UNIVERSITY OF CAMBRIDGE INTERNATIONAL EXAMINATIONS

Cambridge International Diploma in Computing
Advanced Level

Scheme of Work

5218
Further Systems and Software
Core Module

UNIVERSITY *of* CAMBRIDGE
International Examinations

Introduction

This section provides candidates with knowledge and understanding of the following aspects of computer systems:

- the functions of operating systems
- the functions and purposes of translators
- computer architectures and the fetch-execute cycle
- data representation, data structures and data manipulation
- programming paradigms
- databases
- use of systems and data
- systems development, implementation, management and applications
- simulation and real-time processing
- common network environments, connectivity and security issues

Recommended Prior Knowledge

Candidates should have studied Module One.

Tutor Preparation Required to Start This Module

- decide on the programming language to be used ensuring that candidates can fulfill all the syllabus requirements with the selected package. This could be the same programming language used for Module One
- install this software so that it is accessible to all students
- provide instructions showing what the students have to do
- prepare a bank of appropriate supplementary resources such as work done by students in previous years, brochures, catalogues, worksheets to test students' knowledge at each stage

Important note

Some centres may wish to deliver elements of Module Four (Programming Project) alongside this module. To help centres that wish to use this approach, the relevant session plans and performance criteria for the theory work on Module Three are mapped for each element listed in Module Four.

Session Plan 101 – Functions of Operating Systems

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.1**	3.1.1 Features of operating systems	• describe the main features of operating systems

Classroom Exercises	**Notes**
Review Operating System work for Module One concentrating on the characteristics of: • Single-User • Multi-User • Network Systems Introduce the features of Operating Systems that support multi-users and networking • Memory Management • Scheduling • Distributed Systems Short student centred exercise using worksheets to research / reinforce / test knowledge – perhaps filling in the missing words in a series of questions about features needed from a list of the modes.	Include: • Memory Management • Scheduling Algorithms • Distributed systems

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.1**	3.1.2 Scheduling 3.1.4 Job Queues & Priorities	• define and explain the purpose of scheduling, job queues, priorities and how they are used to manage job throughput

Classroom Exercises

Introduce the concepts of jobs, processes and scheduling.

Define the terms

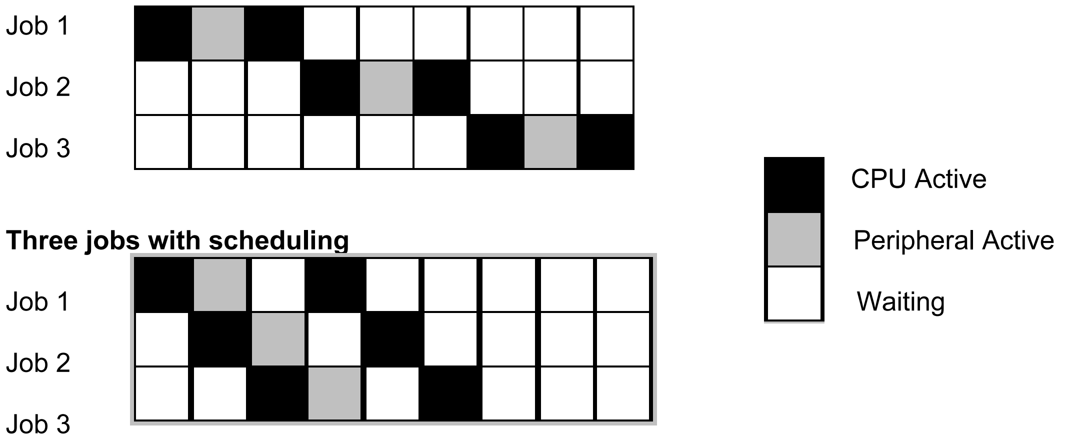
- job
- job queue
- priorities (including the concepts of processor bound and peripheral bound)
- process (including states running, runnable and suspended)
- scheduling

Introduce scheduling and discuss the following benefits:

- maximise use of hardware resources
- maximise throughput
- allocate resources fairly to all users
- provide acceptable response time for interactive users
- provide acceptable turnaround time for batch users
- manage system performance (e.g. temporarily increase time taken to respond if the system is overloaded)
- prevent deadlock

Use simple diagrams to show the benefits of scheduling e.g.

Three jobs without scheduling



Resources

- prepared diagrams to show the benefits of scheduling jobs

Notes

Include the following scheduling algorithms

- shortest job first
- shortest remaining time
- round robin

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.1**	3.1.3 Interrupt Handling	• explain how interrupts are used to obtain processor time and how processing of interrupted jobs may later be resumed

Classroom Exercises	Notes
Define the term interrupt (a signal from some device/source seeking the attention of the processor), the different classes of interrupt and the need to assign different priorities to interrupts (so that when two interrupts occur at the same time or an interrupt occurs whilst another is being serviced, the interrupt with the highest priority is dealt with first). Classes of interrupt should include: • Hardware failure — Highest Priority • Program • Timer • I/O — Lowest Priority Short student centred exercise using a worksheet to research / reinforce / test knowledge – perhaps identifying sources of interrupts and the class and priority of each interrupt.	Typical sources of interrupts should be identified including the following classes: • program generated • processor time generated • I/O both normal operation and error conditions • hardware failure Describe algorithms and data structures • pointers Vectored interrupt handling
Introduce concept of interrupt service routines and outline the sequence of actions 1. save status (registers etc.) 2. determine cause (poll status flags) 3. take relevant action 4. restore status 5. return Explain, using diagrams on the board, the use of vectors to determine the location in memory of the appropriate routine.	

