The International School Seychelles

IGCSE Computer Studies

CIE 0420

Theory Notes

Contents

1 Applications Of Computers, Their Social & Economic Implications

1.1 The Range And Scope Of Computer Applications

General Application Areas

The range of computer applications is vast. You should have an awareness of as many applications as possible, and for each, try to know the following:

1 The **purpose** of the application.
2 The required **outcome**.
3 The overall **system design**, including both the computerised and the non-computerised parts of the application.
4 The necessary **inputs** to the system and the means by which any data is captured.
5 The overall **organisation and processing of the data** within the system.
6 The use and **organisation** of the major **software** and **hardware** components of the system.
7 The need for recovery in the event of a **system failure**.
8 The **interface** between the system and its users.
9 The **effectiveness** of the system in practice.
10 The **effects** of the application on individuals and organisations.

Communication And Information Systems

Data is exchanged between computers. The data can be in the form of text, images, sound or video.

Electronic mail (e-mail)	simple message exchange. Can attach other files
Electronic conferencing	can either be a text, voice or video conference

On-Line Services & Remote Databases

Computers are often placed in locations where the public can use them to lookup information.

Information retrieval	online databases, encyclopaedias, dictionaries, bus timetables, etc.
Library systems	documents are stored digitally where they can be searched
Multimedia systems	stored information contains text, sound, video, etc.

Commercial And General Data Processing

These are often batch-processing systems involving data being collected over a period of time, and then processed later. The collected data is stored in a transaction file, and this is used to update the master file to give a new master file.

Banking systems	bank accounts are updated based on daily transactions
Personnel records	employee's pay is calculated based on hours worked
Stock control	when items are sent / received, the stock is updated
Order processing	received orders are dealt with

Industrial, Technical And Scientific Uses

Data is entered into the computer and the computer can then manipulate the data. In the case of CAD systems, products can be designed and tested completely within the computer's memory. In forecasting systems, predictions are made about the future based on previous data, and a model of how the system works.

Weather Forecasting	try to predict weather, taking data from **sensors**, using computer **models** of weather systems
Computer-aided design	(CAD) designs for objects can be designed, altered and tested within the computer, prior to manufacture

Monitoring And Control Systems

A computer is used to monitor a system through the use of sensors (such as light, heat, etc.) The computer can then operate devices to control the system (such as pumps, valves, etc.)

Monitoring hospital patients	checks heart rate, etc. setting off an alarm if low
Nuclear power station control	monitors temperature, etc. and adjusts coolant
Traffic survey and control	checks the number of cars, alters traffic light timings

Automation And Robotics

The use of computers to control other mechanical devices. Usually requires some sort of interface which allows the computer to receive data from input sensors, and to control output devices such as motors.

Domestic automation	washing machines, microwaves
Automatic navigation	aircraft, ships, cars (GSM satellite navigation)
Industrial robots	used to work on manufacturing lines. Can work non-stop, 24 hours a day, 365 days a year. Can work in hazardous areas

Expert Systems And Artificial Intelligence

Artificial Intelligence (AI) is the attempt to simulate the human brain, and its thought processes, using computer hardware and software. Expert systems use AI techniques to replace a human expert. All of the human's knowledge on a subject is entered into the computer as a series of rules.

Medical diagnosis	data is fed into the system, and questions are answered. The expert system rules then come up with the best diagnosis (usually with a % confidence level)
Speech recognition	checks voice patterns to determine what was spoken

Miscellaneous Areas

Some other applications of computers are:

Computer-aided learning	(CAL) CD-ROMs, etc. where you can learn at your own pace, like having your own personal tutor
Computer animation	for films and TV. Special effects, etc.

1.2 The Social And Economic Implications Of The Use Of Computers

Social And Economic Effects

Effects on people, organisations and on society in general.

Redundancies	jobs lost when staff replaced by computer-based systems
De-skilling	replacement of skilled staff by computers. Staff are then left to do less skilled jobs
Electronic 'scabbing'	if staff are striking, work can easily switched to non-striking staff via a network, even in a different country
Tactical striking	unions maximise impact of strikes by selecting computing staff first. Whole company is then affected
'New tech,' agreements	benefits to workers (cleaner, safer workplace) and management (more cost-effective) of using computer-based systems

Economic Reasons For The Use Of Computers

There are many financial reasons for using computers, and computer controlled systems, even though they are expensive to set-up initially.

More efficient	tasks completed quicker, and with lest wastage
Work longer than people	automated systems need no rest, can work for 24 hours a day
Save on wages	computers can often do the work of several people, so people are made redundant

Changes To Existing Methods, Products And Services

Businesses change the way they work, and provide different, and better services to clients.

On-line banking	more convenient for customer, cheaper for banks, less real staff required to deal with cash
E-commerce	selling goods online means less overheads, so better prices for customers, and more profit for company
EPOS	(Electronic Point of Sale) tills in shops where the good purchased are automatically taken from the database of stock. Can automatically generate orders for new stock
EFT	(Electronic Funds Transfer) allows people to pay for goods using a card which is 'swiped' in the store, authorising the transfer of money from the customer's bank account

Development Of New Products And Services

Computers have lead to the development of new markets and businesses.

- Internet service providers
- Web-design services
- Internet cafes

Changes In The Working Environment

Using computers within businesses has altered the environment that we work in

Cleaner and safer	dangerous / messy jobs done by computer systems
Work injuries can go up	due to prolonged computer use – RSI, back ache, etc.

Changes In Employment

The use of computers in the workplace has an impact on the way people work

Retraining of staff	software packages upgraded, staff need to be trained. This is often a regular thing. However…
Individual training	training can be personalised to staff's needs through the use of CAL systems, CD-ROMs, etc.

Privacy And Integrity Of Data

So much personal data is stored within computer systems that companies and governments have to have guidelines and laws to protect the privacy of people and their information.

Data Protection Legislation

The Data Protection Act gives the following requirements to anyone who stores data about someone else:

- Person must give **permission** for data to be stored
- Data must not be used for purposes **other than those it was given for**
- Must not store **more data than is necessary** for the purpose
- Data must be kept **up-to-date**
- Data should not be kept for **longer than is necessary**
- Data must be **protected** against unlawful access, or accidental loss
- Data must not be transferred **outside of EU** unless the country also has data laws

Security And Reliability

Data must be protected. This involves a number of procedures.

