

GCSE Computing (OCR)

3rd edition

Susan Robson



PG ONLINE



GCSE Computing (OCR)

Computer Systems and Programming

Susan Robson B.Sc. (Hons)

Published by

PG Online Limited

The Old Coach House

35 Main Road

Tolpuddle

Dorset DT2 7EW

Email sales@pgonline.co.uk

Website www.pgonline.co.uk

GCSE Computing (OCR)

Acknowledgements

I would like to thank Pat Heathcote for her careful editing of the 3rd edition and for her valuable comments on some aspects of the text, which have resulted in a better book.

I am grateful to the OCR Examination Board for permission to use questions from past examination papers.

Design and artwork by Direction. www.direction123.com

Cover picture © 'Starting point' 2007 by Katherine Palmers-Needham

First edition published 2010

Second Edition published 2011

Third Edition 2014

A catalogue entry for this book is available from the British Library

ISBN 978-1-910523-00-1

© 2014 PG Online Limited

Please note that the contents of this downloadable book are protected under copyright law. It is supplied under licence and may be used and copied only in accordance with the terms of the licence. Except as permitted by the licence, no part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic or otherwise, without the prior written permission of PG Online or as strictly permitted by applicable copyright law.

Licence agreement

This is a legal agreement between you, the end user and PG Online Limited. The downloadable version of this book is licensed, not sold to you by PG Online Limited for use under the terms of the licence.

The contents may be freely copied and used by members of a single institution on a single site. It is not permitted to share any of the material with users on another site or belonging to a separate institution.

Preface

The aim of this book is to provide comprehensive yet concise coverage of all the topics covered in *Unit A451: Computer Systems and Programming* of the OCR GCSE Computing Specification J275, on which the written examination is based. It is written in clear and easily understandable English and will be invaluable not only as a course textbook but also as a revision guide.

Each of the seven chapters covers one of the topics in this unit, and contains sample exam questions from OCR GCSE papers, as well as a glossary of terms. Answers to all questions, with some hints and tips on how to tackle questions, are given in an Appendix.

The book is also available as a downloadable PDF, sold as a lifetime site licence to schools.

To accompany this book, PG Online also publishes a series of seven downloadable teaching units for this specification. Each lesson in the unit consists of a PowerPoint presentation, teacher's notes and worksheets to be completed during the lesson. Homework sheets with exam-style questions are also included in most lessons. These units are sold as a lifetime site licence and may be loaded onto the school's private network or VLE.

For more information on the PDF version of the book and the teaching units, please visit www.pgonline.co.uk

About the author

Susan Robson has a degree in Computer Science from Manchester University and a PGCE from Portsmouth University. She worked for International Computers Ltd (pre Fujitsu days) straight out of university, followed by 12 years in technical pre-sales and then sales for ECI Telecom before moving into teaching Computing at Queen Mary's Sixth Form College, Basingstoke. She is currently Head of Computing at Bedales in Petersfield where she teaches GCSE and A Level Computing courses.



PG ONLINE

Table of Contents

CHAPTER 1: FUNDAMENTALS OF COMPUTER SYSTEMS	1
Basic Computer System Model	1
Importance of Computers	2
Professional Standards	3
Considerations When Creating Computer Systems	5
Glossary of Terms	7
Past Exam Questions	8
CHAPTER 2: HARDWARE	9
The Central Processing Unit (CPU)	9
Memory	11
Secondary Storage	14
Innovative Computer Design	18
Input and Output Devices	19
Binary Logic	26
Glossary of Terms	29
Past Exam Questions	31
CHAPTER 3: SOFTWARE	33
Operating Systems	34
Utility Programs	38
Application Software	42
Glossary of Terms	45
Past Exam Questions	48
CHAPTER 4: DATA REPRESENTATION IN COMPUTERS	49
Units	49
Numbers	50
Characters	54
Images	56
Sound	57
Instructions	58
Glossary of Terms	60
Past Exam Questions	63

CHAPTER 5: DATABASES	64
The Database Concept	64
The DBMS	67
Relational Databases	69
Glossary of Terms	78
Past Exam Questions	80
CHAPTER 6: NETWORKS	82
Local Area Networks (LAN)	82
Wide Area Networks (WAN)	88
How Devices Communicate on a Network	89
The Internet	91
Creating Web Pages in HTML	96
Security	99
File Types and File Compression	103
Glossary of Terms	106
Past Exam Questions	110
CHAPTER 7: PROGRAMMING	112
Algorithms	112
Control Flow in Imperative Languages	115
Programming Languages	119
Handling Data in Algorithms	122
Errors	127
Testing	129
IDE Tools	131
Glossary of Terms	135
Past Exam Questions	138
APPENDIX	140
Answers to Sample Exam Questions	140
Index	151

Chapter 1: Fundamentals of Computer Systems

This topic is about developing a mental model of a computer system in terms of its key components.

Basic Computer System Model

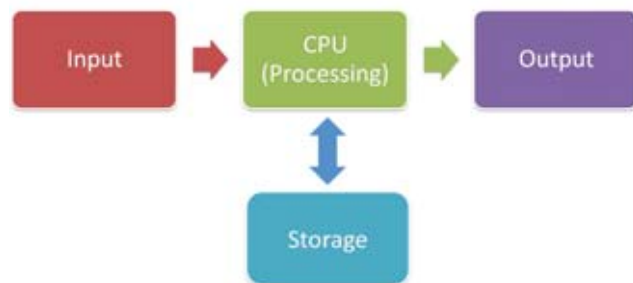
The OCR Specification says that you should be able to:

- define a computer system

A computer system is made up of hardware and software. Hardware is any physical component that makes up the computer. Software is any program that runs on the computer.