Session Plan 102 – Memory Management, PC and Network Operating Systems

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.1**	3.1.7 Modern personal computer operating systems	• describe the main components of a typical desktop PC operating system • describe the main components of a network operating system

Classroom Exercises	Notes
Define the terms boot file, a file containing commands to automatically configure a personal computer on start up, and file allocation table (FAT), a list held on disk by an operating system to maintain and manage disk space used for file storage. Provide a selection of annotated printouts showing boot files e.g. config.sys, win.ini etc. Discuss the types of boot files used by different operating systems e.g. DOS, Windows, Linux etc. which make them appropriate for use by different types of users. Use classroom discussion to identify the components of a personal computer operating system, as students should have experience of using at least one operating system. Extend this discussion to identify the extra features required by a network operating system including the following features: • transparency • directory services • security • network printing (including a definition of spooling) Short student centred exercise using a worksheet to research / reinforce / test knowledge – perhaps identifying features of operating systems and the advantages they provide for users.	Include: • use of the file allocation table • purpose of the boot file Include: • transparency • directory services • security • network printing (including a definition of spooling)

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.1**	3.1.6 Spooling	• describe spooling, explaining why it is used

Classroom Exercises

Use classroom discussion to identify the components of a personal computer operating system, as students should have experience of using at least one operating system. Extend this discussion to identify the extra features required by a network operating system including the following features:

- transparency
- directory services
- security
- network printing (including a definition of spooling)

Short student centred exercise using a worksheet to research / reinforce / test knowledge – perhaps identifying features of operating systems and the advantages they provide for users.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.1**	3.1.5 Memory management	• explain how memory is managed in a typical modern computer system

Classroom Exercises	Notes
Define the following terms: • virtual memory (include the reasons for use e.g. allows more processes to be run than could be held in main memory) • paging • segmentation Using diagrams on the board (or pre-prepared as a handout), explain the operation of segmentation and paging in virtual memory systems, highlighting the differences between the two systems.	Include: • virtual memory • paging • segmentation

Session Plan 103a – Functions and Purposes of Translators

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.2**	3.2.1 Types of Translator 3.2.2 Lexical Analysis 3.2.3 Syntax Analysis 3.2.4 Code Generation 3.2.4 Linkers and Loaders	• describe the difference between interpretation and compilation • describe what happens during lexical analysis • describe what happens during syntax analysis, explaining how errors are handled • explain the code generation phase • explain the purpose of linkers and loaders

Classroom Exercises

Review types of translator (Module One Session Six) for High-level languages and the conversion of source code to object code. Extend this to highlight the differences between compilation and interpretation including at a minimum:

- compiler translates the whole program (source code) into object code that can be stored and re-used
- interpreter translates and executes a program line by line. No object code is stored for further use, a program has to be translated each time it is used

Discuss the advantages and disadvantages of compilation and interpretation highlighting when it would be appropriate to use a compiler or an interpreter (e.g. use an interpreter during program development as errors can be easily checked and modified). As students have used translators they should be able to contribute to a discussion.

Introduce the stages of compilation:

- lexical analysis
- syntax analysis
- code generation
- linking and loading

Describe in general terms what happens during each phase including tokenisation, the use of the symbol table and handling errors. Use sample code from a programming language that your students are familiar with to demonstrate the general principles.

Short student centred exercise using worksheets to research / reinforce / test knowledge – perhaps filling in the missing words in a series of questions about what happens at each stage of compilation.

Resources

- review translators from Module One
- worksheet to re-enforce knowledge – perhaps using examples from a familiar High Level Language

Notes

Include:

- source code
- object code
- instruction explosion

Session Plan 103b – Computer Architectures and the Fetch-Execute Cycle

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.3**	3.3.1 Von Neumann architecture 3.3.2 Registers: purpose and use	• describe basic Von Neumann architecture, identifying the need for, and the uses of, special registers in the functioning of a processor

Classroom Exercises	Notes
Introduce the concept of Von Neumann architecture – any computer that takes a single instruction then obeys it before processing the next instruction. Describe the contents and the use of the following registers: • Sequence Control Register • Current Instruction Register • Memory Address Register • Memory Buffer Register • Accumulator • Status Register Test orally or using a worksheet ensuring students clearly understand the purpose of these registers.	Include the following registers: • Sequence Control Register • Current Instruction Register • Memory Address Register • Memory Buffer Register • Accumulator • Status Register

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.3**	3.3.3 Fetch-execute cycle	• describe, in simple terms, the fetch/decode/ execute/reset cycle, and the effects of the stages of the cycle on specific registers.

Classroom Exercises	Notes
Prepare a diagram showing the elements and flow of the fetch-execute cycle. If possible provide a demonstration of the fetch-execute cycle using one of the computer programs commercially available and/or search for and use one of the demonstrations available on the world-wide-web. Using a set of simple Assembly Language/Machine Code instructions trace the contents of each of the registers, this can be done as a whole class exercise giving the opportunity to work through the cycle several times using different types of instruction.	Include the following registers: • Sequence Control Register • Current Instruction Register • Memory Address Register • Memory Buffer Register

Resources

- prepared diagram showing the fetch-execute cycle

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.3**	3.3.4 Parallel processors	• discuss parallel processor systems, their uses, advantages and disadvantages

Classroom Exercises	**Notes**
Define parallel processing (the simultaneous use of several processors to perform a single job). Compare this to the Von Neumann computer. Perhaps use a simple everyday analogy to introduce this concept e.g. the use of multiple checkouts in a supermarket rather than a single checkout. Describe the different types of parallel processing including at a minimum: • SIMD (Single Instruction Multiple Data Stream) • MIMD (Multiple Instruction Multiple Data Stream) • MPP (Massively Parallel Processing) Provide mini pre-determined scenarios of the use of parallel processing e.g. weather forecasting, processing live images from a satellite, artificial intelligence. Ensure that the pre-determined scenarios include advantages (e.g. higher throughput, better fault tolerance, better price: performance ratio etc.) and disadvantages (e.g. requires OS capable of managing parallel processing, application needs to be able to break down larger tasks into smaller tasks that can be performed in parallel, processing overhead to synchronise processes etc.)	Include: • SIMD (Single Instruction Multiple Data Stream • MIMD (Multiple Single Instruction Multiple Data Stream) • MPP (Massively Parallel Processing)