Back-ups of critical data	should be performed on a **regular** basis. A full copy of all files is taken (can use CD-R, magnetic tape, etc.) Usual technique is to keep **3 generations** of back-ups (today's, yesterday's and the day before's). This is the grandfather, father, son system.
Batch-Processing	Backup of **both transaction files and master files** required. If today's master file is lost, we can re-build it from yesterday's backed-up master file and transaction file.
Archiving of old data	to **reduce** the amount of **system resources** required (disk space, etc.) and keep the system running smoothly, **old data** that is no longer used (but may be required for future reference) is moved into an archive file.

Consequences Of System Failure

The requirements for security and reliability vary considerably depending on the nature of the application. For example, a failure during a batch update of a sequential master file is irritating and will cause delay, whereas a failure in an air traffic control system could well have catastrophic results.

Safety-critical systems (such as air-traffic control, or medical systems) have much more emphasis (and money) placed on reliability.

Hacking And Other Computer Crime

Computer crime includes activities such as the cracking of ineffective security systems to gain unauthorised access to commercially sensitive or confidential personal files, and fraud through the improper transfer of funds from one account to another. Computer criminals may work within the organisation or may be outsiders. Precautions can be taken.

Physical security	only allow authorised users near the computers
Complex security codes	use of good passwords (not just 'Fred' or 'Password')
Firewall soft/hardware	prevents access from outside the network
Encryption of data	can only be read by someone with the password
Monitoring of all access	trace users who are accessing and misusing system

Computer viruses

Sensible precautions should be taken.

- Up-to-date **virus protection**
- Limited use of disks, etc from **outside** your network
- **Firewall** software / hardware – prevents viruses using your Internet connection

2 System Analysis

2.1 Systems Analysis

Systems Analysis describes the process followed when replacing or upgrading a system with a computer-based one. The steps followed are:

- Fact-finding
- Feasibility study
- Analysis
- System design
- Implementation
- Testing
- Documentation
- Evaluation

Identification Of The Problem (Fact-Finding)

What exactly is/are the problem(s) that needs to be solved? How do we find out as much as possible about the present system? We need to do some fact-finding:

- **Interviewing** people who work for the company
- **Questionnaires**
- **Observation** of people at work within the company
- Inspecting **documents** that are used within the present system

Deciding And Stating Specific Desired Outcomes (Feasibility Study)

Before continuing with solving the problem identified, it is important that it will be worthwhile. The feasibility Study, and the resulting Feasibility Report, will give an idea of:

- A brief **description** of the business and the sections which will be affected
- A description of what the system is required to do (**objectives**)
- Some early system **designs** to allow estimations of time and cost
- Details of why some designs were **chosen**, and some **rejected**
- Approximately **how long** the project will take
- Approximately how much it will **cost**
- **Cost / benefit** analysis detailing whether the solution will save money or not
- **Plan** of further action
- **Conclusions** as to whether to continue, and if so, how

Based on the contents of the study and report, the company will decide if they are going to proceed with the project.

Analysing The Flow Of Information And Data In Existing Solutions and Evaluation Of Existing Solutions

If the project is to continue, then a much more detailed look at the business and its systems is required. Data will be collected and analysed thoroughly to aid in the design of a replacement system. In particular this stage will provide:

- Charts and **diagrams** of the present system, and the proposed new system (structure diagrams, data-flow diagrams, etc.)
- Detailed **objectives** for the new system

- Details of exactly which parts of the **old system** will be replaced
- **Constraints** of the new system (things it will not be able to do)
- An updated **cost / benefit** analysis
- A new **plan** for further action (including timing, responsibilities, etc.)

Consideration Of Alternative Solutions

Designs will need to be produced including:

- Required **outputs** (documents, reports, etc.)
- Required **inputs** (data, method of entry, validation, etc.)
- **File / database** design (number of tables, links between them, etc.)
- **Hardware** configuration (computers, networks, printers, etc.)
- **Software** required (operating systems, application packages, custom written, etc.)

A solution will need to be selected, based on the objectives that were set: Which solution fulfils all of the objectives?

Implementation

This can be done in one of three ways:

Direct implementation	the new system simply replaces the old one. Only suitable for small systems
Phased implementation	parts of the old system are slowly replaced by the new one, taking time to make sure each bit works before the next is begun
Parallel running	the new system runs alongside the old one to check it works as well. When all problems have been solved, the old one stops.

Testing

We must ensure that any new system functions as expected, both in normal use, and when potential problems arise. We test the system with:

Normal data	to see if correct, expected results are achieved
Data with **errors**	to check that our validation and error-checking routines
Large amounts of data	to ensure that the system can cope with high loading
Rare / uncommon events	like yearly reports, just to see that they work as expected
Extreme data	to make sure that validation works, and calculations cope with the large / small values

Documentation

There are two main type of document that are required:

User documentation	explaining how to use the new system (a user guide)
Technical documentation	to allow people to alter / upgrade the system in the future

Evaluation

Finally, the new system has to be assessed. Did it **meet its objectives**? Ask the new users!

3 Problem Solution, Algorithm Design & Programming Concepts

3.1 Making An Overall Plan

Defining The Scope Of Separate Modules

Software problems should always be broken down into small modules of code (subroutines). These modules should be small and as self-sufficient as possible as:

- The module is **easy to produce**, not being dependent upon other modules
- The module is **easy to test** for the same reason
- The code is **re-usable** - the same module can be used in another project

e.g. Software for teaching children about shapes could be broken down into the following modules:

- One that deals with drawing shapes (further broken down into shape modules)
- One that deals with asking questions about shapes, and uses the drawing modules
- One that saves and loads the student's progress to / from disk

Each module has a clear purpose and it's dependency on other modules is clear. If the modules are well defined, then the overall problem becomes easier to solve.

Designing Algorithms

When writing algorithms to solve software problem, you must always bear in mind that your solution must satisfy the requirements of the system. If the software is required to calculate the average scores of 1000 students, then that is what your algorithms must do.

Explaining Algorithms And How They Relate To The System

The design of any algorithm should be driven by:

Inputs supplied	what data will be passed to the algorithm to work with?
Outputs required	what data is expected from the algorithm?