Computer systems are all around us. They are not just the PCs on the desk but include mobile phones, cash machines, supermarket tills and the engine management systems in a modern-day car.

The diagram **to the right** shows the basic model of a computer system.



All computer systems must have at least one **input device** that gets data from the real world. These could be a mouse and keyboard on a conventional PC but could be a temperature sensor (thermistor) in a commercial greenhouse or the microphone on a mobile phone. Input devices take real world data and convert it into a form that can be stored on the computer. More details on this can be found in *Chapter 4: Data Representation*.



The input from these devices is processed and the computer system will generate outputs. The **output device** could be, for example, a conventional computer screen, an **actuator** that opens or closes a greenhouse window, or the speaker that produces sound on a phone.



The fourth component is **storage**. The computer system may need to use stored data to perform the processing and, as a result of processing input, may generate data that is then stored.

Any computer system will have these four basic components. These components will be discussed in more detail in *Chapter 2: Hardware*.

Importance of Computers

The OCR Specification says that you should be able to:

- describe the importance of computer systems in the modern world
- explain the need for reliability in computer systems

Think about all the places you visit and the computer systems that you see and use in the course of a day. Computers are embedded in household appliances such as fridges and microwave ovens. Your mobile phone, TV and games console are also computer systems. When you ride in a car, a computer system is controlling the engine as well as the car's GPS, windows and stereo.

We are becoming increasingly dependent on computer systems at school, at work and at home. Consider what problems you would have if all of these computers stopped working! If the games console didn't work you might be upset but it wouldn't be a big problem. If the network at school went down you wouldn't be able to access any of the resources and teachers wouldn't be able to show you their slides or use smart-whiteboards. If there are problems with computer systems in shops they can't sell anything and if bank systems have problems, people cannot access their money or worse still, other people can!



For you to find out...

Look at some websites to find stories about computer systems failures.

If computer-controlled signage on motorways goes wrong there could be health and safety implications. If computer systems in hospitals go wrong people could die. The reliability of computer systems is extremely important. In some situations it is more important than how quickly they work or how many features they have.

Measuring Reliability

How do you know how reliable a computer system is? We talk about system **availability** meaning the percentage of time it is available for. For example, if a system is down (unavailable) for 1 hour in a 100 hour period, then the system availability is 99% for this period. Normally equipment manufacturers will quote average availability over a long period of time.

Another way of measuring a system's reliability is to measure the length of time between system failures. **MTBF** is the **Mean Time Between Failure** and is quoted by equipment manufacturers as an indication of reliability.

Protecting Against System Failures

If a computer system's reliability is very important then a company may need to have spare hardware in place just in case of system failures. **Hardware redundancy** is one way to protect systems. This means having more than one of a critical component. For example, computer systems in commercial aircraft have triple redundancy; three separate identical computer systems control each major function so that if one or even two systems fail, the third will hopefully get you to your destination safely.

In general, if software or hardware problems occur, it is the data which is most at risk. This will be stored on a disk on a server somewhere. As well as redundant hardware, companies must make sure that data is backed up regularly.

Sometimes computer systems are so important to a company that it cannot function without them. A computer system failure could mean that a company goes out of business in just a few days.

Disaster recovery plans need to be put in place so that a company can switch over to a complete new system with all its own data on it, often within hours. There are disaster recovery companies that specialise in offering this service to companies who depend on their computer systems.



For you to find out...

Look up "RAID computer storage" to see how companies protect their computer systems against disk failure.

Professional Standards

The OCR Specification says that you should be able to:

- explain the need for adherence to suitable professional standards in the development, use and maintenance of computer systems

Computer systems are complex. If everyone involved in creating, using and maintaining systems did things their own way it would make them even more complicated and difficult to understand. Professional standards may be formally agreed or may be **de facto** ("de" is pronounced "day").

De facto standards evolve when lots of people start working in a certain way, and more people think it looks like a good idea and start doing it that way as well. Over time, this way of working becomes a standard, just because that is how most people do it. It is not a formally agreed standard where a committee of people sit down and agree how something should be done.

Professional Standards in System Development

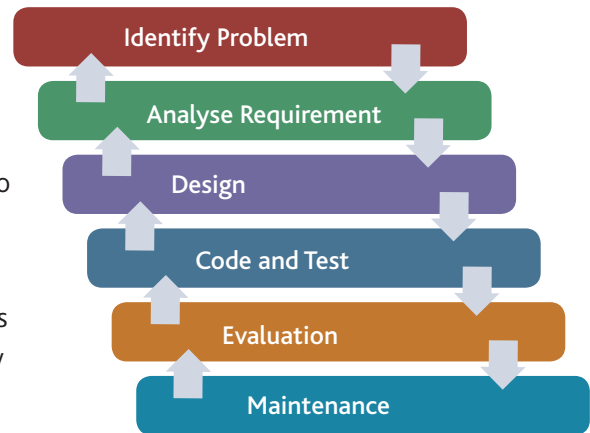
There are several approaches, or models, that define how a computer system is developed. Any new system starts with an idea or a need that must be analysed so a system can be designed and created. This development may take many months or years so there needs to be a way of splitting the process down into logical steps and defining what each step will involve.

One such model is the **Waterfall Model**. This model defines specific steps that are completed one at a time to guide the process from beginning to end. Each step has specific outputs that lead into the next stage. You can return to a previous stage if necessary but you then have to work back down through the following stages.

Another model is **Rapid Application Development (RAD)**, where the client is much more involved in the process. This method starts with a prototype that is developed gradually into a full solution with customer feedback at each stage.