Session Plan 104 – Data Representation – Number Systems

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.4**	3.4.1 Number systems	• express numbers in binary, binary coded decimal (BCD), octal and hexadecimal • describe and use two's complement and sign and magnitude to represent positive and negative integers • perform integer binary arithmetic: addition and subtraction

Classroom Exercises	Notes
Revise the use of number bases, counting in binary (base 2) and hexadecimal (base 16) and conversion to and from each of these number bases and denary (base 10). Extend this work to include octal (base 8) and Binary Coded Decimal (BCD). Provide students with a worksheet containing codes in binary, octal, hexadecimal and BCD to be converted into denary. Also provide conversions from denary values in all three number bases and BCD (include how many bytes would be required). Marking these work sheets in class and providing model answers will reinforce the necessity of showing good, clear working. Demonstrate, with board work, the use of two's complement and sign and magnitude to represent positive and negative numbers. Stress how to represent both positive and negative numbers because many students often only consider the use of negative numbers. Introduce addition and subtraction using both two's complement and sign and magnitude for integers. Encourage checking by adding/subtracting denary numbers and converting the answer to two's complement or sign and magnitude and comparing this with the binary integer addition/subtraction. Provide a worksheet with practice questions converting positive and negative denary integers to two's complement and sign and magnitude and addition and subtraction of the binary integers. Provide questions that given the number of bits available (e.g. 1 byte, 2 bytes etc). This will allow for discussion of overflow.	• revision of conversion denary-binary-hexadecimal • extend to include octal and BCD • students need practice in both conversion and addition/subtraction

Resources

- prepared questions for students to attempt with model answers

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.4**	3.4.2 Floating point binary	• demonstrate an understanding of floating point representation of a real binary number

Notes

Include definitions of :

- Mantissa
- Exponent
- Overflow
- Underflow

Use both representations of exponent:

- two's complement
- sign and magnitude

Resources

- prepared examples of decimal numbers and their floating point representation

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.4**	3.4.3 Normalisation of floating binary numbers	• normalise a real binary number • discuss the trade-off between accuracy and range when representing numbers

Classroom Exercises

Explain the structure of a floating-point number, including definitions of the mantissa (non-zero fractional part) and exponent (integer power). Provide examples showing the range of values that can be stored and how a normalised number allows for the greatest precision for a given size of mantissa. Explain how the increase in range leads to a decrease in precision and introduce the ideas of underflow (exponent too small) or overflow (exponent too large) as the result of a calculation.

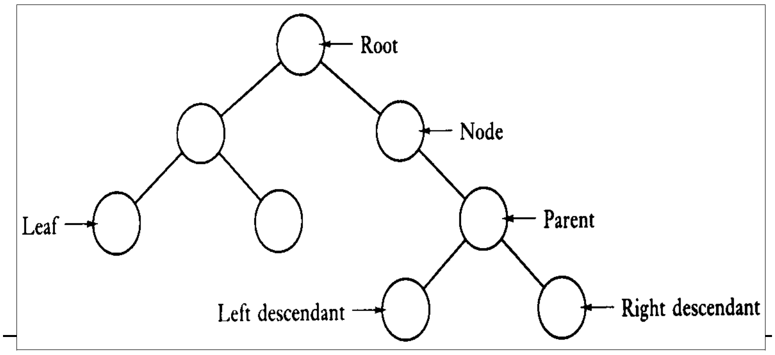
Set worksheet exercises to practise the conversion of a decimal number to binary floating point and binary floating-point numbers to decimal. Include positive and negative numbers, large numbers and fractional values.

Resources

- prepared questions for students to attempt with model answers

Session Plan 105 – Data Representation – Data Structures

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.4**	3.4.4 Implementation of data structures	• explain the difference between static and dynamic implementation of data structures, highlighting the advantages and disadvantages of each • describe algorithms for the insertion, deletion and amendment of data items stored in linked-list, stack and queue structures • describe insertion, deletion and amendment of data items in a tree structure

Classroom Exercises	Notes
Revise the purpose of, and structures of arrays, stacks, queues (including the concepts of LIFO, FIFO, stack pointers, pushing and popping to and from stacks and queues) and linked lists. Discuss the advantages and disadvantages of static and dynamic data structures. Introduce the tree data structure using simple diagrams and explaining the different types of tree traversal (pre-order, post-order and in-order). **Tree Structure**  Root Node Leaf Parent Left descendant Right descendant Provide a set of worksheets showing algorithms for: • insertion of an element • deletion of an element • amendment of an element for use with the following dynamic data structures: • stacks • queues • linked lists • trees Demonstrate the use of these algorithms with one or two sets of data. It may be appropriate to provide an algorithm say for insertion of data in a queue and let the students attempt to write their own algorithm for deletion before introducing a model answer. Students attempt worksheets with questions on pushing and pulling to and from stacks and queues, insertion and deletion of	Students are not expected to use any particular form to present algorithms, but should be able to write procedural algorithms in some form Include the following data structures: • linked lists • stacks • queues • trees

elements from linked lists and trees. Again the marking of these questions may be better as a class discussion to reinforce the concepts studied.	