The inputs and outputs will be defined by the algorithm's location within the overall system, and it's connections to other modules.

e.g. If an algorithm is required to draw a square on the screen, then we need to define how it will be used. We want to use it several times for different sized squares, in different places on the screen, so we'll need to give it some sizes and a location:

- Inputs: side length l, and position of top-left corner (x, y)
- Outputs: a square! Sides l, located at (x, y)

This algorithm can now be easily used to draw any square, anywhere, as long as we supply it with the correct data.

The actual algorithm produced is almost irrelevant as long at it produces the desired results, from the inputs supplied (there are issues of speed, good programming practice, etc. to be considered though)

Explaining How Hardware Needs Arise From The Output Required

Any software system will need some hardware to run on. The configuration of the hardware is dependent upon:

Outputs required	items like printers, monitors, loudspeakers, etc. will be required depending on the output of the system
Processing needs	a modern computer will have a processor running at about 3GHz. That's a lot of processing power! If less speed is required, money can be saved
Environment	Certain items of hardware are suited to certain workplaces (e.g. extra-rugged keyboards for factories)
Space available	some hardware (e.g. flat-screen monitors) is very space efficient
Costs	hardware costs very greatly. A CRT monitor is much cheaper than a flat-screen one.

Algorithm Tools

Top-down design	a software problem is solved by **breaking it down** into smaller and smaller modules, until each of these is easy to solve, and write software for. The final solution is formed by joining the modules together. Usually represented using a structure diagram.
Structure diagrams	(see Section 3.2 below)
Flowcharts	(see Section 3.2 below)
Menus	these are an effective way of breaking up a software task. The user has menu options, each of which is handled by a separate bit of the software (like the top-down structure)
Libraries of subroutines	Many common software problems have been solved already, and you can buy in libraries of software modules that can be 'glued' together to form an application

Interpreting And Testing Algorithms

You should be able to work out the purpose of an algorithm and to suggest and apply suitable test data.

Valid data	data that tests the **normal** operation of the algorithm (e.g. if entering test scores, your test data could include 0%, 20%, 56%, 89%, 100%)
Extreme data	including values that are at the **limits** of the expected input (e.g. 0% or 100% for exam result, or a text string that is the
Abnormal data	data that may be valid, but is **not usual** (e.g. all test scores 0%, or all 100%, or no test scores at all)
Invalid data	this includes data that is not of the **correct type** or format (e.g. text into a number field)

3.2 Programming

The Concept Of A Program

A computer program is a collection of commands that the computer is to follow. The main requirements of a programming language are:

- To allow **manipulation of data** of various types and structures
- The **control of inputs and outputs**
- To provide for **selection** and **repetition** (see below)
- To allow **links between subprograms**

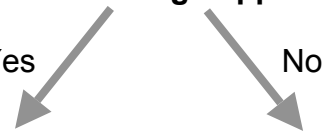
Sequence Following one command by another by another, etc.

i.e. **Instruction 1**
Instruction 2
Instruction 3
etc…

the computer executes each instruction in order

Selection The computer is given a question as then performs different tasks based on the answer

i.e. **Has something happened?**



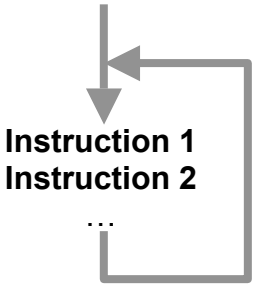
Instruction 1
Instruction 2
etc…

Instruction 1
Instruction 2
etc…

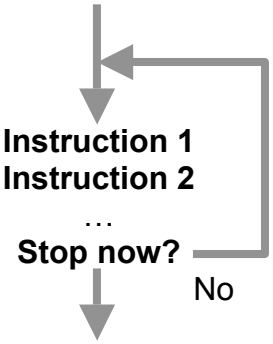
the instructions that are executed depend answer to the question

Repetition Following the same group of commands over and over again (loop)
The loop may go on forever (infinite loop) or may stop due to a certain condition (conditional loop)

i.e.



an infinite loop



a conditional loop

Low-Level Languages

Low level languages are close to the actual language that computers use (machine code)

- **Hard to program** in
- Most common is **assembly** language
- An **assembler** converts assembly language into machine code

High-Level Languages

These are languages that are **closer to human language** than computer.

- There are many different types, some dedicated to specific applications (e.g. Fortran is used for mathematics, Cobol is used for business databases)
- Much **easier to use** than low level languages
- Very easy to create the programming **structures** such as loops
- Lots of **easy-to-use tools** to help debug programs
- Require either an **interpreter** or a **compiler** to convert language into machine code

Assemblers

Convert low-level languages into machine code. This is called assembly.

Interpreters

Convert high-level languages into machine code

- Convert **one instruction at a time**, executing it **immediately**
- To run programs that use an interpreter, requires the user to have an interpreter themselves
- The final program runs more **slowly** than a compiled program

Compilers

Convert high-level languages into machine code

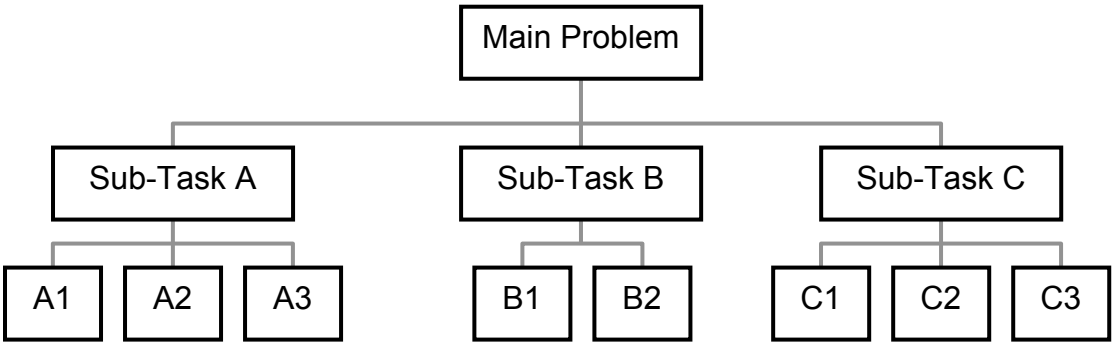
- Convert the **whole program** in one go
- The program cannot be run until it is fully compiled
- Once a program is compiled, the user does not need a compiler to run it (the program is **'stand-alone'**
- The resulting programs run very **quickly** compared to interpreted programs

Structure Diagrams

These diagrams help us design software, by letting us break the task down.