These are both standard approaches that professionals in the industry and teams of developers would be familiar with.

They wouldn't have to learn a new methodology every time they move to another job.



Waterfall Model example

Professional Standards in Coding

De facto standards in programming include the use of:

- comments in code to outline the algorithm and define the purpose of each part of the program
- indents to clearly show where sections of code are inside a construct such as a loop
- meaningful identifiers so anyone can read the code and understand it, for example "Total" instead of "X"

These are all considered "good practice", the best way to do something.

Using professional standards also means that programmers do not plagiarise other people's work. When developers write programs, they create something from their own mind. This is referred to as **intellectual property**. The **Copyright, Designs and Patents Act** protects people's intellectual property rights. It protects the developer or programmer who has worked hard to create something and stops somebody else using it for their own gain. The **Federation Against Software Theft (FAST)** attempts to enforce copyright laws and protect programmers' rights.

However, some developers prefer not to protect their work in this way. Instead they actively collaborate with other programmers and create **Open Source** code, which others can use at no cost. The professional standards applied here are that any new code created from open source code is then also shared as open source code. ("Open Source" is discussed in more detail in *Chapter 3: Software*.)

Professional Standards in Documentation

Professional standards apply to documentation as well as coding. A system developer will document the system using standard diagrams. These diagrams all use recognised sets of symbols.

While studying this GCSE you will use flowcharts to design algorithms. (See *Chapter 7: Programming*.) The symbols used in these diagrams are industry standard. You will also draw logic diagrams using standard symbols for the **NOT**, **AND** and **OR** logic gates.

These standards mean that system developers can:

- work in teams to develop a system because they all have a common understanding of the design tools and diagrams
- move between companies because the standards apply across the whole industry
- pick up someone else's design and code the program to implement it
- maintain (amend or add to) somebody else's program when the customer's requirements change

Professional Standards for Health and Safety

Where the safety of people is concerned there needs to be more than just a de facto standard. There are laws that protect people and define standards for how computer systems should be used. For example, the Health and Safety at Work Act includes Display Screen Regulations. When developing a computer system, its use on a day-to-day basis must be considered. Developers design systems to meet Health and Safety regulations.

Considerations when Creating Computer Systems

The OCR Specification says that you should be able to:

- explain the importance of ethical, environmental and legal considerations when creating computer systems

When creating a computer system the designer will take into account the system requirements, the money available and the timescales it has to be produced in.

There are also legal, ethical and environmental considerations that will affect the system's design.

Legal Considerations

When a computer system is designed and implemented it must meet legal requirements. Listed below are some of the laws that system designers should consider.

- **The Data Protection Act** says that anyone who stores personal details must keep them secure. Companies with computer systems that store any personal data must have processes and security mechanisms designed into the system to meet this requirement.
- **The Health and Safety at Work Act** makes employers responsible for their staff. Design considerations should provide appropriate working conditions for staff. Designers should consider how easy systems will be to use and any health implications there might be based on their choices of software, screen layout, input methods and the hardware used.
- **The Copyright, Designs and Patents Act** makes it illegal to use software without buying the appropriate licence. When a computer system is designed and implemented, licensing must be considered in terms of which software should be used. Is Open Source the way to go or is the cost of proprietary software worth it in the long run?

Ethical Considerations

Ethical considerations are all about fairness. When creating a computer system a company should consider fairness. This may impact where call centres are located and where programming work is done. Here are some examples of ethical issues around computer systems:

- Is it fair that some people cannot afford computers?
- Are countries like India being exploited as a source of cheap labour for call centres and for programming?
- Should companies use local programmers and call centres?
- Does the system design disadvantage some part of the community?
- Does the system design promote accessibility for all?

Environmental Considerations

Environmental issues include the carbon footprint and waste products that result from manufacturing computer systems, but this is often outweighed by the positive effects on the environment of using computerised systems to manage processes that might otherwise generate more pollution.

Considerations may include:

- Does a computer system mean that people can work from home and therefore drive less?
- Does a computer system mean more manufacturing?
- Is working at home more environmentally friendly than everyone working in a big office, in terms of heating and lighting?
- Do computer-managed engines work more efficiently? Create less pollution and use less fuel?



GLOSSARY OF TERMS

Basic Model

Computer System	A combination of hardware and software components that allow input, processing and output of data.
Hardware	The physical components that make up a computer system.
Software	The programs that run on a computer system.
Input Devices	Hardware that accepts data into the computer. They take real world analogue data and convert it into a digital form that can be stored on a computer. For example: keyboard, mouse, microphone, webcam, scanner, sensors.
Output Device	Hardware that presents the results of processing to the user or actuators that perform a task automatically. They use digital data from a computer and produce it in a form that is understandable or usable. For example: monitors, printers, speakers, projectors, actuators (motors).
Storage Device	Hardware that is used to store files long term and is non-volatile, such as hard disks, memory sticks, magnetic tapes and CDs.

Importance of Computers

Reliability	How far you can depend on the computer system being available when you need it. Usually measured in terms of availability.
Availability	The proportion of time that a system is operational, usually expressed as a percentage over a period of time. E.g. 95% measured over one year.
MTBF	Mean Time Between Failure: a measure of availability often quoted by hardware manufacturers. For example 2.56 years between failures means that, on average, the hardware can be expected to last 2.56 years before it goes wrong.
Redundancy	Spare hardware components are built into a system so that, in the event of a component failing, the system can swap over to the spare one. <i>(Discussed in Chapter 6: Networks)</i>
Disaster Recovery	Where a company has plans to replace a system quickly if there is a catastrophe (fire, flood, bomb etc). Designed to minimise the time the system is down. <i>(Discussed in Chapter 6: Networks)</i>

Past Exam Questions



June 2011, Question 1

- 1 (a) State what is meant by a storage device, an input device and an output device in a computer system.

