understanding was obtained, were an invaluable asset in developing MCCHS.

5. CONCLUSION.

As a revision aid MECHS has proved to be a good system. Students who have not studied physics for a long time say that it can help them remember the subject. This is probably because it served as a 'memory jerker'. The results of MECHS use by non physics students, though not representative, showed that to physics students highlighted its inappropriateness as a teaching aid for classical mechanics. But nearly all of these thought it served as a useful introduction to the subject.

The original version of MECHS was more than adequately suited, in terms of the facilities provided by the other terminal. This is not so for future development of the program while allowing it to be legible for another pascal programmer. There is no file example. One could imagine some compression of the text files or mechanics may be covered in separate files.

MECHS has met its design objectives and it is apparent that Pascal has been a good medium for writing a CAL, allowing a user centred computer design. A working version of the program to have been produced at an early stage in the development. Because of pascals structured nature and through the use of structured development techniques work, on MECHS, proved to be wholly compatible with the prototype.

Ordinary level Physics in general, and classical mechanics in particular, is a good subject for CAL. An understanding of the subject matter to a depth greater than needed for O level was required. However, the insight into 're-learning' thus gained, as well as the process by which this understanding was acquired, were an invaluable asset in developing MECHS.

ACKNOWLEDGEMENT

Without the feedback that users provided, during development, the MECHS program would not have been able to meet the design objectives. This proved to be an essential part in the development process. The final product met with approval by most of the users, and would give in the probability that all but most of the users, They were satisfied that it was an aid to revision.

The implementation of MECHS on the PRIME was more than adequate allied, as it was, with the facilities provided by the GT100 terminal. There is scope for future development, on the subject of mechanics, in the direction of IBM PC for example. Or by adding more chapters so that a greater area of mechanics may be covered.

using an Authoring language and more chapters, many more as a tool and to a much larger, and more complex, CAL system.

MECHS has met its design objectives and it is consistant with current CAL systems design.

RECOMMENDATIONS.

The main recommendation is that MECHS would be well suited to running on the IBM PC. This would allow the adding on of a graphics capability and would give it the portability that a CAL system should aim for.²⁵⁴ The outcome of running and the end of running time and computational demands.

Currently the MECHS program is linked exclusively to the classical mechanics, the subject of implementation.²⁵⁵ If the titles, chapter-headings and menu parts of MECHS were made subject independent then that would be a step in the direction of a CAL Revision Shell. Where the subject matter would be written using an authoring language and MECHS could then, easily serve as a front end to a much larger, and more complete, CAL system.²⁵⁶

Based learning (at Leeds University).

REFERENCES.

1. A. F. ABBOTT (1978). Ordinary Level Physics (3rd Edition), (Heinemann Educational Books).
2. B. F. SKINNER (1954). The science of learning and the art of teaching, (Harvard Educational Review).
3. B. F. SKINNER (1965). Waldon Two, (Macmillan London) Press.
4. E. J. PERKINS (1979). Test Yourself Physics, (Celtic Revision Aids). Witten Compilers Digital Press.
5. J. R. HARTLEY (1978). Computer Assisted Learning, (Computer Based learning Unit Leeds University).
6. J. R. HARTLEY, J. TAYLOR & Computer Assisted Learning in the United Kingdom, (Cambridge Education Press Ltd).
7. T. O'NEIL, M. DUFF - Learning and Teaching with Computers, (Harvester Press).

BIBLIOGRAPHY.

1. R. C. ATKINSON, H. A. WILSON - Computer Assisted Instruction. (Academic Press).
2. P. BARKER, H. YEATES - Introducing Computer Assisted Learning. (Prentice/Hall International).
3. G. BEACH - Computer Based Learning. (Sigma Technical Press).
4. A. BORK - Learning With Computers. (Digital Press).

THE APPENDIX.

5. D. GODFREY, S. STERLING - The Elements of CAL. (Reston Publishing Company Incorporated).
6. R. HOOPER, I. TOYE (ed) - Computer Assisted Learning in the United Kingdom. (Councils & Education Press Ltd.).
7. T. O'SHEA, J. SELF - Learning and Teaching with Computers. (Harvester Press).

APPENDIX 1 - THE MELCHI PROGRAM

1.1. USER GUIDE.

1.1.1. PURPOSE OF THE PROGRAM MELCHI.

This is a Computer-Aided Learning program called MELCHI. It is meant to be used as a revision aid for an ordinary level physics course on some aspects of classical mechanics. MELCHI will present you with some text, after which you will be asked to answer some questions. The questions will take the form of multiple choice type with correct answers allowing progression to the next question. **THE APPENDIX** puts the program into presenting some additional text before progression. This pattern is repeated, forming the chapters that you decide to attempt. At the end of the session you will be given a break down of the marks you earned at each section.

1.1.2. DESCRIPTIONS OF MELCHI'S FACILITIES.

1.1.2.1. Introduction.

The introduction is followed by a brief description of the facilities available. Following this is a brief history of the program.

The facilities available are:

APPENDIX I. MECHS - A COMPUTER AIDED LEARNING PROGRAM

1. USER GUIDE. This provides a list of your options at the start stage, these are explained below.

1.1 PURPOSE OF THE PROGRAM MECHS.

The program covers Kinematics, Newton's Laws, Work & Energy, Momentum, Circular Motion, Harmonic Motion, Gravity, Friction, and Heat.

This is a Computer Aided Learning program called MECHS. It is meant to be used as a revision aid for an ordinary level physics course on some aspects of classical mechanics. MECHS will present you with some text, after which you will be asked to answer some questions. The questions will take the form of multiple choice type with correct answers allowing progression to the next question. An incorrect answer prompts the program into presenting some reinforcement text before progression. This pattern is repeated for all the chapters that you decide to attempt. At the end of the session you will be given a breakdown of the marks you scored at each section.

1.2 DESCRIPTIONS OF MECHS'S FACILITIES. If this option is chosen then the program will produce diagnosticary. It will indicate whether a particular chapter was attempted by displaying the chapter name along with the mark scored on the questions. If a chapter was not attempted then this is indicated aussi. Finally, an overall mark is produced before MECHS ends.

1.2.1 Introduction.

If requested a few introductory frames of text will be displayed. Outlining the objectives and function of the program.

1.2.2 Menu.QMRED.

The menu provides a list of your options at that particular stage, these are : as written in Sheffield parlance.
The test files require a minimum of 10 pages of store. A typical session will reinforce Gravity, Weight & Friction, of run time to complete. 2.1 Speed, Velocity & Acceleration, that MECHS will run on, or the 3. the Newtons Laws of Motion.
4. Work, Energy & Power.
5. Machines.

1.3 TO P6.1 MEDensity and Relative Density.

(Each of these chapters follows the pattern outlined if you have an account at the Reims then so run the program earlier on in this guide.)
you must :

7. not End This Session, or do not do through a GETGO.

8. Then type SCS STPZM1010 PROJECT.MECHS.BIN
(If this option is chosen then the program will produce some diagnostics. It will indicate whether a particular and do the following:
chapter was attempted by displaying the chapter name along with the mark scored on the questions. If a chapter was not attempted then this is indicated as well. Finally, an overall mark is produced before MECHS ends.

and last for the message LOGIN OKED when it done

type LOGIN.JSIDY you will then be asked for the Password : you should type COPPER /the password will not appear on screen/. You will now have logged in.
As soon as the OK, prompt appears you can begin the session by typing SCS PROJECT.MECHS .

1.3 RESOURCES REQUIRED.

The MECHS program runs on a PRIME 750 minicomputer and requires 8 pages of store. It was written in Sheffield pascal. The text files require a minimum of 15 pages of store. A typical session will need between twenty and forty minutes of run time to complete. The GT100 terminal is the only one that MECHS will run on at the time of writing.

1.4 TO RUN MECHS.

If you have an account on the Prime then to run the program you must :

First LOGIN to your UFD (N.B must be through a GT100).

Then type SEG <TPB2>JS1CY4>PROJECT>MECHS.SEG

If you have not got an account then find a GT100 terminal and do the following:

Switch the terminal on,

press the SHIFT and BREAK keys together and wait for
FACILITY REQUIRED ? to appear. You should type C01
and wait for the message [CONNECTED]. When it does
type LOGIN JS1CY4 you will then be asked for the
Password : you should type COPPED (the password will
not appear on screen). You will now have logged in.

As soon as the OK, prompt appears you may begin the
session by typing SEG PROJECT>MECHS .

Example When you have finished the session type LOGOUT to leave the UFD.

Example run sessions have, unfortunately, not been able to be included. This is as a result of there not being any easy way of getting what appears on the screen of the BT100 on to paper.

However, if you want to see a run session without actually running MECIG then this is possible. First login to the UFD as described in the USER GUIDE, then instead of running MECIG using the SEQ command, type PRINT EXAMPLE.

Logout as before.

EXAMPLE RUN SESSION

Example run sessions have, unfortunately, not been able to be included. This is as a result of there not being any easy way of getting what appears on the screen of the GT100 on to paper.