- The programming problem is **broken down** into a number of **sub-tasks**
- These sub-tasks can also be broken down into **smaller tasks**
- And so on…

The smaller tasks are then simple to solve, and when they are all joined together, we should end up with a solution to the overall problem.



Variables

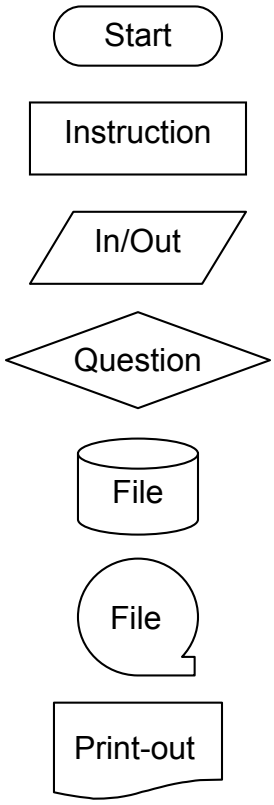
Within any computer program, data needs to be stored whilst it is being processed. We store it in special locations in memory called variables (their contents can change, so they are 'variable'!)

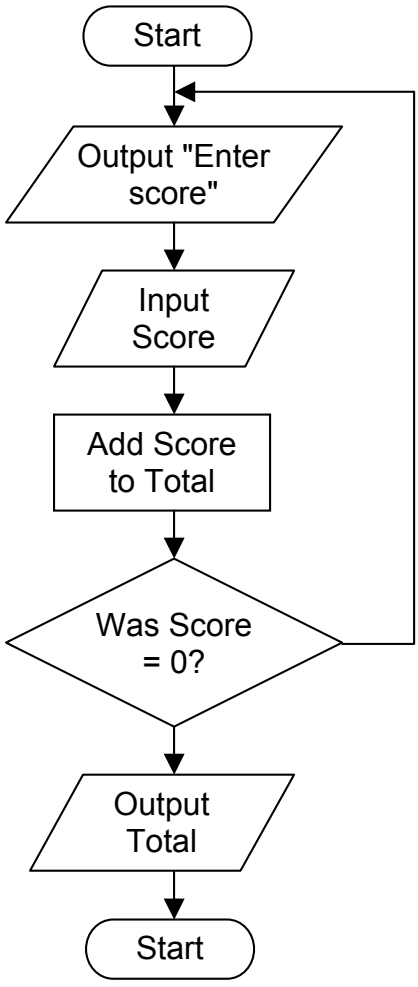
- Variables have **names** – pick names based on what you want to store (e.g. 'score')
- They can store **different types** of data: numbers, text ('strings'), etc.
- You can use variables within calculations (e.g. 'total = score1 + score 2')
- The content of a variable can be set (e.g. 'Set score to 0', or 'score = 0')

Flowcharts

These diagrams show the flow of our program (the order that instructions are executed)

- The **start and end** of the program is shown as a 'sausage'
- Any **tasks** that are just simply executed are shown in rectangles
- **Inputs and outputs** are shown in parallelograms
- Places where a **decision** is made (**selection**) are shown using diamonds
- Places where **files** are saved to or read from disk are shown using cylinders
- When files are **master files or transaction files**, they are shown as reels of tape
- Places where **print-outs** are generated are shown as a piece of paper
- **Arrows** join the symbols together





Pseudocode

This is a way of describing an algorithm (or program) using words. It is an alternative to a flowchart, and is often easier to write, and more understandable, since it is close to English.

- Words are **close to English**
- Uses **some standard words** for program structures such as loops (see below)
- **Indentation** is used to highlight the structure of the code
- Inputs and outputs are shown using the words Input (data) and Output (data)
- Mathematical operations can be written in words (e.g. 'Add 6 to score')

Pseudocode Structures

There are several standard pseudocode structures that are commonly used:

Input (variable) — Gets data from the user and stores it in the variable

Output (variable / "text") — Prints the contents of the variable, or the text on the screen for the user to see

Repeat ... Until — Repeats the instructions between the 'Repeat' and 'Until' lines until a condition is met
e.g.

```
Repeat
    Output ( "Please enter a score:" )
    Input ( score )
Until score equals 0
```

If ... Then ... Else ... Endif — Allows for selection or decision-making. A condition is tested and if it is true then one set of commands is executed, else if it is false another set is executed.
e.g.

```
If Score is greater than 60% Then
    Output "You have passed. Well done!"
Else
    Ouput "You have failed. Sorry!"
Endif
```

Case of ... Otherwise ... Endcase — If you want to have different things happening depending on the value of a variable.
e.g.

```
Case of
score is greater than 70
    Output ( "You have an A" )
score is greater than 60
    Ouptut ( "You have a B" )
score is greater than 50
    Output ( "You have a C" )
Otherwise
    Output ( "You have failed" )
Endcase
```

While ... Do ... Endwhile — This is another form of loop (like Repeat…Until) except that this one keep going whilst a condition is **true**, and the condition is checked **before** the loop starts
e.g.

```
Set count to 10
While count is greater than 0 Do
    Output ( count )
    Take 1 away from count
Endwhile
Output ( "Blast off!!!" )
```

User Documentation

Any piece of software requires some kind of user documentation.

- Instructions on how to **set-up** and **install** the software
- Instructions on **using** the software (User Guide)
- **Troubleshooting** tips (for when things don't work as expected)

Technical Documentation

This is required by those responsible for improving and maintaining a software solution in working order, or for developing the solution to meet new needs.

- A guide showing the **structure** of the software (using structure diagrams for example)
- What **inputs** are expected
- What **outputs** are generated
- What **algorithms** are used to process data, etc. (using pseudocode or flowcharts)
- Any actual code should contain **comments**. These are notes explaining what a particular set of instructions is supposed to do.