Resources

- revision of work done in Module One
- a set of algorithms for
  - insertion
  - deletion
  - amendment
- sets of data for use with the algorithms

Session Plan 106 – Data Representation – Data Manipulation

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.4**	3.4.4 Searching and sorting	• explain the difference between binary searching and serial searching, highlighting the advantages and disadvantages of each • explain the difference between insertion sort, quick sort and merge sort • describe algorithms for implementing insertion sort, quick sort and merge sort methods

Classroom Exercises	Notes
Provide a set of worksheets showing algorithms for: • binary search • serial search Demonstrate the use of these algorithms with several sets of data. Choose the data sets very carefully to show the advantages and disadvantages of each type of search by using both algorithms on the same set of data. Provide a worksheet that contains more sets of data to be searched; the same set of data could be used with several search criteria. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.	Develop a clear, consistent style of writing algorithms for your students to follow. Ensure that they have plenty of practice in dry running these algorithms and setting out the process in a clear diagrammatic form.
Demonstrate on the board the following sorts: • insertion sort • quick sort • merge sort Start with the insertion sort, as it is the easiest one to understand. Demonstrate each sort and provide a worksheet containing the algorithm and further sets of data for student practice. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied. **Extension Activity** Students write a program to demonstrate some/all of the sorts. The insertion and quick sorts can use arrays held in memory, for the merge sorts it could be appropriate to use files.	Include: • insertion sort • quick sort • merge sort

Resources

- algorithms for
  - binary searching
  - serial searching

- carefully selected test data to illustrate advantages and disadvantages of each type of search

- algorithms for
  - insertion sort
  - quick sort
  - merge sort

- carefully selected test data to illustrate advantages and disadvantages of each type of sort

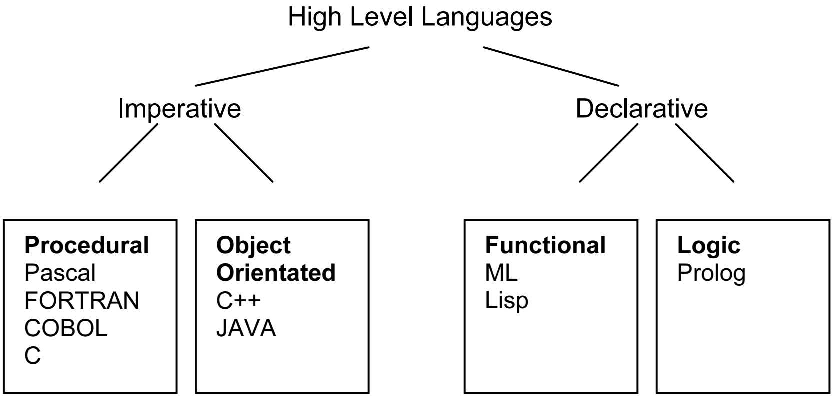
Session Plan 107 – Programming Paradigms

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.5**	3.5.1 Types of languages and typical applications	• describe with the aid of examples, the characteristics of a variety of programming paradigms • explain the terms object-oriented, declarative, procedural, and functional as applied to high-level languages

Classroom Exercises

Introduce the different types of High Level programming languages

Classification of Computer Languages



Provide short sample programs from a variety of paradigms:

- procedural
- object-oriented
- declarative (logic)
- declarative (functional)
- low level

Use these programs to show the characteristics of these paradigms bearing in mind the minimum requirements set out below for each language.

Provide definitions of the following types of programming languages:

- declarative
- imperative
- procedural
- object oriented
- functional
- logic

Student centred exercise using worksheets to research / reinforce / test knowledge – perhaps filling a table to show types of programming languages, examples of languages, examples of suitable uses, advantages and disadvantages.

Notes

Include the following paradigms:

- low level
- imperative
- procedural
- declarative
- functional
- object-oriented

Students are not expected to write or interpret the meaning of simple segments of low-level language code.

A detailed knowledge of the syntax of programming languages is not required.

Resources

- provide sample programs from a variety of paradigms

Assessment Objectives	Performance Criteria	Classroom Ideas
• 3.5	3.5.2 General features and features of procedural languages	• explain how functions, procedures and their related variables may be used to develop a program in a structured way, using stepwise refinement • describe the use of parameters, local and global variables as standard programming techniques • explain how a stack is used to handle procedure calling and parameter passing

Classroom Exercises

Review top down approach, procedures and functions (Module One, Sessions Three and Five) and introduce stepwise refinement. Describe the use of global variables, local variables and parameter passing (by value and by reference). Illustrate these concepts with a programming language that your students are familiar with e.g. C++, Pascal, Java etc.

Student centred exercise to write short programs that use:

- global variables
- local variables
- procedures
- functions
- value parameters
- reference parameters

Each function and procedure should have one purpose e.g. calculator program with input procedure, output procedure, calculation functions (+ - * /) etc.

Review functions of a stack, explain the use of a stack to handle procedure calling and return including pushing of return address, parameter values/addresses on entry to a procedure and popping of same on exit from a procedure. Perhaps use examples from the programming exercise above to demonstrate.

Session Plan 108 – Procedural, Declarative, Functional and Object Oriented Languages

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.5**	3.5.2 Features of object-oriented Languages	• discuss the concepts and, using examples, show an understanding of data encapsulation, classes and derived classes, and inheritance

Classroom Exercises	Notes
Explain the concepts of object-oriented languages including at a minimum: • encapsulation (keeping together data structures and methods) • classes • derived classes • inheritance (derived classes carry the data structures and methods of the superclass) Use everyday examples to introduce these ideas e.g. class definition of clock, derived classes – analogue clock and digital clock. The use of Java or C++ examples can provide easily understood sections of program to demonstrate the above concepts.	Use the pre-prepared programs from session 107 to demonstrate the concepts.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.5**	3.5.2 Features of declarative languages	• discuss the concepts and, using examples, show an understanding of backtracking, instantiation and satisfying goals

Classroom Exercises
Explain the concepts of declarative languages including at a minimum: • rules • facts • backtracking • instantiation (binding of a variable to a value during resolution, lasting only long enough to satisfy one complete goal) • satisfying goals Use everyday examples to introduce these ideas e.g. classes of animals and the food that they eat. The use of Prolog examples can provide easily understood sections of program to demonstrate these concepts.

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• 3.5	3.5.2 Features of functional languages	• discuss the concepts and, using examples, show an understanding of list processing, and recursion

Classroom Exercises

Explain the concepts of functional languages including at a minimum:

- functions
- expressions
- list processing
- recursion

Use of simple binary tree examples to introduce these ideas as a tree is a recursive data structure. The use of Logo or Lisp examples can provide easily understood sections of program to demonstrate these concepts.