Storage Device:

Input Device:

Output Device:

[3]

Jan 2012, Question 1

- 2 Bytes, Kilobytes and Megabytes are units used for the amount of data stored in a computer.

- (a) State which of these units is most appropriate for the following items of data.

A one page text document

A ten minute movie clip

A person's surname

[3]

- (b) A computer has a hard disk of 2 Terabytes.

How much is this in Gigabytes?

[2]

Jan 2013, Question 7

- 3 The accident and emergency department of a hospital uses a computer system to decide the order in which patients are treated.

Describe advantages of using a computer system instead of a person to decide the order, and the need for this system to be reliable.

The quality of written communication will be assessed in your answer to this question.

[6]

Chapter 2: Hardware

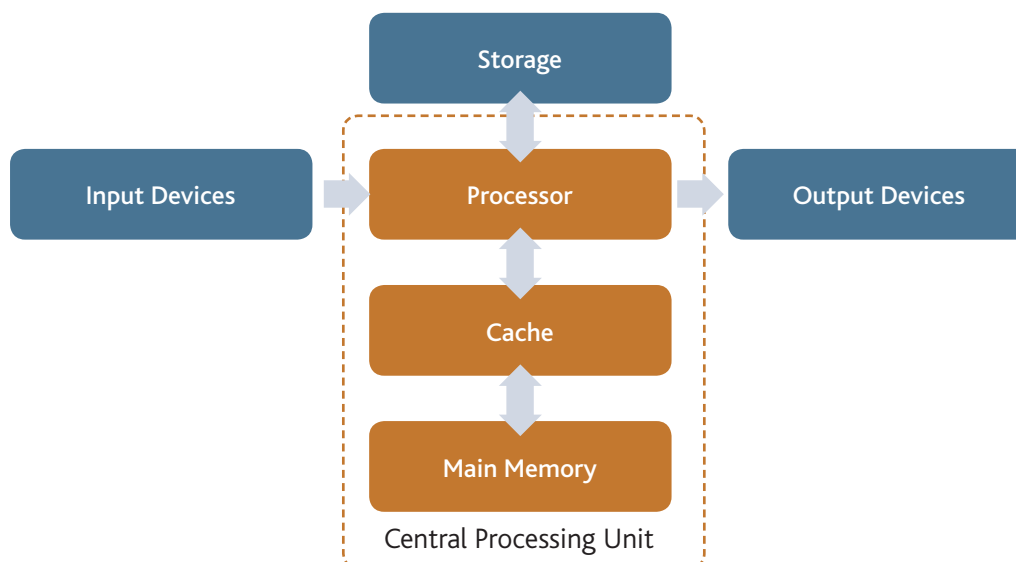
In Chapter 1 we looked at the basic components of the computer system. This topic is all about the hardware components in this system and looks at the CPU, Memory, Storage, I/O devices and hardware logic circuits that make up these systems.

The Central Processing Unit (CPU)

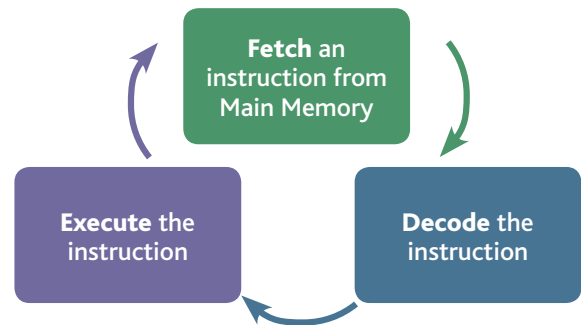
The OCR Specification says that you should be able to:

- state the purpose of the CPU
- describe the function of the CPU as fetching and executing instructions stored in memory
- describe cache memory
- explain how common characteristics of CPUs such as clock speed, cache size and number of cores affect their performance

The **central processing unit (CPU)** of a computer is the hardware that executes programs and manages the rest of the hardware. Think of the CPU as the brain of the computer. Just as your brain contains parts that remember things, parts that think and parts that make the rest of your body operate, the CPU does the same for the computer. The CPU is made up of the main memory, the processor and the cache memory (we'll talk about the cache later). The program instructions and data move between the main memory and the processor using internal connections called **system buses**.



When a program is to be run (executed) on a computer it first has to be loaded into the main memory. From here it can be accessed by the processor, which runs each instruction in turn. When the program is loaded, the processor is given the start address of where the program is held in main memory. To run the program the processor fetches an instruction, decodes it and then executes it. The processor executes one instruction at a time. This is called the **fetch-execute cycle**.



The speed at which a processor operates is quoted as the **clock speed**, in Hertz. Hertz is the name for the number of electrical cycles per second, or the rate at which the electrical current changes in the actual circuits. Everything the processor does happens on the tick of the clock so a faster clock means that more instructions are fetched, decoded and executed in a second. The speed of the processor is measured in MHz (Megahertz) or GHz (Gigahertz). At time of writing (Aug 2014) processors in typical home PCs might be rated at 3 - 4GHz.

CPU Performance

In theory a computer with a 400MHz processor should operate twice as fast as one with a 200MHz processor but it isn't that simple. There are lots of other components that contribute to the overall speed of the computer. Each one can create a bottleneck in the system and slow it down. Imagine a three lane motorway where the traffic can go really fast until it gets to a single lane road going into a town. No matter how many lanes you add to the motorway, the single lane part of the journey will limit how quickly traffic can get into town. In the same way, a slow component can slow the whole computer down.