However, if you want to see a run session without actually running MECHS then this is possible. First login to the UFD as described in the USER GUIDE, then instead of running MECHS using the SEG command, type PRINT EXAMPLE.

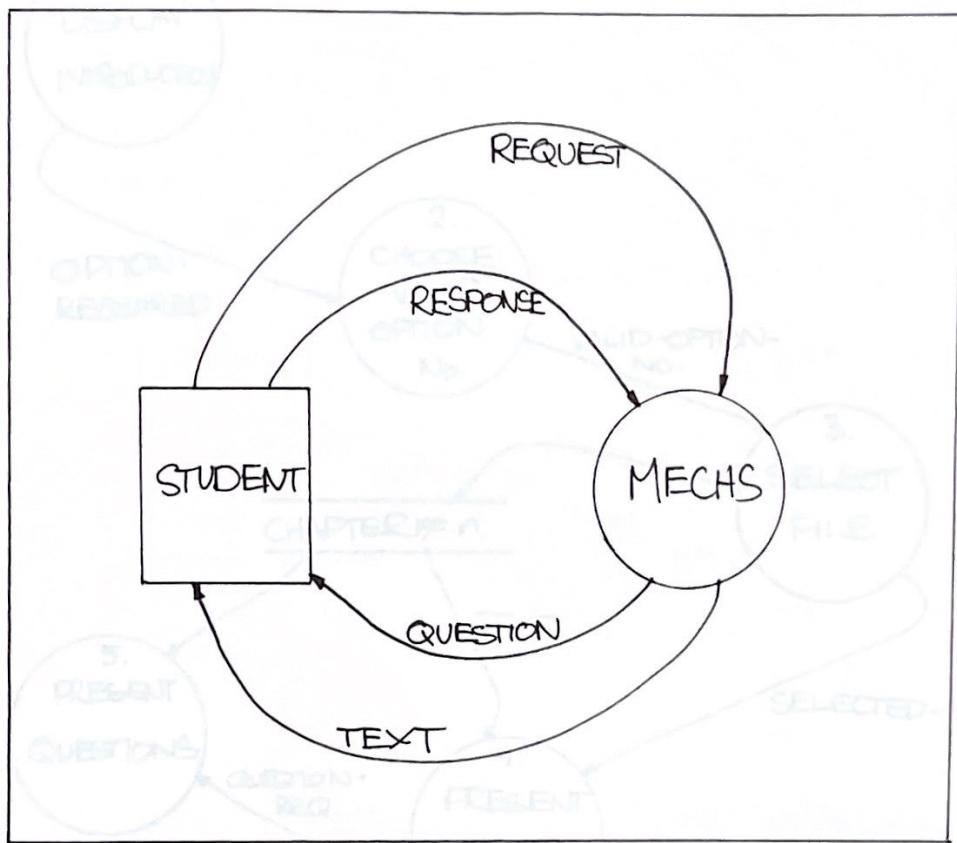
Logout as before.

THE FLOWCHARTS.



CONTEXT
DIAGRAM.

DIAGRAM B



CONTEXT
DIAGRAM.