Session Plan 109 – Features of Low level Languages, Generations of Programming Language and Syntax Definition

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.5**	3.5.2 Features of low level languages	• explain the concepts of addressing of memory

Classroom Exercises	Notes
Using either a simplified Assembly Language or Machine Code instructions (in Hexadecimal) describe the following ways of addressing memory: • direct (using the contents of the address) • indirect (using the contents of the address as a pointer to another address) • indexed (using the contents of the address in combination with the contents of an index register to determine the address) This could be demonstrated by the use of a simple set of examples on the board or the use of a commercially available simulation program depending upon the resources available.	Must include: • direct • indirect • indexed Only concepts required here, as students are not expected to be able to write or interpret the meaning of low-level language code.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.5**	3.5.3 Methods for defining syntax	• explain the need for, and be able to apply, BNF (Backus-Naur form) and syntax diagrams

Classroom Exercises

Demonstrate on the board the use of Backus-Naur form (BNF) as a formal method to describe simple syntax of a programming language.

Use the following meta symbols:

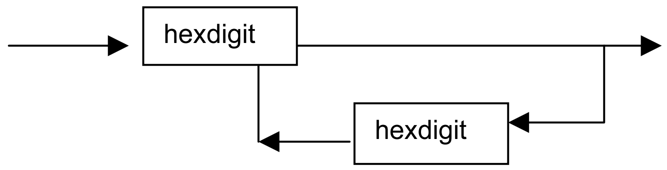
::=	is defined by
/	OR
<>	meta variable

E.g.
<hexdigit> ::= 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9 / A /B / C / D / E / F (0,1,…….,F are terminal symbols)
<hexnumber> ::= <hexdigit> / <hexdigit><hexnumber>

Student centred exercise using worksheets to reinforce / test knowledge – perhaps providing simple examples to extend. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.

Demonstrate on the board the use of syntax diagrams as a formal method to describe simple syntax of a programming language as a set of rules.

e.g. the syntax diagram for hexnumber is



To read the diagram, trace it from left to right, in the direction shown by the arrows, there must be at least one hexdigit followed by none or more hexdigits.

Student centred exercise using worksheets to reinforce / test knowledge – perhaps providing simple examples similar to those used for BNF. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.

Resources

- worksheets with exercises in defining syntax rules using both methods

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.5**	3.5.2 Generations of Programming Language	• using examples, describe the nature and purpose of 3rd and 4th generation languages

Classroom Exercises	**Notes**
Introduce the idea of classification of programming languages, into generations, according to changes in methodology: • 1st Generation (Machine Code) • 2nd Generation (Assembly Language) • 3rd Generation (High Level Languages) • 4th Generation (4GLs have very powerful commands e.g. SQL and usually accompany an application) • 5th Generation (Very-high-level Languages) Provide worksheets that describe in more detail the nature and purpose of 3rd and 4th generation languages then use classroom discussion to review the current session's work on programming languages and set the scene for the use of data manipulation languages in Session 110. Note: the above is one of the ways of classifying generations of programming languages; there are other classifications, e.g. starting with the zeroth generation.	This provides the link between programming languages and DMLs.

Session Plan 110 – Database Structures, Normalisation and E-R Modelling

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.6**	3.6.1 Database design	• describe flat files and databases • explain the advantages that using a relational database gives over flat files

Classroom Exercises	Notes
Review work done on files, indexing and key fields from Module One (Session Nine). Contrast this with the use of databases including the following types: • network • hierarchical • relational Use diagrams to show each type of data structure and explain about the advantages of each type concentrating on the speed of access to data for specific queries (Network and Hierarchical) and the availability of ad hoc queries (relational). Since relational databases are being used for more and more large databases, a pre determined scenario of a commercial relational database, e.g. a large on-line book seller, could help introduce the advantages of using a relational database rather than a flat file including: • data independence • data consistency • lack of duplication of data • less redundant data	Include: • network databases • hierarchical databases • relational databases

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.6**	3.6.2 Normalisation and data modelling	• design simple relational databases to the third normal form (3NF)

Classroom Exercises

Using a practical example of a previously set up relational database introduce the concepts of:

- tables
- primary keys
- foreign keys
- secondary keys
- views of data

Demonstrate and explain the purpose of each of these concepts using the pre-prepared database then introduce the students to the formally set out underlying data structures.

e.g. Table_loan (loan_no, bookno, libmemno, borrowdate, expreturndate, actreturndate)

Where loan_no is the primary key of the loan table
Bookno and libmemno are foreign keys from other tables in a library database.

Use whole class teaching with a board to demonstrate the principles of normalisation starting with a flat file data structure and working through the stages of normalisation:

- 1st normal form – remove repeating data
- 2nd normal form – remove partial key dependencies
- 3rd normal form – remove non key dependencies

Choose your examples very carefully to ensure the one used for demonstration and the first few that the students attempt need work to be done at all stages (many examples may not yield composite keys so there can be no partial key dependencies).

Prepared worksheet(s) to give the students practice at normalisation. Perhaps use this for homework and mark in class, as class discussion will help reinforce these concepts.

Provide pre-determined scenarios e.g. customer orders, student records etc. that allow the students to identify specify and normalise the data structures required. Again use this for homework and mark in class, as class discussion will help reinforce these concepts.

Notes

Include:

- tables
- primary keys
- secondary keys
- foreign keys
- views of data

The practical use of a relational database management system such as ACCESS to allow the students to develop their own database would reinforce these concepts and the use of forms, DDL, DML and access rights.

Resources

- worksheets providing different data structures to be normalised

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.6**	3.6.2 Normalisation and data modelling	• draw entity-relationship (E-R) diagrams to represent diagrammatically the data model

Classroom Exercises
Introduce the concepts of entities and relationships (one to one, one to many, many to many). Use everyday occurrences to demonstrate these concepts e.g. the student teacher model can be discussed showing the idea of a many to many relationship between student and teacher and how the introduction of other entities such as class meeting can help organise the model. Explain how the relationships need to be carefully labelled in order to show understanding and provide worksheets with pre-prepared examples and questions. Similar data structures can be used to the ones prepared for the normalisation exercise, this will help enforce how these two techniques complement each other. Note: These examples need to provide the students with practice in drawing E-R diagrams.