One bottleneck that can occur is the access speed of main memory. Reading from and writing to main memory is much slower than the speed at which the processor can work. The logical answer is to use faster memory technologies but this increases the price of the computer. Modern computers need to run many programs at the same time so they need lots of memory. There needs to be a compromise between speed and cost.

One way of improving speed at minimal cost is to use a small amount of much faster memory where frequently used instructions or data can be stored for a while. We call this special sort of high-speed memory the cache (pronounced "cash"). If the processor has to access main memory less often it can work faster so the CPU performance increases.

A typical PC (in Aug 2014) might have 16GB of RAM (main memory) but only 4MB of the faster more expensive cache memory. Notice the different units here and remember that there are 1024 Megabytes in a Gigabyte. This computer therefore has 4096 times more RAM than cache memory.



For you to find out...

Look at some PC adverts in magazines and on the internet to get a feel for current computer specifications

The purpose of the CPU can be summarised as follows:

- fetch instructions from main memory
- fetch data from main memory
- decode the instructions
- execute the instructions
- perform calculations
- manage the movement of instructions and data to and from peripheral devices

Multi-core Processors

When we looked at the basic fetch-execute cycle we assumed that there was a single processor and a single main memory. You have probably heard terms like “dual-core” and “quad-core” so where do these fit in? Today’s more complex CPUs can include more than one processor, or core. A dual-core processor has two processing components within the CPU. A quad-core, likewise, has four. In theory having two processors means that the computer can operate twice as fast but this isn’t always the case.

Imagine a bakery where one chef is making cakes. If you brought in three more chefs they could all make their own cakes and the bakery would make four times as many cakes. If there was only one recipe, though, you couldn’t have Cook 1 doing the first line of the recipe, Cook 2 doing line 2, Cook 3 doing line 3 and so on. Cook 2 would need to wait for the first cook to mix the flour and butter before he could fold in the eggs! So four cooks working their way through one recipe wouldn’t be any quicker, in fact it would be more complicated (you’ve heard the saying “Too many cooks spoil the broth”). Likewise a program is a series of instructions that need to be done in order. Multiple processors could work on different programs that operate in parallel but unless the computer is designed to use multiple cores it isn’t necessarily four times faster. On the whole, though, a PC with lots of programs going on at the same time will have a multiple-core processor and will operate faster than a single-core processor.

Memory

The OCR Specification says that you should be able to:

- describe the difference between RAM and ROM
- explain the need for ROM in a computer system
- describe the purpose of RAM in a computer system
- explain how the amount of RAM in a personal computer affects the performance of the computer
- explain the need for virtual memory
- discuss how changes in memory technologies are leading to innovative computer designs

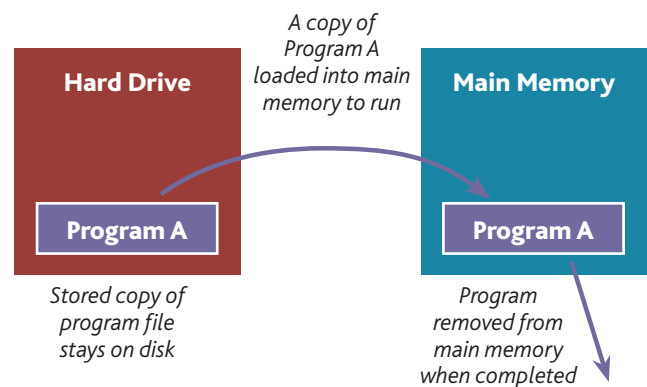
Memory in a computer system refers to the components that store (or remember) instructions and data. There are different types of memory with different purposes. Some are very fast and expensive such as the cache memory discussed earlier, and some are cheaper and slower such as the memory sticks you probably use at school.

These different types of memory have different characteristics and can be compared in terms of their access speed (how quickly you can read from them or write to them), their price and the whether they can store data when the power is turned off (volatility). Let's consider some common types of memory in computer systems:

Random Access Memory (RAM)

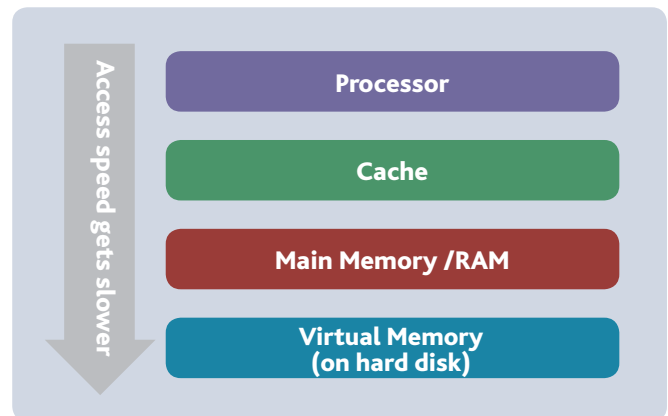
RAM is the type of memory used in the computer's main memory. Many people say "RAM" when they mean main memory. Nothing in a computer is really "random" so random access just refers to the fact that you can write anywhere in that memory space at any time, you don't have to put the next thing straight after the last one like you do on a magnetic tape, for example.

As we said earlier in the chapter, when a program is running it has to be loaded from the hard disk into main memory so that the processor can access the instructions. Any data needed for that program to run is also loaded into main memory while the program is running. The main purpose of RAM is to act as temporary storage for programs and data, just for the duration of that program. Once the program has finished and is closed, it is no longer held in main memory.