DIAGRAM Ø

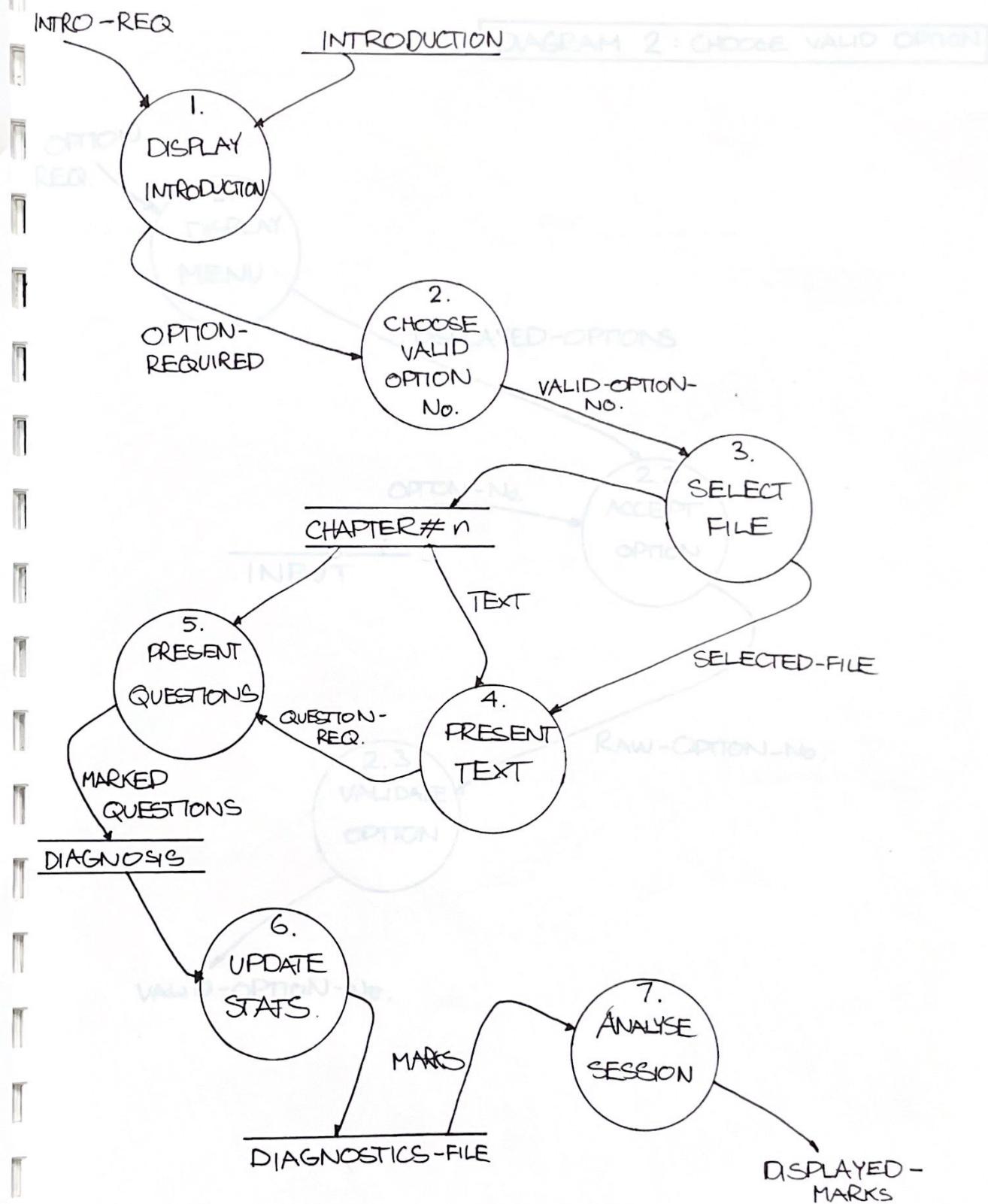


DIAGRAM 2 : CHOOSE VALID OPTION

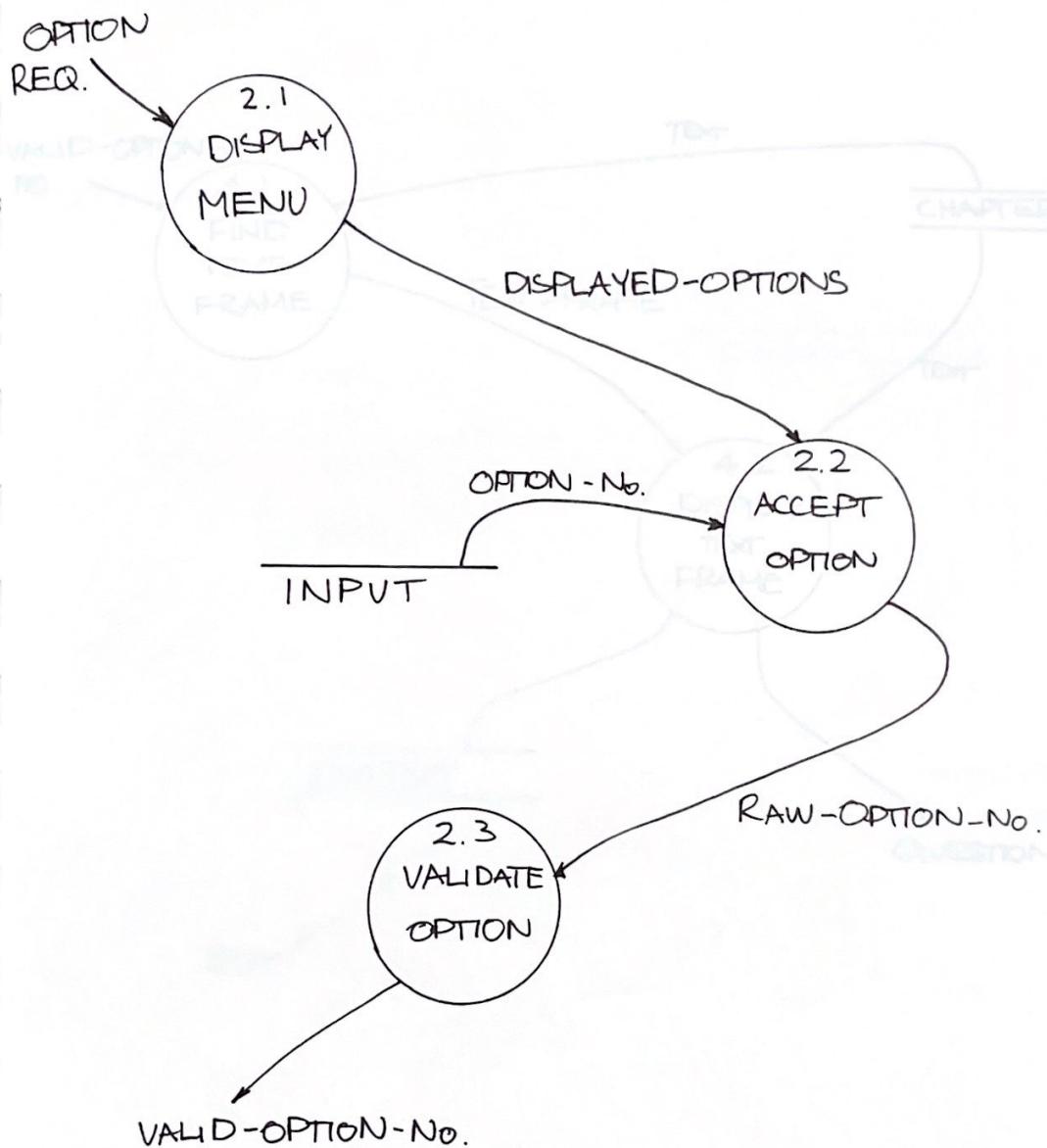


DIAGRAM 4 : DISPLAY-TEXT

DIAGRAM 7 : ANALYSIS SESSION