4 Generic Software And The Organisation Of Data

4.1 Generic Software

Word-processing (e.g. Word)

- Letters, etc. can be typed
- Full control over **font, font size, styles**, etc.
- Most WP applications also include some **graphics** and DTP functions
- Has resulted in many people doing their own typing, making office staff redundant

Database management (e.g. Access)

- From simple databases similar to card files (**flat databases**) through to complex databases, with many groups of data linked together (**relational databases**)
- A database consists of a collection of **records** which in turn are made up of **fields**
- Database can be interrogated using **queries**
- Can easily generate **reports** containing selected data

Spreadsheets (e.g. Excel)

- Made up of individual **cells** which can contain data (number, text, etc.) or formulae
- Allows complex and powerful **calculations** on collections of data
- Can experiment with data values and see the results (**'what-if' analysis**)
- A mathematical **'model'** of a situation can be produced (e.g. company future profit)
- **Rows** are numbered, **columns** are lettered, cells by letter/number (e.g. A3)
- Formulae all begin with **'='**
- Many built-in **functions** such as SUM, AVERAGE, etc.
- Numbers can be **formatted** as currency, percentage, etc. (e.g. a cell containing 0.5 could be shown as $0.50, 0.5000, 5e-1, 50%)

Graphics (e.g. Photoshop)

- Images can be produced from scratch, or **photos** manipulated
- **Effects** can be added (e.g. blurring, colour alterations, etc.)
- Input can be from a **graphics tablet** for more natural lines, etc.
- Images can be **exported** into many file formats (e.g. GIF, JPEG, TIFF, etc.)

Communications (e.g. E-Mail, Internet)

- **Very quick** delivery of messages
- Used both across the Internet, and within businesses (internal e-mail)
- Normal e-mail is **text-based**, but other files can be **'attached'**
- People now communicate more **regularly** since it is so easy, but are possibly **less sociable** since they now use the phone / meet less often (less human contact)

Multimedia Authoring

- The production of CD-ROM / websites that **include text, images, video, sound**, etc.
- Common applications are **education, CAL, reference** (encyclopaedia, etc.)

Data-logging

- A computer uses **sensors** to detect events, etc.
- Examples could be measuring traffic flow, science experiments, etc.
- Data is more **accurate** than that taken by a person, and can be **analysed instantly** (graphs, etc.)

CAD

- Products are **designed on the screen** rather than on paper
- Much faster, and far **more flexible** than drafting designs by hand
- The products sizes (dimensions) can be altered easily (no rubbing out!)
- The product can be **viewed from different angles** with lighting effects, etc.
- Some CAD systems allow products to be **tested** using modelling techniques, so that weaknesses can be found before anything is actually physically manufactured

Programming

- **Software** can be produced using **programming languages**
- Many modern programming languages are now so easy to use that applications can be produced in minutes

Desktop Publishing (e.g. Publisher)

- Allows the user to **layout text, images, etc.** on the page with total **flexibility**
- Items are placed in **'frames'** which can be layered, moved, re-shaped, etc.
- Authors can produce books that are fully laid out without the need for publishers
- Anyone can easily make flyers, posters, etc.
- Used to produce printed matter from labels to magazines
- Has made **publishers almost redundant** as designers can now do every stage of the work, passing on the final files directly to the printers

Typical Problems Capable Of Solution By Packages

Many problems can be solved using generic software packages:

Small businesses	accounts can be calculated using a spreadsheet, letters written using a word-processor, stock details kept using a database
Schools / charities	don't need to employ expensive consultants to produce professional looking reports, posters, flyers, etc.

Use Of Standard Techniques Or Routines

Cut / copy / paste	data can be easily moved between applications
Mail-merge	a **standard letter** can be sent to a **number of people** by merging a database of names and address with the letter template. Special codes in the template are replaced each time by the appropriate name / address
Sorting of data	a list of items (e.g. names) can easily be sorted into order
Import / export of data	Applications can save data in a format that is suitable for another application (export), or can open files that they wouldn't normally use (import)

Spell / grammar check	within many applications, the computer will automatically check and correct your spelling and grammar
Wizards	Many complex tasks are made extremely easy for novices through the use of wizards. The user is asked a **series of questions** and their answers result in an appropriate complex set of actions (e.g. producing a poster)
Macros	When a number of operations need to be **repeated over and over again**, they can be 'recorded' and stored in a macro. This macro can be 'replayed' any number of times

Integrated Packages Versus Custom-Written Software

Integrated packages are completely general-purpose, so they are:

- **Cheap** to buy (many copies will be sold, so the unit cost is low)
- **Well documented** (since many people will use them – lots of training courses)
- **Stable** (mostly bug-free) because of the large user-base
- **Not always suitable** for every business's needs, unlike custom-written software

Some business's needs cannot be satisfied by generic software, and custom-written applications are necessary. These are:

- Very **expensive** since they are usually one-offs
- **Perfectly suited** to the business's needs
- **Easy to learn** since they can be exactly matched to the skills of the workforce
- Sometime quite **unstable** (buggy) since they have only limited testing

4.2 Data

The Relationship Between Information And Data

Information is data that has been processed.

e.g. 234567 is simply data (a number) but it could mean many things – a membership number, a mathematical result, etc. Once it is given this meaning, it becomes information.

The Collection Of Data

There are many methods for collecting data for computer systems:

Questionnaires	These can be closely matched to the on-screen computer form so the data from them is easily transferred, or…
OMR forms	where **marks are made with a pencil** (e.g. multiple choice test) and read into a computer by a special reader
Bar-codes	Bar-code readers can scan the codes in very rapidly, e.g. at a till in a store, or from books in a library
Scanners	Paper documents can be scanned into a computer. They become large, **graphic images** which can then be…
OCR of documents	(Optical Character Recognition) after text documents are **scanned**, the image file can be processed by software that recognises the shapes of the characters. Output is a **text file** containing the original text. Not always 100% right
Sensors	these can record things such as **temperature, light levels, sound, etc.** and the readings can be entered into a computer for **logging** and **processing**

Methods Of Ensuring Data Correctness

Data can be checked in two ways: validation (is it valid and logical?) and verification (is it correct?)

Data Validation

To ensure that the data being processed is accurate it must be checked to see that it makes sense. It is asking too much of a computer to check if it's true - somebody may just be lying! Certain validation checks can be carried out:

Range-check	e.g. a driver's age should be between 17 and 99
Type-check	e.g. a name should only contain alphabetic characters
Format-check	e.g. a data should be entered as dd/mm/yyyy
Presence-check	e.g. certain data must be entered (e-mail address)
Checksum	e.g. the numeric values in a file should always add up to a known amount
Check-digit	e.g. the last digit of an account number is calculated from the previous digits, and then recalculated on input to see if it matches

Data Verification

Once we know our data is valid (i.e. it is logical, and in the right format, etc.) then we also need to check if it is correct (it maybe a valid date-of-birth, but is it *your* date-of-birth?!)