Resources

- worksheets using the same data structures to develop E-R models so that the results can be checked against the normalised data structure when there are no many to many relationships

Session Plan 111 – Database Structures and Management

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.6**	3.6.3 Methods and tools for analysing and implementing database design	• design forms for input, deletion, modification and querying of a database • describe the structure of a database management system (DBMS)

Classroom Exercises	Notes
Use a practical exercise to review the concepts of good HCI design from Module One Session 17 and design forms for: • data entry • data deletion • data modification • querying data	Students to reinforce understanding with practical work.
Referring to the practical work completed introduce the main functions of a DBMS: • Data Dictionary (an internal file containing the name, description, characteristics, relationships for each data item and information about programs and users. *NB the term Data Dictionary can also be used to describe similar information provided as part of system documentation*) • Data Description/Definition Language (DDL) • Data Manipulation Language (DML) Explain that this information is stored with the data in a database system. Students may have used a GUI to define and manipulate data but a demonstration of the underlying commands actually used (e.g. showing the SQL commands produced by a QBE query) could be used to show the functions of a DDL and a DML as SQL has both properties.	Include the function and purpose of: • Data Dictionary • Data Description Language (DDL) • Data Manipulation Language (DML)

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.6**	3.6.4 Control of access to relational database elements	• explain the importance of varying the access allowed to database elements at different times and for different categories of user

Classroom Exercises	Notes
Extend the discussion on DBMS to include the use of different views of the system for different categories of users and at different times. Discuss the type of access e.g. read, read/write etc; the view allowed e.g. different types of user could be allowed access to certain elements of data; the effect of time on availability of data e.g. some elements not yet released or some elements may have been archived. A pre-prepared database and/or pre determined scenario would help students understand these concepts e.g. a database of students, courses, and examination results or chose another example that the students will be able to relate to.	Include the following types of access: • read data • read/write data • design chapter

Session Plan 112 – Use of Systems and Data

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.7**	3.7.1 The commercial value of data	• identify data that has commercial value; explaining why such data has this value and discuss contemporary trends in the compilation and use of valuable databases

Classroom Exercises
Use pre determined scenarios to illustrate the commercial value of data. Current articles from the computer press could be used to provide examples that highlight the following: • commercial value of data (and the need for it to be relevant, from a reliable source, kept up to date etc.) Compilation and use of databases

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.7**	3.7.2 The importance of standards	• explain the advantages of standardisation and describe some areas of standardisation such as file formats, ISDN, OSI model and its use together with communications protocols.

Classroom Exercises	**Notes**
Describe how the use of standards has aided computerisation by using examples of current de facto (wide use has led to market domination e.g. Windows) and de jure (pre-defined industry standards e.g. OSI model). Discuss in class the advantages of the following standards and also use this to review the work on networks (Session 12) from Module One: • file formats • ISDN • OSI model and communications protocol	Include: • file formats • ISDN • OSI model • communications protocols

Resources

- pre determined scenarios

Assessment Objectives	Performance Criteria	Classroom Ideas
- **3.7**	3.7.3 Communications and electronic commerce	- describe ways in which computers aid communication - identify situations in which the transmission of data has created/could create new opportunities for businesses and individuals to conduct

Classroom Exercises	Notes
Describe the use of computers in communications, including as a minimum: - voicemail - email - digital telephone facilities - Internet use by e-commerce - tele/video conferencing - Electronic Data Interchange (EDI) Ensure that a discussion about how e-commerce works is included e.g. use a flow diagram to describe the process of buying goods on-line. As current information is readily available, small groups of students (two or three) could be given the task of researching one use for homework, in preparation for a class discussion on the underlying issues. Each group should report their findings and the interactive discussion/marking of these ideas should develop a sound understanding of these concepts.	Include: - voicemail - email - digital telephone system facilities - e-commerce over the Internet - tele/videoconferencing - Electronic Data Interchange (EDI)

Resources

- pre determined scenarios

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.7**	3.7.3 Communications and electronic commerce	• select and justify an appropriate network system for a particular application

Classroom Exercises	Notes
Review work on networks from Module One Session 12, provide worksheets with examples of requirements for networks (e.g. use, data storage, data communications etc) and elements required to build a network (e.g. hardware, cabling, bandwidth etc), the students can select and justify the appropriate elements required for each network. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.	Include bandwidth required to transmit different forms of data: • text • sound • real-time sampled data and Video Hardware required: • file servers • hubs • repeaters • switches

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.7**	3.7.4 Training	• identify and describe training and re-training requirements for a given situation.

Classroom Exercises

Use pre determined scenarios (try and include both critical and non-critical systems) to illustrate both long-term and short-term changes that occur when a computerised system is introduced, consider patterns of work and quality of output. Use these scenarios to identify training and retraining requirements. Discussion of the requirements should develop a sound understanding of both these concepts.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.7**	3.7.5 Effects of introducing systems	• describe the substantial changes which occur as a result of introducing computing systems

Notes

Include short-term and long-term changes:

- in patterns of work
- in quality of output

Session Plan 113 – Systems Development and Implementation

Assessment Objectives	Performance Criteria	Classroom Ideas
- **3.8**	3.8.1 Methodologies and software tools for system development	- identify and describe how the use of methodologies/ techniques and software tools for developing computer systems aid the systems analyst/designer and programmer in terms of the documentation, step-by-step logical progression through tasks and cross-checking mechanisms