So why doesn't the processor get the instructions straight from the disk? It is about access speed. Reading from and writing to a hard disk is very slow compared to the speed of the processor. Just as we used the high speed cache between the processor and main memory we need main memory to store the programs currently in use or the computer would be really slow.

As every program in use should, ideally, be in main memory while it runs, the amount of main memory also affects the performance of the computer. If a computer system has lots of programs running at the same time it needs lots of main memory.



Virtual Memory

Sometimes there just isn't enough main memory for all the programs that need to run. Computers can be configured so that part of the hard disk behaves like main memory. This is called **virtual memory**. The access speed on a hard disk is much slower than the speed of RAM so this isn't ideal. It is used to store parts of programs currently being run but the parts actually being executed still need to be in main memory. As the processor gets to the next part of the program, sections are swapped between virtual memory and main memory. Sometimes this works well but sometimes the computer spends more time swapping bits around than it does executing the program!

Volatility

When you are in the middle of a piece of work at school and your "friend" turns the computer off you will notice that you lose the work you did since you last saved. This is because the saved version goes onto the hard disk but the most recent version was only in main memory/RAM when the power went off. RAM is described as **volatile**; it loses its contents if there is no power. The hard disk is designed for long term storage of files and is **non-volatile** memory.

Read Only Memory (ROM)

RAM is volatile so when you turn off your computer it loses its contents. When you turn the computer back on it needs to get the basic startup routine from somewhere that is not volatile. The operating system and all your programs will be stored on the hard disk but these need to be loaded into RAM to run.



The computer has a piece of software called the **bootstrap loader**. This is a small program that loads the operating system. Once the operating system is loaded it takes care of the rest. The term comes from the idea that when you're not doing very well in life you can "pull yourself up by your bootstraps". In this day and age you're more likely to hear "sort yourself out" but the meaning is the same – you are in effect getting yourself restarted. Bootstrapping became abbreviated to booting, a term you have probably heard before. To "boot" a computer is to start it up from scratch.

ROM is **Read Only Memory**; you cannot write over the contents once it has been created. It is also non-volatile; you can leave the computer switched off for months and it will still start up as soon as it has power again. RAM on the other hand is only used for temporary storage of programs when they are running. RAM is read-write and volatile.

ROM	RAM
Read only	Read – write
Not volatile	Volatile

Secondary Memory

Main memory is also known as **Primary Memory** or **Immediate Access Store**. Memory is all about storage so, strictly speaking, storage devices such as hard disks and memory sticks (also called USB flash drives) are also a type of memory. These long term, non volatile types of storage are also called **Secondary Memory**.

Typically a PC will have a hard disk that stores all the files long term. Secondary memory/storage also includes memory sticks, tapes, CDs and DVDs. These are discussed further in the next section.

Secondary Storage

The OCR Specification says that you should be able to:

- explain the need for secondary storage
- describe common storage technologies such as optical, magnetic and solid state
- describe flash memory
- select suitable storage devices and storage media for a given application and justify their choice using characteristics such as capacity, speed, portability, durability and reliability

We all want to store files for a long period of time. We keep photographs, projects, music, films, letters and spreadsheets on our computers. We also expect our programs to be there when we switch the computer on. This long term storage is sometimes called **Secondary Memory** or **Secondary Storage** (primary memory is the main memory).

Secondary storage is usually much larger and, as we want such a lot of it, it needs to be cheap. Cheaper memory technologies tend to have slower access speeds than main memory, as discussed earlier. Secondary memory is non-volatile and needs to be robust and reliable. Has your memory stick been through the washing machine yet?

Choosing the right type of storage medium for a particular use is important. You need to consider the following features:

- **Capacity:** How much space there is to store files. Compare the size of a floppy disk at 1.44MB to a CD that can store 700MB. You can only fit half of this book on a floppy disk.
- **Speed:** How quickly the computer can read data from a storage device or write data to it. (When you write to an external hard disk or memory stick, you should use the little utility program that tells you whether or not the device can be safely removed from the computer before you remove it, or you may lose some data.)



- **Portability:** Can you easily unplug it and carry it away? Does it fit in a pencil case or do you need a large bag?
- **Durability:** How easily is it damaged? Will it survive dropping or having coffee tipped on it?
- **Reliability:** How long will it last? Anything with moving parts is likely to be less reliable.

There are different storage technologies available:

- **Magnetic:**

This is the oldest of the technologies and is used in floppy disks, hard disks and tapes. It is cheap and high capacity (stores a lot of data). Magnetic disks are read with a moving head inside the disk drive and magnetic tapes are read by moving the tape past a read-write head. Moving parts make these media quite slow to read from or write to and moving parts go wrong more often than with solid state media. Magnetic media are also vulnerable to damage as well. Just as a video tape left on a stereo speaker is damaged by the magnets, so too a magnetic disk or tape can be easily wiped.



Internal hard drive (opened)



External hard drive

MAGNETIC STORAGE DEVICES			
	Internal Hard Disk	Portable Hard Disk	Magnetic Tape
Cost	Very cheap but not as cheap as tape 3TB internal for £95		Cheapest bulk data storage medium 1.5TB tape £23
Capacity	Up to 4TB		
Access Speed	3Gb/s	480Mb/s <i>(speed of USB2 interface)</i>	Slow access speed because tape drive has to read through the tape serially
Portability	Not portable, built into PC	Can fit in a large pocket	Cassette tape is compact but needs a tape drive to use
Durability	Good durability when disk not in use but vulnerable to movement when spinning. Can write to the disk an infinite number of times. Affected by heat and magnetic fields		Limited lifetime, wears out with repeated use. If used once for archiving can last 15 years. Affected by heat and magnetic fields.
Reliability	Extremely reliable		Very reliable if not damaged
Typical use	Inside a PC as secondary storage	Supplementary storage for a PC or portable storage where high capacity is required	Good for backups and archiving but access speed is too slow for general use

- **Optical Storage Media:**



The word “optical” should make you think about the eye and how we see the world in terms of reflected light. Optical media work in a similar way. Lasers write data to the disc and read data from it. Optical media include CDs and DVDs.