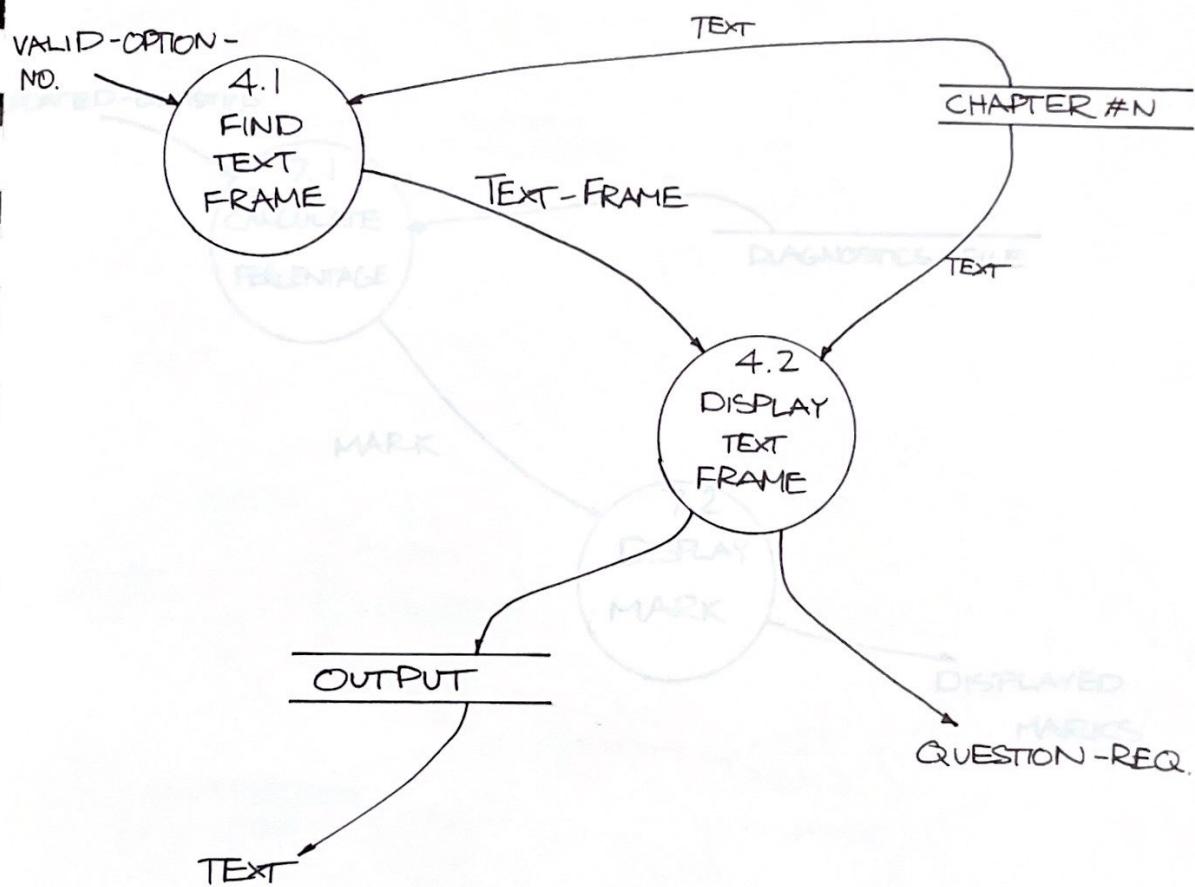


DIAGRAM 7 : ANALYSE SESSION

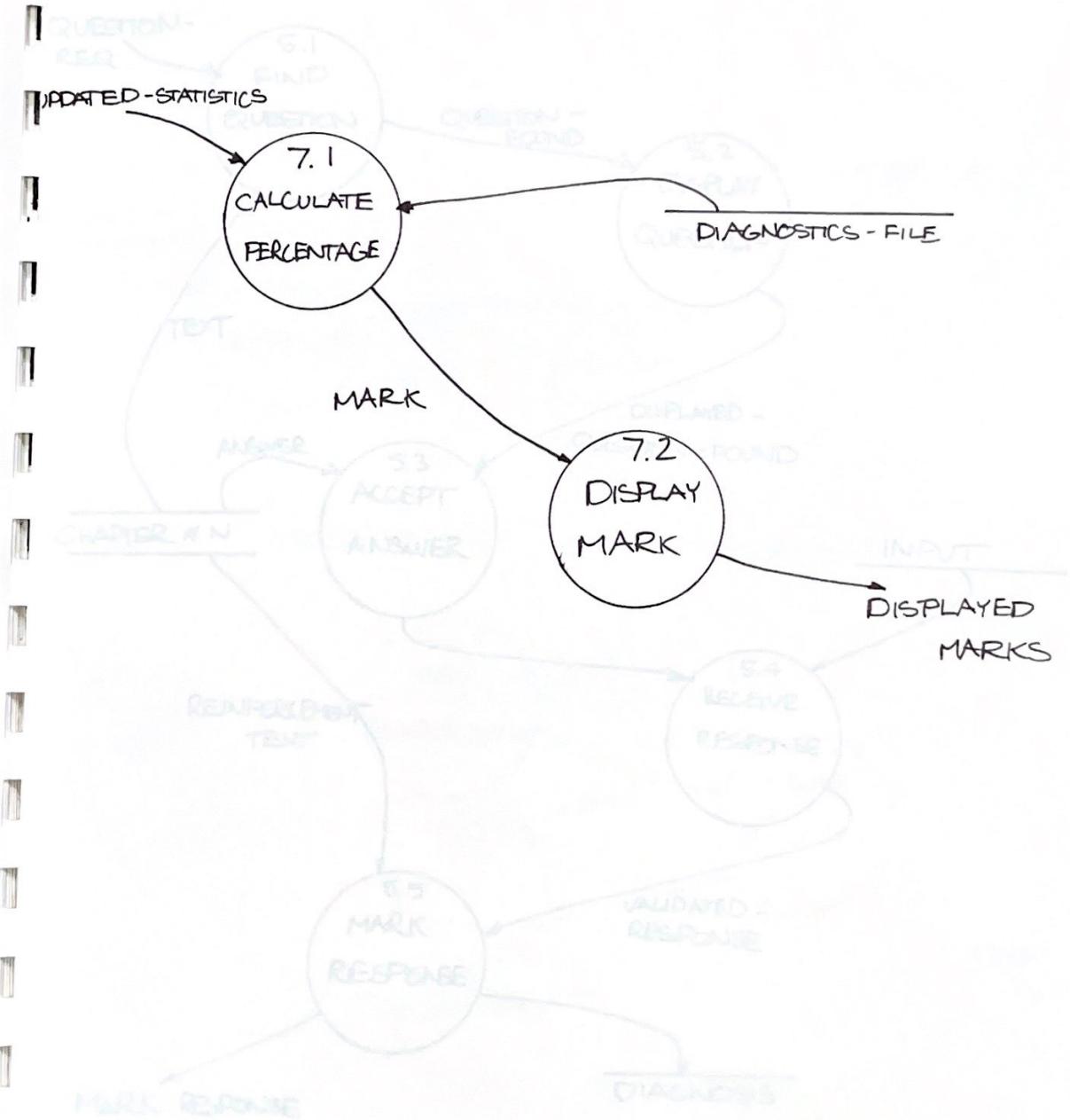


DIAGRAM 5 : PRESENT QUESTIONS

DIAGRAM 5.5 MARK RESPONSE