Classroom Exercises	Notes
Discuss the ways that a system can be decomposed using a variety of methods including: - Data Flow Diagrams (flow of data through system, starting with context diagram, level 1 diagram and level 2 diagrams) - E-R Modelling (Identification of data objects, their structure and the relationships between them) - Process Definition (description of the processes identified in the level 2 data flow diagram) - Data Dictionary (details of all the elements in the system) Show how formal methods can be used to document and crosscheck the above e.g. the use of SSADM (structured systems analysis and design method). Also discuss the benefits that abstraction (using different kinds of model to concentrate on one aspect of the system at a time) bring to the system developer. CASE tools for SSADM would provide a useful demonstration.	Include: - E-R Modelling - Data Flow Diagrams - SSADM

Resources

- review work from Module One, Sessions 13 and 14, and Module Three, Session 110

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.8**	3.8.2 Application types and technical requirements	• discuss the technical requirements of a system necessary to implement a range of different computer applications • explain the need to provide appropriate response times for different applications and its implications for hardware, software and data structures

Classroom Exercises	**Notes**
Use pre prepared scenarios (try and include applications requiring different response times) to discuss the technical requirements of a computer system, including: • hardware • data structures • operating systems • communications • interface software • other utility software Discuss response times and why they are sometimes different.	Include: • hardware • operating systems • communications • interface software • other utility software

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.8**	3.8.3 Choice of implementation approaches	• select, plan and justify appropriate implementation approaches for a range of different applications

Classroom Exercises	**Notes**
Review work from Module One, Session 14 on implementation approaches; provide extra pre determined scenarios for the students to take decisions on. Discuss the justification for each decision taken. Ensure that the students understand the differences between different approaches to implementation and that they can explain which is most appropriate in given circumstances.	Include these approaches: • direct • parallel • phased • pilot

Session Plan 114 – Project and Systems Management

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.8**	3.8.4 Project management and software tools	• describe the process of project management and give examples of different aids a project manager may use to plan and monitor a project identifying the benefits and drawbacks of each • discuss, giving examples, the requirement for effective project management for the implementation of different computing applications, including the benefit of using project management software

Classroom Exercises

Describe the process of project management including:

- timescales
- deadlines
- deliverables

Discuss the problems of managing the overall progress of projects with many sections (usually the case if a top-down approach has been used). Introduce the idea of a systematic calendar approach using a Gantt chart (showing timed project activities) to map and time proposed project activity.

Provide pre determined scenarios in a worksheet to demonstrate the use of Gantt charts and a set of examples for students to draw their own charts for homework. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.

Extension Activity

Students use a Gantt chart to map and time the proposed activities for their own project. This will reinforce the need for a well-structured analysis and design.

Introduce critical path analysis by looking at problems with using Gantt charts for larger projects, where later project activities depend upon the successful completion of earlier activities. Discuss the need for the early identification of these critical activities.

Provide pre-prepared examples of simple PERT charts to illustrate the ideas of paths, with timed activities and numbered events. These charts can be abstract using activities A, B, C etc. These charts can be used to show how to calculate the critical path.

Provide some pre-prepared PERT charts for the students to use to calculate the critical path, ensuring that these are of varying difficulty. Mark these in class and use the problems in calculations where there are many possible paths to introduce the benefits of using project management software.

Notes

The following aids should be included:

- Gantt charts
- critical path analysis

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.8**	3.8.5 Systems management and monitoring	• discuss the implications of managing, monitoring and maintenance of systems

Classroom Exercises	Notes
Discuss the management, monitoring and maintenance of working systems including at a minimum: • the need for quality control and management and the use of appropriate tools • the need for up-to-date documentation of the system • the benefits and implications of the use of a Software Audit • the implications of any hardware updates Again the use of pre determined scenarios and classroom discussion will benefit students' understanding of these concepts.	The following should be included: • up-to-date documentation • software audit • quality control and management • hardware updates

Session Plan 115 – Real Time Applications

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.9**	3.9.1 Applications of real-time computing	• describe real-time applications

Classroom Exercises

Describe a variety of real time systems stressing the need for speed of response to external events but also including the need for reliability and recovery.

Introduce the idea of a feedback loop by describing a simple system e.g. a temperature control system attached to a heater and a fan. Also discuss the need for sensors and actuators to implement this system. Extend this work to look at a variety of other real time systems that use the following types of signals:

- visible
- tactile
- audible
- other physical signals

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.9**	3.9.2 The feedback loop; input and output; sensors and actuators	• explain the use of sensors and actuators for visible, tactile, audible and other physical signals

Assessment Objectives	**Performance Criteria**	**Classroom Ideas**
• **3.9**	3.9.3 The use of robots	• discuss the use of robots in a variety of situations

Classroom Exercises	**Notes**
Extend the discussion on sensors to include robots. A useful starting point could be to consider the three laws of robotics devised by Isaac Asimov in his book ‘I, Robot’ published in 1950. 1. A robot may not injure a human being, or through inaction, allow a human being to come to harm 2. A robot must obey the orders given it by human beings except where such orders would conflict with the first law 3. A robot must protect its own existence as long as such protection does not conflict with the first or second law Make use of pre prepared scenarios to stimulate discussion of the use of robots in manufacturing (e.g. high precision jobs such as painting, welding and riveting) and hazardous environments e.g. (cleaning toxic waste or bomb disposal).	Cover these situations: • manufacturing • hazardous environments

Session Plan 116 – Simulation and Parallel Processing

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.9**	3.9.4 Uses of simulation	• explain the reasons for simulation, such as to change time-scales and/or save costs and/or avoid danger • describe the uses of simulation to assist in design, to make predictions, to test hypotheses