OPTICAL STORAGE DEVICES		
	CD	DVD
Cost	Very cheap 50 CD-R for £12 (24p each)	Very cheap 50 x 4.7GB DVD-R for £15 (30p each)
Capacity	640MB	4-17GB
Access Speed	Up to 7.6 MB/s (52x)	16 MB/s (12x)
	<i>Much slower than magnetic hard disk</i>	
Portability	Easy to carry in a large pocket or bag	Easy to carry in a large pocket or bag
Durability	Depends on how it is stored. Quality degrades over time, life expectancy of a CD-R about 20 to 100 years but can start degrading in 18 months!	Recordings last 2-15 years so not considered reliable for long term storage
Reliability	Good for medium term but degrades over time	
Typical use	CD-ROM for software distribution CD-R or CD-RW for backup/archive	Backup/archive where higher capacity is needed

• **Solid State/Flash Storage Media:**



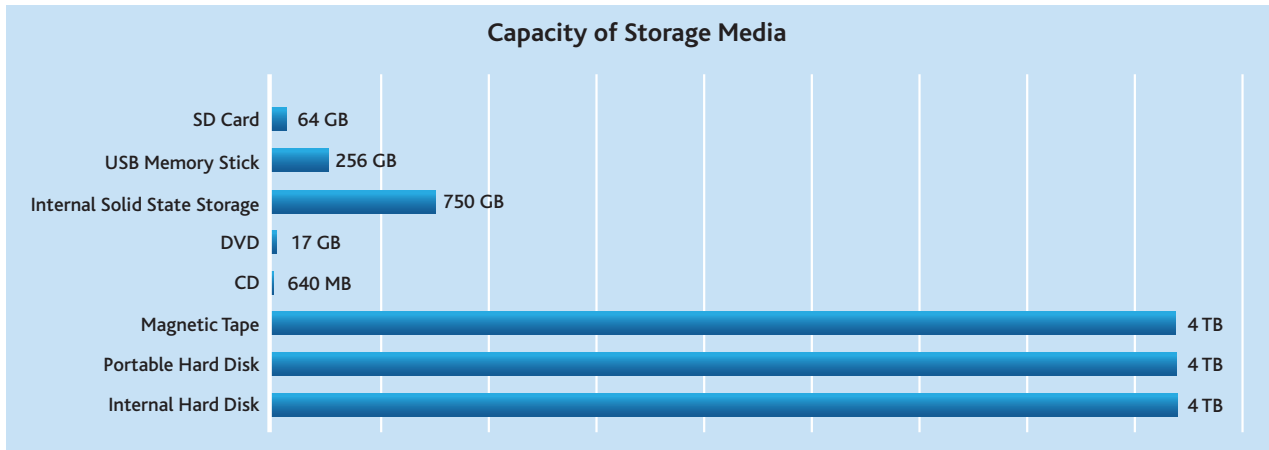
If you need to move work from school to your home PC you might use a USB Memory Stick (sometimes called a pen drive). These use a newer technology called **flash memory**. It is solid-state memory, which just means that it does not have any moving parts. This makes the access speed of flash memory very high (but not as fast as RAM) and there is less to go wrong. Flash memory is a type of RAM but will not replace the type of RAM used in the main memory as its access speed is too low. It is a non-volatile type of RAM so it makes an excellent replacement for a hard disk in devices like notebooks and tablet PCs. This type of memory is also used in mobile phones, tablet PCs and cameras.

? For you to find out...

How much storage/memory there is on a modern camera?
How do hard disks and CD ROMs work?

FLASH STORAGE DEVICES			
	Internal Solid State/ Flash Storage	USB Memory Stick	Memory Card
Cost	128GB for £65 750GB for £330	32GB for £15 128GB for £35	64GB SD card £29
Capacity	128 to 750GB	64MB to 256GB	128 MB to 64GB
Access Speed	6Gb/s <i>(faster than magnetic disk because no moving parts)</i>	480Mb/s <i>(speed of USB2 interface)</i>	<i>Dependent on type of card and device interface</i>
Portability	Not portable, built into PC	Very small, can put in any pocket or on a key-ring	Very small, designed for portable devices
Durability	More robust than hard disks with moving parts. Said to be 5-10 times more durable than a hard disk drive	Very durable – often survive washing machine! Some can be snapped quite easily	Very durable - not sensitive to temperature or knocks
Reliability	Extremely reliable	Very reliable but can corrupt files if removed from PC too soon	Very reliable
Typical use	Notebooks, tablets, slim laptops	Personal use, moving files between computers	In phones and cameras

Storage Media Capacity Compared



Note:

Details in the above tables and graph may already be out of date (written August 2014). You should research products that are currently on the market to see how quickly technology advances.