Proof-reading	the person entering the data simply **reads through** what they have entered to check it is correct
Double-entry	the **data is entered twice**, either by the same, or another operator. The computer **compares** the data, and any that doesn't match is questioned.

The Coding Of Data For Input

Before data is entered into a computer system, it needs to be put into a suitable format:

- Dates should all be of the **same format** (e.g. dd/mm/yy or dd/mm/yyyy)
- Text that is to be entered into a database field may be limited to a certain **number of characters** (if your paper form is designed well with little squares for each letter so the user knows how much space they have, this will be easy)
- Data from **sensors** needs to be converted into **digital** data for the computer

Storing Data In Files

Data, particularly data held in files, requires access in different ways depending on the particular application. Some applications require the data to be ordered in a particular way for processing, whilst some need the data to be very easily searchable for rapid access to specific items

The medium on which the data is stored, and particularly the way in which it is organised, depend on the requirements for access. You would not store a file on magnetic tape, if you needed quick access to any part of the data – it would take too long to find items.

Sequential access	is the storage of data in a way that you can only read **one item at a time** (like recording a film on video tape – you have to work your way through the film from the start) A **magnetic tape** is a sequential access device. You read it from the start until you find the data you want, re-winding and starting again if you need to do another search
Sequential indexed	the file is still accessed sequentially, but **each item of data has an index** (a number or code to identify it). This index can be used to search for data more rapidly
Random access	any item of data can be **accessed immediately**. A hard drive is a random access device – you do not need to read all of the data from the start to get to a particular file, you simply jump straight to it.

Some common file processing:

Sorting	the data on the file is **arranged into a logical order** (perhaps by numeric index, or alphabetically)
Merging	two files are **combined**, creating a new file. The data is often sorted at the same time so the two sets are mixed in the **correct order**

Updating	when data records in a file are **altered** because the data has changed, the file is said to be updated
Insertion	if **new data** is to be **added** in the correct place in a sequential file, the file is first searched to find the right location and the data added. The remaining data must be moved to make room
Deletion	when an item of data needs to be **removed**, the sequential file is searched until the item is found, it is removed and the remaining data is moved to fill in the gap left behind

The Presentation Of Useful Information From Processed Data

There are many ways in which processed data can be presented so that it may be better understood:

Graphs and charts	numeric data is presented **visually** as bar charts, line charts, etc.
Images	data can be converted into an image, for example weather information from sensors can be shown as lines / colours on a map to help aid weather prediction

Analogue-To-Digital And Digital-To-Analogue Conversions

There are two categories of data:

Analogue data	this is **constantly varying data**, such as temperature, time, etc.
Digital data	this is **discrete data** that has **exact values**, such as exam grades (A, or B, or C… There is no level in-between

Analogue data can be made digital if it is **sampled**. This involves measuring the level at a particular moment in time

Since computers are digital devices, whilst many other devices are analogue, we need to be able to convert analogue data into digital for input into a computer, and back again.

Analogue to Digital Converter (ADC)

The analogue signal is sampled at regular time intervals to give a sequence of digital values. An example:

- Temperature sensor supplies an analogue voltage dependent upon the temperature. This voltage passes through an ADC to become binary (digital) values

Digital to Analogue Converter (DAC)

The computer's digital data (binary numbers) is converted into an analogue signal. An example:

- Sounds for a computer game are generated as digital data. These are passed to a DAC which produces an analogue signal which can drive a loudspeaker (all sound cards in PCs are essentially DAC devices)

Data Types

Data can take many types including:

Numbers	these can be integers (whole numbers) or real numbers (with fractional parts). Numbers can be formatted to be displayed as currency, dates, etc.
Characters	are the symbols on the keyboard, including letters, numbers, punctuation, etc. Characters can be joined together to make…
Strings	a load of characters joined together (e.g. "Steve", "HHGF45sju")
Arrays	are a collection of a particular data type. So an array of integers would be a whole set of numbers stored together. Arrays have a name, and the individual parts are numbered (e.g. Scores(3) is the third item in the 'Score' array)

The Need For Different Data Types And Structures

As part of our system design, we need to specify what types of data are to be used to store information.

e.g.

A person's **name**	stored in a **string** (as it's a collection of characters)
A person's **age**	stored in an **integer** (since always whole number of years)
A person's **height** in metres	stored in a **real** number (since will have a decimal part)
A **collection** of **temperature** values	stored in an **array of real numbers** (since can be fractions of a degree)

5 Hardware, Systems And Communications

5.1 Hardware

Computer

Comprised of a number of components (parts):

Central Processing Unit (CPU) the 'brains' of the computer, comprises

- **Arithmetic and Logic Unit** (ALU) which processes instructions and performs calculations
- **Control Unit** (CU) which co-ordinates the operation of the different parts of the computer
- **Primary memory** (RAM) which holds data that is currently being used.

Peripheral devices all of the devices that the computer interacts with, and that make it useful

- **Input devices** which get data into the computer (e.g. a keyboard)
- **Output devices** which convert data from the computer into other forms (e.g. a printer)

Backing storage devices that allow us to store data that is not currently being processed, or which needs to be saved when the power is turned off (non-volatile storage)

- **Magnetic storage** such as hard drives, floppy disks, magnetic tape
- **Optical storage** such as CD-ROMs, DVD-ROMs, etc.
- **Memory cards** such as the ones found in digital cameras. Use a form of ROM that can be changed electronically. These are replacing floppy disks

Microcomputer

The original digital computers from the 1940s were huge beasts that took up whole rooms, or even buildings. Since then, computers have got a lot smaller, a lot more powerful and a lot more user-friendly.

Mainframes the original, large computers were called mainframes

Minicomputers the next generation of computers, and the first ones that were sold in large number to businesses

Microcomputer the small computer that you will be used to using, one that is self-contained and sits on your desk

Laptop a portable microcomputer

Palmtop a portable computer that is small enough to be hand-held

PDA **(**Personal Digital Assistant) tiny computer that usually has no keyboard, using a touch screen instead

Microprocessor (Embedded Computer)

Not all computers need to have floppy drives, monitors, etc. Some are designed to do just one specific job and don't need to be able to load new software, etc. These computers are very small and are often on a single chip. Examples of devices where microprocessors are found are:

Digital watches	only task of the software is to tell the time. Inputs come from buttons, and the outputs are the display and a buzzer
Digital cameras	the tiny microprocessor takes inputs from sensors about the light level, etc. and then outputs control the shutter speed when the picture is taken.
Engine Management	in cars a microcomputer controls all of the engine's systems, taking a range of inputs from sensors around the car (speed, etc.) Outputs devices control the flow of petrol, etc.