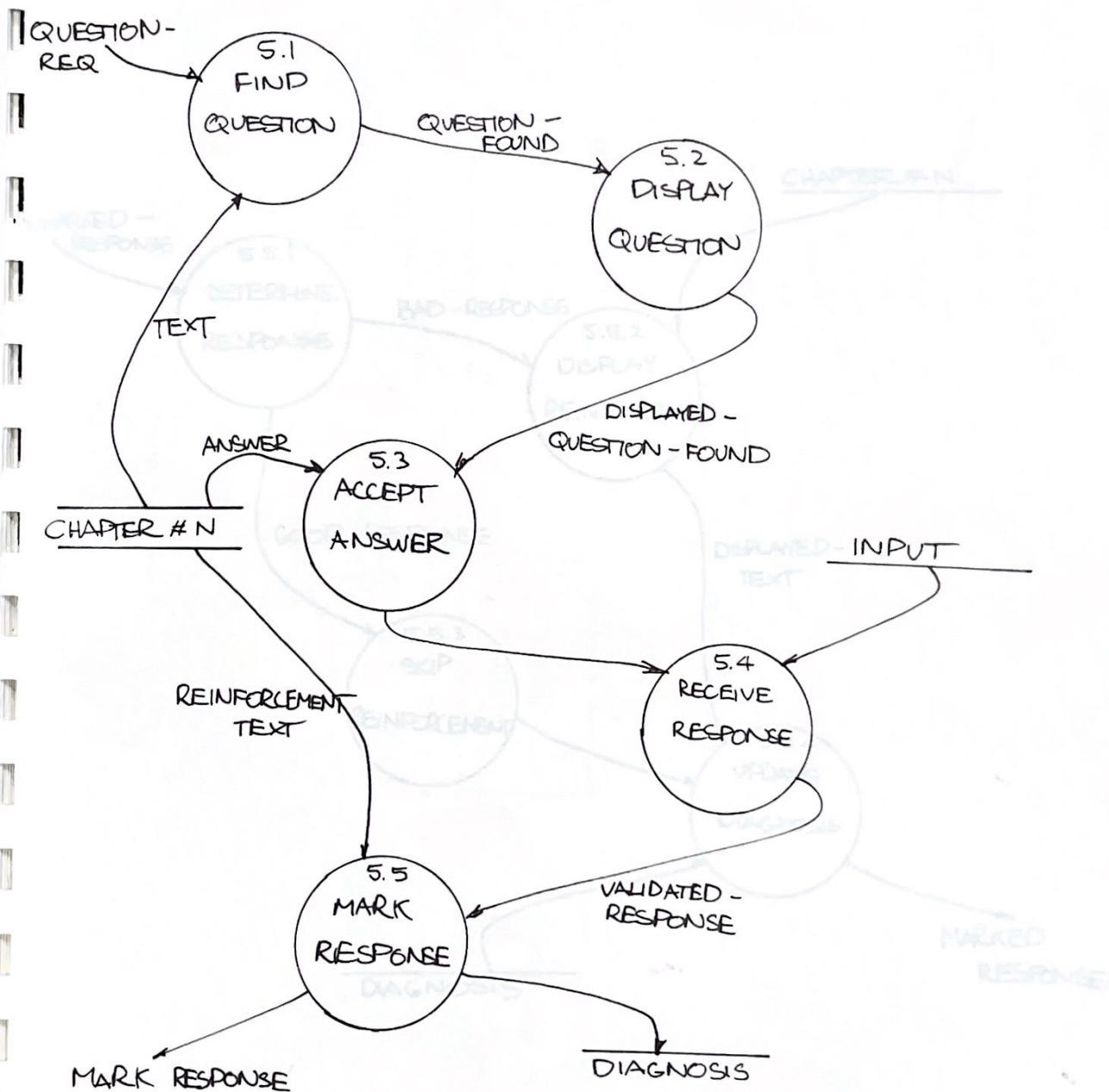
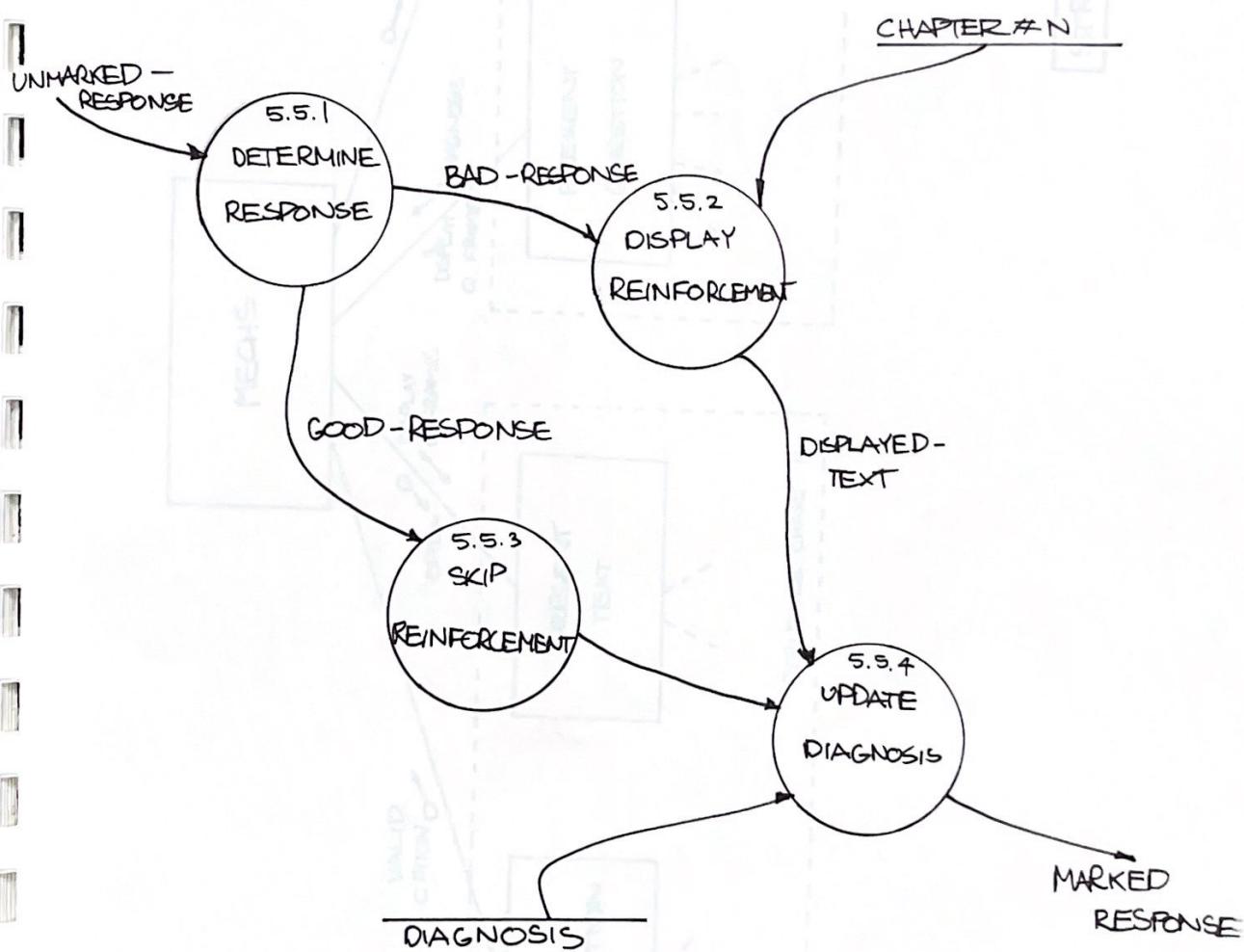
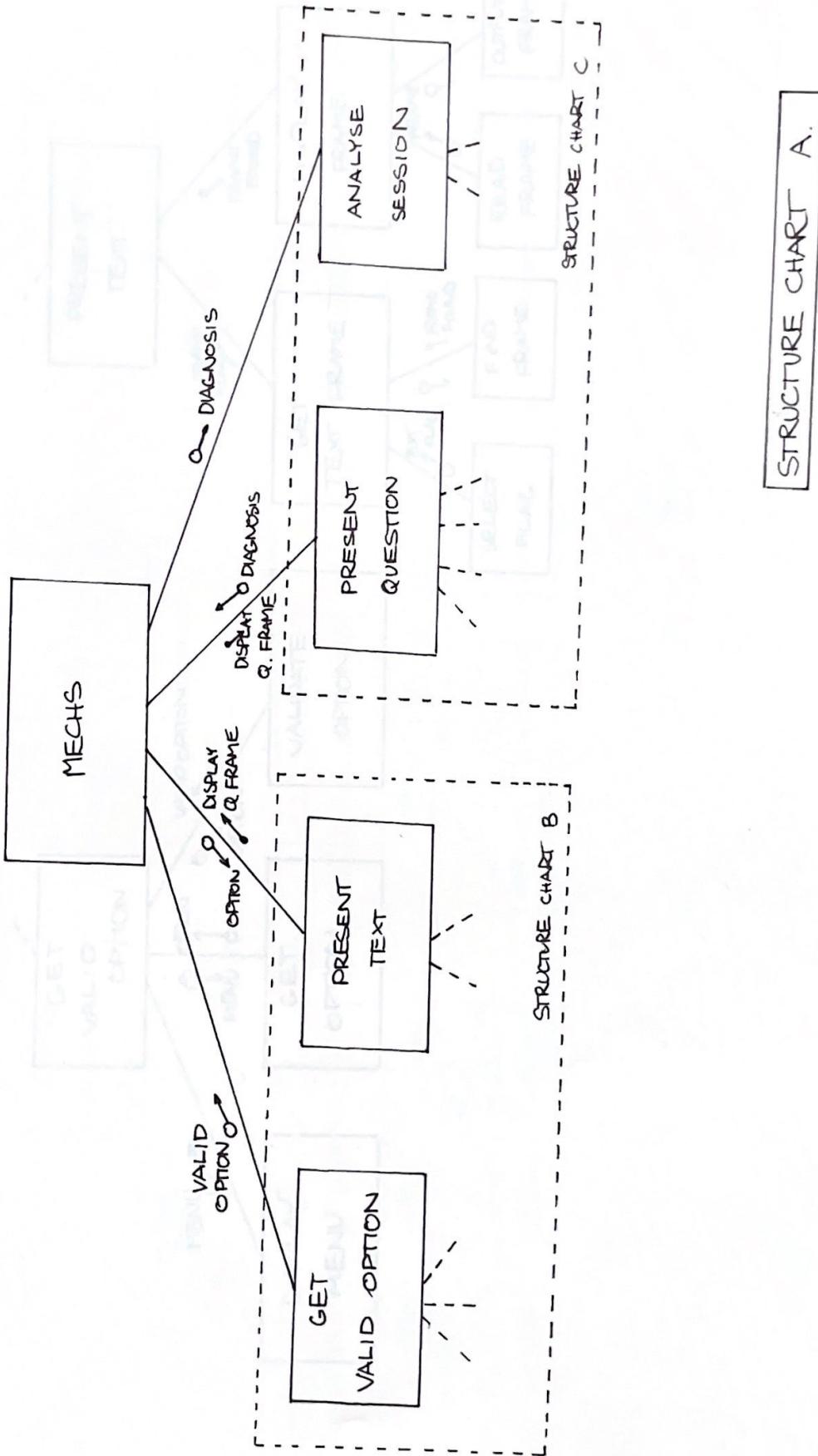
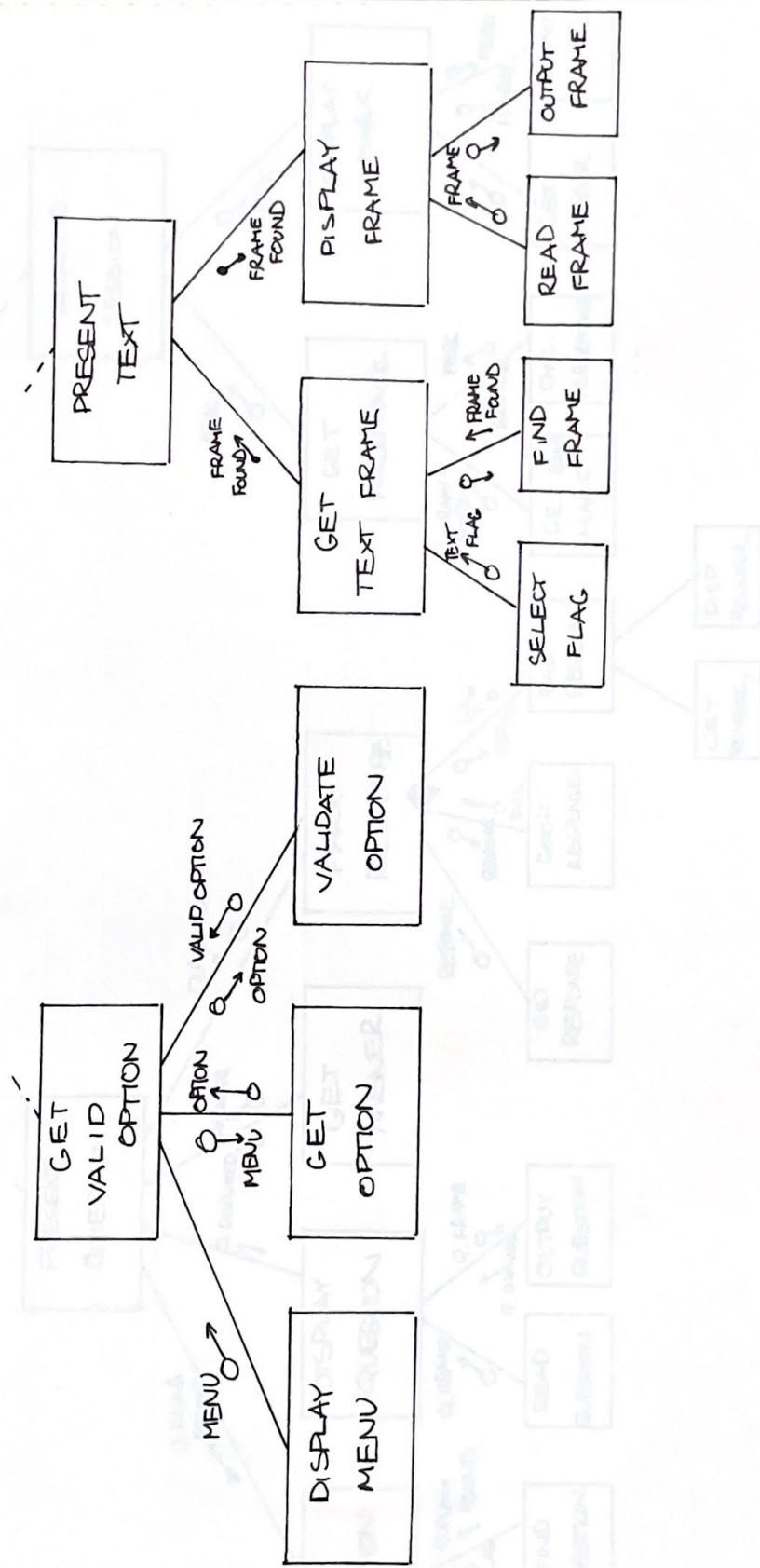