Classroom Exercises	Notes
Introduce simulation with a discussion about the life-like properties of games machines used at home or in computer arcades. Extend this discussion to look at the different reasons for modelling different types of situation: • predictions e.g. weather forecasting • design e.g. testing stresses in bridge design • hypotheses e.g. a country's economics over varying time scales and conditions Discuss the importance of observing the effect of the variable elements in any simulation and also simulation limitations where there are unpredictable, random events e.g. the effect of a coup d'etat on a country's economy, very bad weather in flight simulators. Set an exercise for students, perhaps in small groups, to research a simulation and identify the following: • type (situation simulation e.g. flight simulator, design, prediction, hypothesis) • uses and reasons for use • variables • limitations • processing requirements including identification of those systems that require MPP (Massively Parallel Processing) Ensure that a variety of types are chosen and students prepare a short presentation of their research. Presentation to the class of this research by the students, and guided discussion would help develop the students' understanding of a wider range of simulations.	Ensure that a variety of different types of simulation are included.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.9**	3.9.5 Variation of parameters and conditions; time steps	• describe a simulation and its variables, the facility to vary conditions and observe the sensitivity of results to such variations

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.9**	3.9.7 Advantages and limitations of simulations	• discuss the advantages of simulation in testing the feasibility of a design • discuss the limitations of simulation, especially where the situation is subject to random events

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.9**	3.9.6 Processing requirements	• explain the large processing requirements of some systems and hence recognise the need for parallel architectures

Session Plan 117 – Data Transmission, Network Components and Environments

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.10**	3.10.1 Data transmission	• describe methods used to organise LANs and WANs • describe typical rates of data transmission associated with different topologies and methods • describe different media for transmitting data and their carrying capabilities

Classroom Exercises	Notes
Review work done on networking in Module One Session 12. Provide two pre determined scenarios one of a large LAN that contains several smaller LANs and one of a WAN showing the following elements. (Large diagrams showing elements, topology, transmission rates etc. would be useful): • topology • connections (include the media used and possible/actual transmission rates) • routing (include the use of switches, bridges, routers, modems/terminal adaptors and their effects on the transmission rates) • protocols (also discuss effects on transmission rates) • network operating systems Student centred exercise using worksheets to reinforce/test knowledge – perhaps providing simple scenarios for the students to design a suitable network and provide justification for the components chosen. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.	Include: • topology • connections • routing • protocols • network operating systems

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.10**	3.10.2 Network components	• explain the different purposes of network components

Notes

Include the following:

- switches
- routers
- bridges
- modems

© CIE 2002

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.10**	3.10.4 Common network environments	• discuss common network environments, their facilities, structure and ability to exchange information using appropriate software and techniques • describe how a network environment affects the user interface provided • describe the facilities provided by electronic mail systems (including voicemail) • explain that distribution of a network can have implications both for data and responsibility

Classroom Exercises	Notes
Define open networks, referring to previous work on standards and networks. As students will probably have used open networking systems e.g. the Internet and/or a school Intranet, use a brainstorming session to identify the facilities available and the advantages offered. Describe the layers of the OSI model, using a pre-prepared diagram, show how this theoretical system relates to TCP/IP used for the Internet and Intranets. Explain at which level packet switches, bridges and routers operate. Discuss the exchange of information over networks, perhaps compare and contrast the use of different methods e.g. FTP (File Transfer Protocol) with the use of ATM (Asynchronous Transfer Mode) or Netbui/Samba. Set the students an exercise containing a list of different types of information to be exchanged. The students have to decide and justify which method would be most suitable. Verbal debrief on the above exercise.	Include: • Intranets • the Internet • other open networks
Describe and demonstrate the use of email including the following facilities: • composing • responding • filing • copying • attaching • sending on • multiple recipients Set a practical exercise for the students to complete each of the above tasks. Also give the students a paper exercise containing a list of different email tasks to perform e.g. sending a copy of a technical drawing to a group of people to study before a meeting. The students have to decide and justify which facilities would be used. Verbal debrief on the above exercise.	Include: • composing • responding • filing • copying • attaching • sending on • multiple recipients The practical use of electronic mail would be useful.

Session Plan 118 – Hypertext Linking Systems, Confidentiality, Encryption and Authentication

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.10**	3.10.3 Use of networks to support hyperlinking systems such as the world wide web (WWW)	• describe the purpose of hypertext linking, identifying the means by which it can be achieved such as hotwords/links, buttons and hypertext mark-up language (HTML) • describe the basic features of mark-up languages

Classroom Exercises	Notes
Demonstrate the features of a basic web page including the following: • use of tags (including on/off pairing, e.g. <HTML>…..</HTML>, <HEAD>…..</HEAD>, <BODY>….</BODY> etc) • links (within a page, to another page, to another web site) • methods of linking (hotwords, buttons, hotspots etc.) • basic formatting (centring, text size, colours etc.) Show the students a copy of the HTML code for this page and explain how each term relates to the web page. Set a practical exercise for the students to use a text editor to build a simple web page.	Practical preparation of a simple web page using HTML using notepad or similar may reinforce the concepts.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.10**	3.10.5 Issues of confidentiality	• discuss the problem of maintaining confidentiality of data on an open network and how to address this problem

Classroom Exercises

Discuss the problems of ensuring the confidentiality of data as it is being transferred across and stored at nodes on an open network, where coding and transmission methods are freely available. Include the following ideas in your discussion:

- prevention of access to data when stored e.g. physical security, use of access levels and passwords
- protection of data, from malicious interference, during transmission e.g. use of encryption (including public and private keys), screening of cables, problems with radio transmission, benefits of packet switching etc.
- ensuring that information is from a trusted source e.g. use of digital certificates to verify the authenticity of the message sender and provide the receiver with the means to encode a reply
- use of authorisation techniques to ensure that confidential information only reaches the intended recipient e.g. use of passwords, responses to special questions, provision of memorable data etc.

Student centred exercise using worksheets to reinforce / test knowledge – perhaps providing simple scenarios for the students to advise on how to ensure confidentially in a variety of circumstances e.g. transfer of highly confidential information from one government department to another, shopping on the Internet etc. Revise the answers to the worksheet as a class discussion to reinforce the concepts studied.

Assessment Objectives	Performance Criteria	Classroom Ideas
• **3.10**	3.10.6 Encryption and authentication techniques	• explain the need for encryption, authorisation and authentication techniques

Notes

- students will not be expected to know any specific method in detail