Standard Input Devices

You need to be aware of a range of input devices, and the data they are provide, including:

- Keyboard
- Mouse and other pointing devices (graphics tablet, touchpad, touch screen, etc.)
- Microphone
- Joystick
- Barcode reader
- Scanner
- Sensors which detect or measure things (e.g. light sensors, heat sensors, etc.)

Standard Output Devices

You need to be aware of a range of output devices, including:

- Monitor
- Printer, including dot-matrix, inkjet, laser printer
- Plotter
- Loudspeakers
- Robot arms or other actuators used to perform tasks or control things

Processor Power

A processor deals with instructions one at a time. The quicker it can read and process instructions, the quicker tasks will be completed. Because of this, people are always keen to get quicker and quicker processors.

We measure **processor speed in Hertz (Hz)**, and this measures how many instruc**tions** a processor can handle **per second**. Most processors will handle **millions** or even **billions** of instructions per second, so you will see their speed quoted in megahertz (MHz, millions) or gigahertz (GHz, billions)

The Functions And Characteristics Of Storage Media

There are many ways we can store date. It depends on what we are doing, and how we need to be able to access the data. The size of storage is measured in bytes, kilobytes (kB, thousands of bytes) or megabytes (MB, millions of bytes).

RAM — this memory is **volatile** (loses it's contents when the power goes off) so only suitable for **temporary storage** whilst the computer is switched on. RAM is used to hold:

- Some of the **operating system** (e.g. Windows)
- The **applications** currently running (e.g. Word)
- **User data** (e.g. the letter you're typing)

The computer can access RAM extremely **quickly**. Commonly computers have about 256MB of RAM

ROM — this memory is **non-volatile** (keeps it's contents when the power goes off) so it is used to hold data that is permanently required. The main use is to hold the small **boot-up program** that a computer runs when first switched on.

Hard disk — a form of **magnetic storage** that can hold **huge amounts** of data (up to **500GB**, or 500,000,000,000 bytes of data!) Data can be accessed very rapidly (not as fast as RAM though) so it is good for most storage.

Users need larger and larger hard drives since they are starting to save digital images, music and even video, all of which take up a lot of space. For example:

▪ Digital photo, A4 size	500kB (0.5MB)
▪ 4 minute MP3 music track	4MB
▪ 10 minutes of a movie, full screen (VCD)	80MB

Floppy disk — a form of **magnetic storage** that is **removable**, so it can be used to transfer data between computers. Very old, crude technology so not very reliable. Can hold **1.44MB** of data (1,440,000 bytes)

Zip disk — similar to a floppy, but holds much more data (**20MB** or more)

Magnetic tape — a form of **magnetic storage** that is a little like a small video cassette. It can hold huge amounts of data (**24GB**), but the data can only be accessed sequentially (one bit after another) so tape is usually only used for **backing up** data.

CD-ROM — **optical storage** (read by a laser) that is commonly used to hold data such as software, reference data, etc. Normally holds abut **700MB** of data. CD-ROMs are **read-only**, so data cannot be saved onto them.

CD-R and RW — CDs that can be written to, so data can be **saved** onto them. CD-Rs can be used saved to once, CD-RWs can be saved to many times.

DVD-ROM — similar to a CD-ROM, but can hold up to **17GB** of data (25 x bigger)

Memory card — a type of **ROM** that can be **written to electronically**. These little devices are found in digital cameras, MP3 players, etc. and allow data to be easily moved about. They are replacing floppy disks. Still quite expensive. Come in capacities from **16MB up to 2GB**

The Characteristics And Performance Of A Range Of Peripherals

You should be able to discuss the suitability of different peripherals for various applications.

e.g.

Laser Printer	fast, quite, cheap to run suitable for an office environment
Bar-code reader	allows very rapid entry of data codes suitable for POS terminals in stores
Flat-screen monitor	although expensive, saves space, more energy efficient suitable for office desks and shops where space is limited
Touch-screen	very easy for novices to use, natural input, tough suitable for public information points (no mouse required)
Etc…	

5.2 Systems And Communications

Operating Systems

An operating system (OS) performs the following tasks:

- Acts are the **interface** between the computer's **hardware**, and the **application** software: Applications can run on a range of different hardware without needing any alterations (the OS 'hides' the differences
- Manages **system resources**, such as memory, CPU processing time, etc.
- Manages the **transfer of data** to / from **peripheral** devices
- Manages system **security** (login, file access, etc.)

Examples of common OSs are Windows 98, Windows XP, Mac OS, Unix, Linux, etc.

The Interface Between The Operating System And The User

Most computer OSs require user interaction at some point. This human / computer interface (HCI) usually takes one of three forms:

Command-line interface	The user **types in commands** to tell the computer what to do. Very **quick for experts**, but hard to learn for novices. Examples are MS-DOS, and basic UNIX
Menu interfaces	the user is presented with a **number of options**. They pick one, and another set of options appear, and so on until they find the operation / data that they want. Very **user-friendly**, even with a large number of options, and easy to add / remove options. Examples of a menu-driven interface are cash-point machines at banks, and mobile phones
Graphical user interface	(GUI) where computer operations and data is represented by pictures (**icons**); these are clicked by a **pointer** controlled by a pointing device (e.g. a mouse); information is shown in overlapping areas called **windows** and there is usually **menus** too. Often called a WIMP (Windows, Icons, Menus, Pointer). Very **easy to learn** for novices since it mimics the real world (desktop, waste bin, etc.) An example of a GUI is Windows

Management Of Files and File Directories

Files stored on the backing store of a computer include operating system files, programs (e.g. Word) and data (e.g. a letter that you typed). There are often 10,000s of files on a computer's hard drive, so they have to be kept **organised**:

- Usually they are grouped in **folders** (or **'directories'**)
- Folders can contain **sub-folders** allowing for a **hierarchy** of folders (a 'tree' structure). So a file could be stored in /My Letters/2003/January/
- Files should be **named** to make it clear what they are (e.g. "Letter to Mum.doc")

Files often need to be **moved, copied or deleted**. This can be accomplished through a GUI (e.g. in Windows you can delete a file by dragging it to the waste bin) or by typing commands (e.g. In Unix you type "rm filename", in MS-DOS you type "del filename").