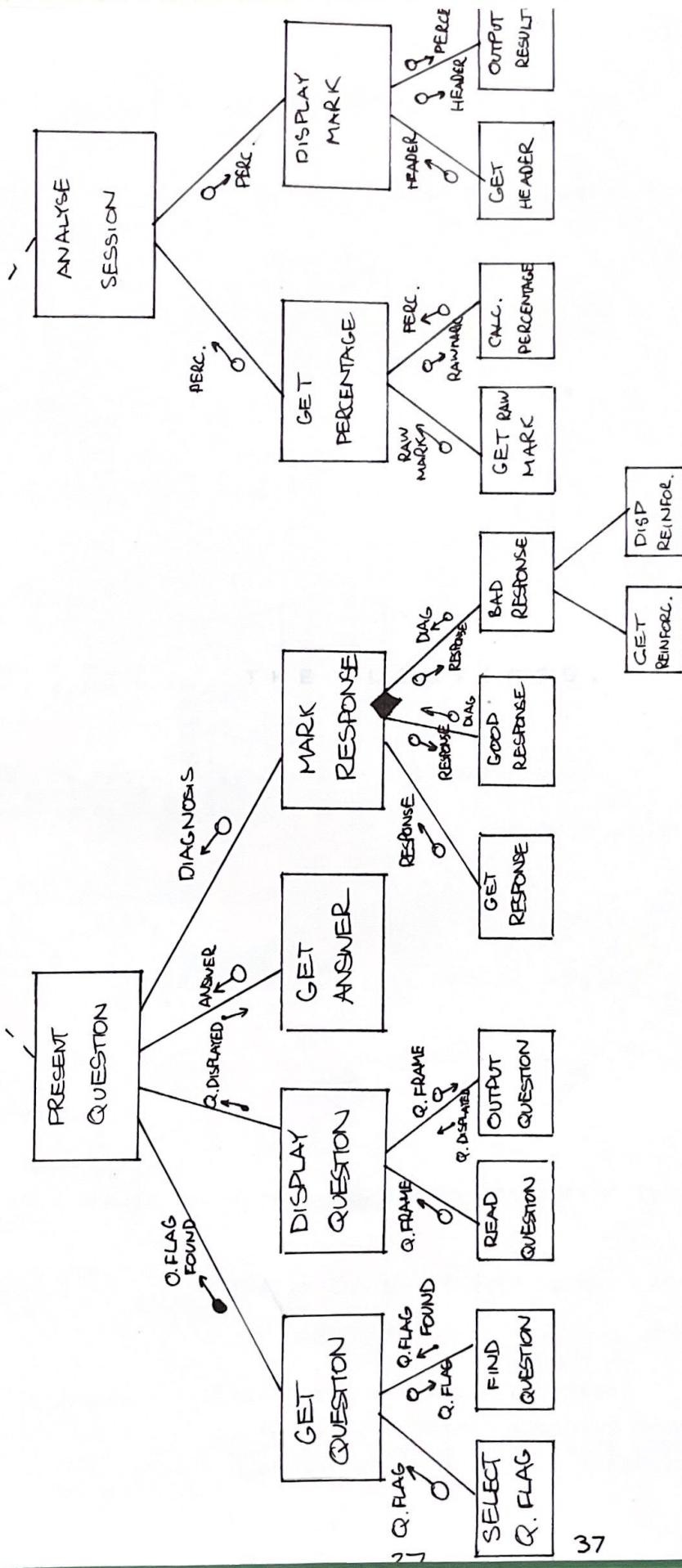
DIAGRAM 5.5 : MARK RESPONSE





STRUCTURE CHART B





STRUCTURE CHART C


```

program mechs;{version 1.0}

{The program for the CAL project to teach 'O' level mechanics. ESC is the}
{Written : February - April 1987. at first 'ESC' is the same as 'HOME',}
{Author : S J Jabbar. at this represents 'ESC B' which is cursor down}
{for point 1011 x 00}
{write(chr(12)*C*) (This represents 'ESC C' which is cursor right)}
{Introduction}

{constants}

const
  null = 0; {initial x,y coordinates}; {maximum}
  finish = 7;
  text_flag = '{-T}'; {parameter note this routine draws a straight line}
  question_flag = '{-Q}'; {maximum}
  answer_flag = '{-A}'; {maximum}
  reinforcement_flag = '{-R}'; {maximum}
  end_of_chapter_flag = '{-E}'; {maximum}
  introduction_flag = '{-I}'; {maximum}

  write(chr(12)) {switches the alternative character set on.}
  write(chr(15)) {maximum}
  write(chr(7) to 70 etc. (the * is a straight line character in}
  declarations {maximum}
  writechr(15)) {maximum} {switches the normal character set on.}

type
  options = 0..7;
  diagnosis_rec = record
    option_no : options;
    no_of_questions : integer;
    correct_answers : integer;
  end;
  diagnostic_rec = array[1..6] of diagnosis_rec;
  x_coordinates = 1..80;
  y_coordinates = 1..24;

procedure draw_line(x1,y1,x2,y2);
var
  screen_1:screen;
  option : options;
  chapter : text;
  header : string;
  diagnosis : diagnosis_rec;
  diagnostics : diagnostic_rec;
  x : x_coordinates; {block 3 and 21 in this routine.}
  y : y_coordinates;
begin {draw_line}
  writeln(screen_1);
  writeln(option);
  writeln(chapter);
  writeln(header);
  writeln(diagnosis);
  writeln(diagnostics);
  writeln(x);
  writeln(y);
end; {draw_line}

{procedures}

procedure clear_screen;
{This routine clears the screen and leaves the cursor in the top left}
{hand corner.} {line is evoked the message in quotes is displayed at line 20.}

begin {clear_screen}
  write(chr(12)) {12 is the ASCII code number for clear screen}
end; {clear_screen}

{continues}
{press any key to continue. when you are ready, press any key to continue.}
{reduces}

procedure gotoxy(x : x_coordinates;y : y_coordinates);
{This routine takes the cursor to the 'home' position (top left)
and then it uses the coordinates x and y and moves the cursor to
the line y, character x position on the screen. The maximum value
of x is 80 and y is 24.}

```

```

    after the screen has been cleared and defined by the lines drawn by
    routine draw_line. This routine displays the chapter titles in this
    count : integer;

begin {gotoxy}
  write(chr(27),'H');           {27 is the ASCII code number for ESC, the}
  for count := 1 to y do       {effect of 'ESC H' is the same as 'HOME'.}
    write(chr(27),'B');         {This represents 'ESC B' which is cursor down}
  for count := 1 to x do
    write(chr(27),'C');         {This represents 'ESC C' which is cursor right}
end; {gotoxy}

procedure draw_line(y : y_coordinates); {Density.}
{Using the alternative character set, this routine draws a straight line
at the specified line number.}

var
  count : integer;
  num : integer;
  count1 : integer;
begin
  write(chr(14)); {switches the alternative character set on.}
  gotoxy(5,y);
  for count := 1 to 70 do {the '*' is a straight line character in }
    num:=chr(49+count); {the alt chr set.}
    write('*');
  write(chr(15)); {switches the normal character set on.}
end;

procedure split_screen_1;
{The screen is split at lines 2,4 and 21 in this routine.}
function power(x,y : integer):integer;
begin {power}
  if y=0 then power:=1
  else power:=x*power(x,y-1);
end; {power}

procedure split_screen_2;
{The screen is split at lines 3 and 21 in this routine.}
function power(x,y : integer):integer;
begin {power}
  if y=0 then power:=1
  else power:=x*power(x,y-1);
end; {power}

procedure continue;
{When this routine is evoked the message in quotes is displayed at line 22.}

var choice_option;
  ch : char;
begin {continue}
  gotoxy(5,22);write('When you are ready, press any key to continue. ');
  read(ch);
  gotoxy(1,22);writeln(' ':80);
end; {continue}

procedure display_menu;

```

```

After the screen has been cleared and defined by the lines drawn by
split_screen 1, this routine displays the chapter titles in this
project 'syllabus'.} {in this section there is no
also validation := false;
}

begin {display_menu}
  clear_screen; { then action := finish else option := valid_ans }
  split_screen 1;
  gotoxy(30,3);writeln('Master Screen');
  gotoxy(1,10);
  writeln('1. Force of Gravity, Weight and Friction.'':59);
  writeln('2. Speed, Velocity and Acceleration.'':60);
  writeln('3. Newtons Laws of Motion.'':60);
  writeln('4. Work, Energy and Power.'':60);
  writeln('5. Machines.'':60); { this procedure resets variable
  writeln('6. Density and Relative Density.'':60);
  writeln;
  writeln('7. End this session.'':60);
end; {display_menu}

```

```

function enumerated(ch : string):integer;
{This function takes the string ch and converts it into an integer.
It does this by taking the left most character in ch and converting
it to an integer value, then this is multiplied by 10 to the power
of the number of positions the character is away from the right most
character. This process is repeated on all the characters.}

```

```

procedure set_header { header }
  num : integer;
  count:integer; { the variable header with the name of the
  current chapter.}

```

```

function power(x,y): integer):integer;
{This function returns the value x to the power of y.} {Friction}
{Speed, Velocity, Acceleration}
{Newton's Laws of Motion}
{Work, Energy and Power}
{Machines}
{Density and Relative Density}

```

```

var
  count : integer;
  dum : integer;
begin {power}{header}
  dum := 1;
  for count := 1 to y do
    dum := dum * x;
  power := dum
end; {power} {by_flag : string}

```

```

{ this routine reads the chapter text until the first occurrence of
the specified flag. Then the screen is cleared and split into
begin {enumerated} the top right hand corner the page number is
num := 0; in three (16 lines) of text are displayed and the user
for count := length(ch) downto 1 do continue. Lines of text are
  num:=(num+(ord(ch[count])-ord('0')))* power(10,(count-1));
  enumerated := num;
end; {enumerated}

```

```

  line : string;
  num : integer;

```

```

procedure choose_option;
{ask for input until stine = flag}
{This procedure is called after the menu has been displayed. It asks
the user for a number, corresponding to a chapter and validates it.}

```

```

var
  clear_screen;
  number : string;
  valid_option : boolean;
  valid_ans : integer;
begin {choose_option}
repeat
  gotoxy(1,22);write(' ':80);

```

```

    gotoxy(5,22);write('Type in the number of the chapter you require : ');
    readln(number);
    if enumerated(number) in [1..7] then valid_option := true
        else valid_option := false;
until valid_option;
valid_ans := enumerated(number);
if valid_ans = 7 then option := finish else option := valid_ans
end; {choose_option}
end; {display_file}

procedure open_file;
{When a valid option has been chosen, this procedure resets the variable
chapter to the relevant CHAPTER.}
begin {open_file} displays the question. The routine reads the text
case option of
    1 : reset(chapter,'CHAPTER#01'); once of the flag.
    2 : reset(chapter,'CHAPTER#02');
    3 : reset(chapter,'CHAPTER#03');
    4 : reset(chapter,'CHAPTER#04');
    5 : reset(chapter,'CHAPTER#05');
    6 : reset(chapter,'CHAPTER#06')
end; {question}
end; {open_file} chapter until (line = question_flag)
clear_screen;
split_screen_1;

procedure set_header; {question}
{This routine assigns the variable header with the name of the
current chapter.}
begin {set_header}
    writeln(header);
    readln(header,line);
begin {set_header}
    case option of
        1 : header := 'Force of Gravity, Weight and Friction';
        2 : header := 'Speed, Velocity and Acceleration';
        3 : header := 'Newton's Laws of Motion';
        4 : header := 'Work, Energy and Power';
        5 : header := 'Machines';
        6 : header := 'Density and Relative Density'
    end; {set_header}
end; {set_header}

question_flag : string;

procedure display_frame(flag : string);
{This routine reads the chapter text until the first occurrence of
the specified flag. Then the screen is cleared and split into
three parts. In the top right hand corner the page number is
written, then a frame (15 lines) of text are displayed and the user
is asked to indicate when they want to continue. Lines of text are
displayed until the second occurrence of the flag.}
var
    line : string; {the chapter file asks for a valid answer from the user}
    pnum, lnum : integer; {etc}
begin {display_frame}
    repeat readln(chapter,line) until (line = flag);
    readln(chapter,line); string;
    pnum := 0; {etc}
    while not(line = flag) do
    begin
        clear_screen;
        split_screen_1;
        pnum := pnum + 1;
        gotoxy(5,3);
        writeln(header); {the user has answered a question correctly.}
        gotoxy(65,3); {before he is correct and then swipe over the
        writeln('at Page ',pnum:2); for comments. It also updates the
        gotoxy(1,5);writeln;
    end;
end;

```

```

lnum := 1;
while not((lnum = 15) or (line = flag)) do
begin
  writeln(' ':10,line);
  readln(chapter,line);
  lnum := lnum +1
end;
continue
end; {display_frame}

{The first question is no_of_questions + 1}
procedure display_question(qnum : integer);
{This routine clears the screen and splits it into three parts.
The first part contains the number of the question being displayed,
the second part displays the question. The routine reads the text
lines encountered before the first occurrence of the question flag and displays the
lines encountered until the first occurrence of the reinforcement_flag.}
var
  line : string; {line is called if the user has answered a question incorrectly.
                  the answer should have been and the told that
                  it will be displayed explaining the answer. After this
                  the diagnosis record is updated.}
begin {display_question}
repeat readln(chapter,line) until (line = question_flag);
readln(chapter,line);
clear_screen;
split_screen-1;
gotoxy(5,3);
writeln('Question ',qnum:1);
gotoxy(1,5);writeln;
while not(line = question_flag) do {see from the terminal.}
begin
  writeln(' ':10,line); {if incorrect, the answer is *correct_answer*}
  readln(chapter,line); {any key for an explanation of this answer.}
end;
end; {display_question}
ent_text;
{The diagnosis record is updated}
no_of_questions := no_of_questions + 1;
end; {display_frame}

procedure present_text;
{The frames of text are displayed using the routine display_frame.}
var
  chapter_flag : string;
begin {present_text}
  display_frame(text_flag)
end; {present_text}

procedure present_questions;
{The record diagnosis is initialised in this routine and while the end
of chapter flag has not been encountered it displays a question, reads
the answer from the chapter file asks for a valid answer from the user
and then marks this answer.}
var
  question_flag, line : string; {valid options are A, B, C, D}
  correct_ans, ans : char; {good response}
  qnum : integer; {question number}
end; {present_questions}

procedure good_response;
{This routine is called if the user has answered a question correctly.
It tells the users that he/she is correct and then skips over the
section of text that serves as reinforcement. It also updates the
diagnosis record.}

```

```

var line : string; ans : character;
begin {good_response}
  gotoxy(25,19);writeln(' ':80);
  gotoxy(25,19);writeln('Yes ',ans,', is the correct answer.');
  repeat readln(chapter,line) until (line = reinforcement_flag);
  repeat readln(chapter,line) until (line = reinforcement_flag);
  with diagnosis do
    begin
      no_of_questions := no_of_questions + 1; {This routine is executed
                                                 after the chapter has been completed and displayed with the
                                                 correct answers. If no chapters were
                                                 finished otherwise it displays an overall
                                                 percentage completion.}
      correct_answers := correct_answers + 1;
    end;
    continue;
end; {good_response}

procedure bad_response;
{This routine is called if the user has answered a question incorrectly.
 He/she is told what the answer should have been and the told that
 reinforcement text will be displayed explaining the answer. After this
 text has been displayed the diagnosis record is updated.}

var ch : char; ans : character;
flag : string;
begin {bad_response}
  write(cfr(7)); {This produces a noise from the terminal.}
  gotoxy(25,19);writeln(' ':80);
  gotoxy(25,19);writeln(ans,' is incorrect, the answer is ',correct_ans);
  gotoxy(5,22);write('Press any key for an explanation of this answer.');
  read(ch);
  header := 'Reinforcement text';
  display_frame(reinforcement_flag);
  with diagnosis do
    begin
      no_of_questions := no_of_questions + 1;
      if (no_of_questions <= correct_answers) then
        percentage := correct_answers/no_of_questions*100;
      else
        percentage := (correct_answers/no_of_questions)*100;
    end;
  end; {bad_response}

begin {present_questions}
  with diagnosis do
    begin
      option_no := option;
      no_of_questions := 0;
      correct_answers := 0;
    end;
  qnum := 0;
  while not (line = end_of_chapter_flag) do
    begin
      qnum := qnum + 1;
      display_question(qnum);
      repeat readln(chapter,line) until (line = answer_flag);
      readln(chapter,correct_ans);
      repeat readln(chapter,line) until (line = answer_flag);
      repeat
        gotoxy(25,19);writeln(' ':80);
        gotoxy(25,19);write('Type in either A B or C : ');
        readln(ans);
        {The next line changes lower case letters to uppercase}
        if (ans >= 'a') then ans := chr(ord(ans)-32);
      until ((ans = 'A') or (ans = 'B') or (ans = 'C'));
      if (ans = correct_ans) then good_response
                                else bad_response;
      readln(chapter,line);
    end;
end; {present_questions}

procedure update_stats;

```

```

The record diagnosis is added to the list of diagnostics.)
```

```

begin {update_stats}
  diagnostics[option] := diagnosis
end; {update_stats}
```

```

procedure title;
begin
  title of the program and name of the author are displayed using
```

```

procedure analyse_session;
```

```

{When the user has chosen the option to finish this routine is executed
to produce the marks. If a user has completed a chapter then the
percentage gained in the chapter is calculated and displayed with the
chapter name. If a chapter has not been completed then a not attempted
comment is displayed with the chapter name. If no chapters were
attempted then the routine finishes otherwise it displays an overall
percentage before completion.)
```

```

var
  count : integer;
  percentage,total_correct,total_questions : integer;
  none_attempted : boolean;
```

```

begin {analyse_session}
  clear_screen;
  split_screen(1);whether he/she would like an introduction to the
  progotoxy(28,3);writeln('DTEA GLEN OSMITHICS'); is displayed
  on total_correct:=0;
  total_questions:=0;
  none_attempted:=true;
  for count := 1 to 6 do
    begin
      option := count;
      set_header;
      if not(diagnostics[count].option_no = null)
      then with diagnostics[count] do
        begin
          progotoxy(28,1);percentage := round((correct_answers/no_of_questions)*100);
          writeln;writeln(' ',5);
          if (ans = ' ') then writeln('In the chapter "',header,'" you gained ',percentage:1,' %');
          begin
            total_correct := total_correct + correct_answers;
            total_questions := total_questions + no_of_questions;
            none_attempted := false
          end;
        end;
      else
        begin
          writeln;writeln(' ',5);
          writeln('The chapter "',header,'" was not attempted.');
        end;
    end;
  gotoxy(27,20);
  if not(none_attempted) then
    begin
      percentage := round((total_correct/total_questions)*100);
      writeln('Your overall mark is ',percentage:1,' % ');
    end;
  gotoxy(1,22);
  writeln(option = finish) go
end; {analyse_session}
```

```

procedure initialise;
begin
  initialise
  open_text;
  set_header;
  set_chapter;
  set_questions;
```

```

procedure initialise;
```

```

The diagnostics records are initialised in this routine.)
```

```

var
  count : integer;
```

```

begin {initialise}
  for count := 1 to 6 do
    with diagnostics[count] do
      begin
        option_no := null;
        no_of_questions := null;
```

```

    correct_answers := null
end; {initialise} attempt to teach you some mechanics.
of physics which is concerned with the
of matter energy.

You will be presented with six chapters any of which you
procedure titles; matter.

The title of the program and name of its author are displayed using
this routine.}

After this you can the chapter you will be asked to
questions on the chapter. Reinforcement
begin {titles} and should it be deemed necessary.
clear_screen;
split_screen_2; when an analysis of the session is
gotoxy(30,10);writeln('M E C H S ');
gotoxy(30,11);writeln('-----');
gotoxy(20,13);writeln('A Program to teach **0** Level mechanics.');
gotoxy(45,14);writeln('by S J Jabbar.');
continue;
end; {titles}