Sometime you need to **see the contents** of a file, or to **print** it out. In a GUI you might drag the file to a suitable application (e.g. In windows you would use Notepad to view text files) or you may type a command (e.g. In Unix you would type "cat filename" to view it)

Peripheral Device Control

There must be some **central control** of peripherals so that data is passed between them without any problems (like data being lost)

Use Of Buffers

A buffer is a small arrear of memory that is used to temporarily hold data that is either being **sent to a peripheral** (e.g. a printer) or **coming from a peripheral** (e.g. a keyboard)

The buffer allows the computer (which is very fast) and the peripheral (which is often very slow) to **operate independently** without having to wait for each other.

For example, when printing, the OS places the print data in a buffer and the printer begins to print it. The OS does not have to wait about for the printer to finish. It can get on with more useful tasks.

Interrupts And Interrupt Priorities

This is one way that peripherals can **interact with the CPU**. If anything occurs that the computer needs to know about, the peripheral sends a control signal which **'interrupts'** the CPU (hence the name 'interrupt'). The CPU then deals with the peripheral before continuing normal operation.

Some interrupts have more importance than other. This is called the interrupt's **priority**. High priority interrupts get dealt with more rapidly by the CPU.

Polling

Another way of peripherals interacting with the CPU is for the CPU to regularly **'ask'** the peripheral if anything has happened that it needs to deal with. It goes around **each of its peripherals in turn**. This is called 'polling'.

Polling is much more **inefficient** than an interrupt system, as often nothing will have happened, so the CPU **wastes time** asking.

Handshaking

When **establishing a communication link**, two computers will **'greet'** each other, and **'introduce'** themselves. Once both computers are happy, and ready, communication can continue. This initial process is know as 'handshaking'

Check Sums

Adding together all of the elements (e.g. the bytes) of a **block of data** produces a single element known as the check sum. This can be stored with the data block and provides a **check** when the block is transferred. The receiver **re-calculates** the data's checksum and compares it with the one sent. If they differ, a communication error probably occurred.

5.3 Types Of System

You must be able to distinguish between the different types of system:

- What is needed to support them?
- Which is the most suitable for any given application
- Implications for the user

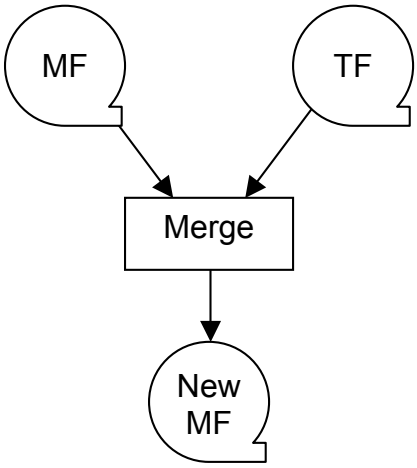
Batch Processing Systems

In broad terms, a batch processing system is one in which a job is processed **without any direct interaction** between the job and the user.

The operation is controlled by a Job Control Language (**JCL**)

Typical applications are payroll and billing systems.

A master file of data is updated using a transaction file that contains all of the changes that have occurred to the data since the last update. The result is a new master file.



Single-User Online Systems

Online systems provide for **interaction** between the job and the user, which may influence the future course of processing. e.g. word processing

Such systems may be single-user (for example, a personal computer)

Multi-User Online Systems

Multi-user online systems allow **several users** to interact with the computer at one time, often from completely different locations.

Typical applications are online information retrieval (e.g. flight booking in a travel agents)

Multi-Tasking Systems

If a user can run **several tasks at once** (concurrently) on the same computer, it is aid to be multi-tasking.

An example is when you have a word-processor, a spreadsheet and a CD player all running at once. The computer is multi-tasking.

Real-Time Systems

A real-time **transaction system** is an online system in which individual, discrete transactions are **processed as they occur**.

An airline booking system and an online stock control system are typical examples.

A real-time **control system** is a system in which physical quantities are **continuously** monitored and processed sufficiently rapidly to be capable of **influencing the sources** of data.

A chemical plant, or an autopilot system in an aircraft are examples.

Control Systems

In a control system, one or more computers is used to **control the operation** of some non-computer equipment, usually involving some **monitoring** and **logging** of physical quantities, providing some **analysis** of performance and allowing some **user interaction**.

Feedback is an essential element in most control systems. Timing considerations are often critical and the term real-time control system is sometimes used to indicate this.

Control systems are used in applications such as oil refining, chemical processing and integrated traffic-control systems.

Automated Systems

Automated systems are broadly similar to control systems but are **dedicated** to a **particular task**, and lack the ability to collect and analyse data and the flexibility to allow for and act on user interaction beyond a very simple level.

Examples are the systems found in equipment such as washing machines and cameras.

Multimedia Systems

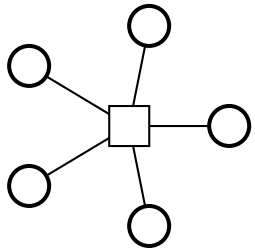
Candidates should be able to specify minimum hardware and software requirements for multimedia applications, and describe typical features and uses of multimedia systems.

Network Systems

A network system is one in which several computers are **joined together** by communication links, and can then **share resources** such as file storage, printers and information resources.

Local resources	hardware and software that reside on your computer
Remote resources	those items that you access via the network (e.g. a printer)

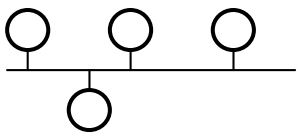
Star Networks



The computers are arranged around a **central hub**. This hub may be a computer or a dedicated piece of hardware.

Star networks are very **fast** as data between computers can always take the most **direct route**. They are also quite **secure** (safe from people seeing your data) as data is only read by the computers sending / receiving.

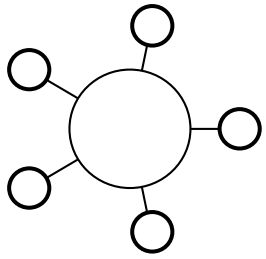
Linear (Bus) Networks



The computers connect to a long, **common cable**. All data is sent onto this cable, and each computer listens out for messages for itself.

Bus networks are **cheap to install**, and if a computer breaks, the rest of the network can still operate.

Ring Networks



Computers are connected to a **circular cable**. Data can go either way, so this network can cope better with **cable breakages** than other network types.