```

procedure introduction;

The user is asked whether he/she would like an introduction to the program. If they do then the file containing this is displayed on split_screen_2.

```

var
  flag, ans : string;
begin {introduction}
  clear_screen;
  split_screen_2;
  gotoxy(20,11);writeln(' ':10);
  gotoxy(20,11);write('Would you like an introduction ? ');
  readln(ans);
  if ((ans = 'YES') or (ans = 'Y') or (ans = 'yes') or (ans = 'y')) then
    begin
      reset(chapter,'INTRO.TEX');
      header := 'Introduction.';
      display_frame(introduction_flag)
    end;
end; {introduction}

```

begin {main body}

```

  titles;
  introduction;
  display_menu;
  initialise;
  choose_option;
  while Not(option = finish) do
  begin
    open_file;
    set_header;
    present_text;
    present_questions;
    update_stats;
    display_menu;
    choose_option;
  end;
  analyse_session
end. {main-body}

```

{I}

CHAPTER 01 : FORCE ON SOLIDS

INTRODUCTION : INTRO.TEX

{I}

This program will attempt to teach you some mechanics. Mechanics is a branch of physics which is concerned with the relation of matter to energy.

You will be presented with six chapters any of which you may attempt. These chapters have some pages of text which explain the subject matter.

If a body is moving it can move. If the body is already moving a force can alter its speed or direction of motion or change its route.

After you have read the chapter you will be asked to try and answer some questions on the chapter. Reinforcement text will be supplied should it be deemed necessary.

When you have finished, an analysis of the session is displayed showing how well you answered the questions from the chapters attempted. You will also be given an overall mark.

Symbols it can be written as:

If $F = m \cdot a$ then F is force

if $m = m \cdot a / F$ then m is mass (mass)

if $a = F / m$ then a is acceleration (acceleration)

GRAVITATIONAL FORCE

There is a force that acts on us constantly and

is the force that pulls us towards the earth. This is called gravitational force. Sir Isaac Newton

studied that gravitational force exists between all objects.

Newton's law of universal gravitation states that any two particles of matter attract one another with a force which is proportional to the product of their masses and inversely proportional to the square of their distance apart.

So two stones are not only attracted towards the earth but also to each other.

Gravitational attraction is also responsible for holding the moon in its orbit about the earth and also the earth and its fellow planets in their orbits round the sun. This particular force, which keeps a body moving in a circle is a force directed towards the centre. It is called centripetal force.

Weight is the effect of gravitational force.

Weight is the name of the force due to gravity and experiment has shown that on average on Earth the force of gravity on a mass of one kilogram is 9.8 N.

So the weight of one kilogram is 9.8 N.

Weightless

Weightlessness

Weightlessness occurs when the body's gravitational pull is equal to the centrifugal force required to make the body move at a constant speed and constant radius. There is a balance between the gravitational force and the centripetal force so that there would be no reaction force acting on the body. It would be weightless.

FORCE

Force is that which changes a body's state of rest or of uniform motion in a straight line.

In general terms this means that when a body is acted upon by a force it will begin to move. If the body is already moving a force may alter its speed or alter its direction of motion or bring it to rest.

The unit of measurement for force is called the newton (N) and is defined as follows :

One newton is the force required to give a mass of one kilogram an acceleration of one metre per second every second.

In symbols it can be written as:

given, $F = ma$ if a mass force is applied to a body it first comes to rest since an equally if $m = 1 \text{ kg}$ and $a = 1 \text{ m/s}^2$ then force $F = 1 \text{ N}$

GRAVITATIONAL FORCE just begins to slip. At this point of contact has reached its maximum value

There is a force that acts on us constantly and this is the force that pulls us towards the earth.

This is called gravitational force. Sir Isaac Newton concluded that gravitational force exists between all bodies.

Newton's law of universal gravitation states that any two particles of matter attract one another with a force which is proportional to the product of their masses and inversely proportional to the square of their distance apart.

So two stones are not only attracted towards the earth but also to each other.

Gravitational attraction is also responsible for holding the moon in its orbit about the earth and also the earth and its fellow planets in their orbits round the sun. This particular force, which keeps a body moving in a circle, is a force directed towards the centre. It is called centripetal force.

Friction to the force of gravity and the surface together are called the coefficients of static and sliding friction, respectively.

WEIGHT coefficient of friction is denoted by the Greek letter μ . Thus:

Weight is the name of the force due to gravity and experiment has shown that on average on the earth, the surface force of gravity on a mass of one kilogram is 9.8 N.

So the weight of one kilogram is 9.8 N.

What is the force acting on it?

A) 9.8 N

B) 99.8 N

C) 999.8 N

WEIGHTLESSNESS

Weightlessness occurs when the earth's gravitational pull is equal to the centripetal force required to maintain a particular speed and orbital radius. Since the difference between the gravitational force and the centripetal force is zero there would be no resultant force acting on the body. It would be weightless.

Friction

Friction is the name given to the force which opposes the relative sliding motion of two surfaces in contact with one another. Without friction trains would not be able to stop since the breaks work through friction. Then weighed again.

There are two types of friction static and sliding.

- (A) Greater than before
- (B) Less than before
- (C) Half the size

STATIC FRICTION

If a gradually increasing force is applied to a body it will, at first, continue to remain at rest since an equally increasing but oppositely directed force of friction comes into action at the under surface of the body. At any instant the particular moment, we say that the pull and the opposing frictional force are in equilibrium. When the distance apart reached when the body just begins to slip. At this point, the friction brought into play has reached its maximum value for the two surfaces concerned and this is called static friction.

SLIDING FRICTION

If the body is pulled along so that it slips at a steady speed we notice that the force required is rather less than the static frictional force. Sliding friction is therefore less than static.

COEFFICIENT OF FRICTION

In any body static and sliding friction are increased in proportion to the force, perpendicular to the surfaces, which is pressing them together. A body to a surface has a value of 30%.

The Ratios of the static and sliding friction to the force pressing the surfaces together are called the coefficients of static and sliding friction respectively.

Coefficient of friction is denoted by the greek letter mu Thus:

$$\mu = \frac{F}{R} \quad \text{where } F = \text{force of friction}$$

(B) and (C) R = force of pressing surfaces together

A car of mass 1000 kg moves with an acceleration of 2 m/s^2 . What is the force acting on it?

Two greater should be same in two parts since the equation involving the forces (A) and (B) is independent of friction (but does not cover the case). It adds up the force of friction (F) and this can be combined with (B) 998 N (action (F)) to give mass (m).

$$m = F/g \quad (C) 2000 \text{ N}$$

(A) force = F and R
(B) force = F and R
(C) force = F and R
(D) force = F and R

The equation relating force, mass and acceleration is

{ T }

t

$$F = ma$$

since $m = 1000 \text{ kg}$
and $a = 2 \text{ m/s}^2$

$$F = 1000 \times 2 = 2000 \text{ N}$$

{ -R }

{ -Q }

The earth is not a perfect sphere but bulges at the equator.
If an object was weighed at the pole and then weighed again

but at the equator would its weight be:

- A) Greater than before
- B) Less than before
- C) Still the same

{ -Q }

{ -A }

B

{ -A }

{ -R }

A force between two bodies, in this case the earth and the object, depends upon their masses (which is constant) and is inversely proportional to the square of their distance apart.

Since the distance to the centre of the earth is greater at the equator, the weight would be less.

{ -R }

{ -Q }

Is it possible to walk on a perfectly smooth surface ?

- A) Yes
- B) No
- C) It depends on mass

{ -Q }

{ -A }

B

{ -A }

{ -R }

Walking is not possible if there is no friction between the ground and the soles of the shoe. A perfectly smooth surface is frictionless.

{ -R }

{ -Q }

A force (R) pressing a body to a surface has a value of 50 N. The coefficient of friction (μ) is 2 and the body is accelerating (a) at 10 m/s^2 .

What is the mass of the body ?

- A) 1000 kg
- B) 250 kg
- C) 10 kg

{ -Q }

{ -A }

C

{ -A }

{ -R }

This problem should be done in two parts since the equation linking the force (R) and the coefficient of friction (μ) does not give the mass. It does give the force of friction (F) and this can be combined with the acceleration (a) to give mass (m).

$$\mu = F/R$$

$$\begin{aligned} \text{therefore } F &= \mu R \\ &= 2 \times 50 \\ &= 100 \text{ N} \end{aligned}$$

T)

t $F = m a$
Therefore $m = F/a$
 = $100 / 10$
 = 10 kg

R}

E}