Innovative Computer Design

The OCR Specification says that you should be able to:

- discuss how changes in memory technologies are leading to innovative computer designs

Computer technology changes very quickly so you should keep in touch with current changes by reading magazines and websites that are up to date. Memory technologies are changing in several ways so consider how these affect the products coming into shops now. Factors include:

- **Capacity and density:** You can now buy a 64GB SD card for a camera where a couple of years ago a 4GB card was considered high capacity. Simply increasing capacity isn't too challenging; just add more cards. The ability to increase the capacity on the same size card is the challenge. Higher density means devices can store more without getting bigger, or devices can store the same amount but get smaller. You only have to look at mobile phones to see how these have become smaller and lighter and yet can now store more music and even video. The capacity of primary memory has also improved so computers can run more applications at the same time. Complex games can run easily on today's "basic" computers.

- **Speed:** Although Flash memory is still much slower than RAM it is significantly faster than a magnetic hard disk. This means that bulky and sensitive hard disks can be replaced with smaller, lighter, faster and more reliable flash memory. Computers start up quicker as this is dependent on how long it takes to load the operating system from secondary storage into RAM. So now we have notebooks you can slip into a pocket, tablet PCs and smartphones that behave like computers.
- **Price:** Flash memory is cheaper and easier to produce so devices using flash memory become cheaper as well. Lower cost memory has improved the capacity of gadgets like MP3 players, hand-held game consoles and phones. The cost of RAM has dropped so a PC costing £500 today has more RAM and more storage than a £500 PC just a couple of years ago.

Input and Output Devices

The OCR Specification says that you should be able to:

- understand the need for input and output devices
- describe suitable input devices for a wide range of computer-controlled situations
- describe suitable output devices for a wide range of computer-controlled situations
- discuss input and output devices for users with specific needs

This book does not attempt to describe the full range of devices currently available. There are many websites that comprehensively describe how these work. It is however important to understand what input and output devices exist and how to choose appropriate devices for a situation.

Input Devices

We live in an analogue world of continuously changing variables. In order to process or store data we must somehow capture it and convert it into a digital representation. Chapter 4, "Data Representation", describes how we store text, numbers, sound, images and instructions on a computer. Here we are concerned with the devices that will capture the inputs.

Keyboard: Used for data entry into computers. This is a full keyboard with function keys and a number pad but laptops and other devices may have cut-down versions of this. Pressing a key generates a character code which is sent to the computer.



Mouse: Used for pointing at objects on the screen and selecting items by clicking or double-clicking. Right-click usually brings up a context-sensitive menu. A scroll wheel between the buttons makes it easy to move up and down the screen. Connected to the computer by a wire and USB connector or wirelessly, the mouse detects movements and position either with a ball or a reflected light source under the mouse. A variation on the mouse is a trackerball where the user moves a larger ball on the top surface instead.



Touch screen: Used on phones and tablet PCs as well as applications such as help screens in banks and tourist information offices. Instead of using a mouse you can just touch the icons to select them. On some devices moving two fingers apart or together on the screen zooms in or out and tapping icons is equivalent to a double-click.

Barcode scanner: A laser scans the barcode on a product to detect the width of the black and white stripes. This generates the product number so the rest of the details can be found in a database.



OMR: Optical Mark Recognition is used for multiple-choice tests, evaluation forms and lottery tickets, for example. This uses a light source that reflects a different amount of light back from shaded boxes to detect the data for input.



OCR: Similar to OMR but this is Optical Character Recognition. The software interprets the marks as characters and generates text and some formatting detail which is transferred to a file or document for further word processing.

Joystick: Although mention of a joystick probably makes you think of a game controller, joysticks do have other uses! They are used to control wheelchairs and to remotely control lights, security cameras and bomb-disposal robots. The game controller's cross-shaped control pad is a type of joystick.





Microphone: used to input sound that can either be recorded in a file or processed by voice recognition software for other uses. Voice recognition could be used for many applications such as hands-free phone dialing, controlling devices in a house if you are not mobile, a karaoke game or for security identification.

Sensors: These measure temperature, light levels, pressure, humidity and many other physical properties in the real world. These can then trigger processes and the appropriate output action as required. For example, if a thermistor (temperature sensor) reading is higher than 23°C, the computer could signal motors to open the greenhouse windows (see “actuators” under Output Devices in this chapter). The sensor shown is an Xbox movement sensor.



Sensors are also used to track eye movement or head movement to allow disabled people to use computers. Eye and head movements operate a pointer on the screen, instead of using a mouse.



Specialised keyboards: There are many variations on the keyboard and touch-screen that allow people with various disabilities to operate a computer. Features may include: bigger keys, specialised layouts for different purposes, braille overlays and brighter colours for contrast.

Foot-operated mouse: This works in the same way as a normal mouse, complete with left- and right-click buttons. It is designed to be easily operated with a foot instead.



Camera: A camera can be used as a web-cam so you can see the person you are phoning over the internet but it can also be a way of communicating with the computer if you do not have the use of your hands. Instead of using a mouse a disabled person can control an on-screen keyboard with eye or head movements detected with a camera.

Sip-and-Puff Switches: These are activated by the user’s breath. A “puff” into the device might be the equivalent of a mouse click and a “sip” (or suck) might be the equivalent of holding a key down. This would be used with an on-screen keyboard.



For you to find out...

Use www.howstuffworks.com to find out how these devices work.

Output Devices

Computer systems create many outputs for us. These can be printed reports, data on a screen, sound, pictures or an action that needs performing via a motorised control. Here are some output devices you may come across:

CRT Monitor: CRT stands for Cathode Ray Tube. These are the heavy, old-fashioned monitors that take up a lot of space on the desk. Very few people use these now but in places like factories they are still used because the glass screen is more robust than a TFT screen.



TFT Monitor: TFT stands for Thin Film Transistor. This type of screen is the thin screen you probably have at home or on a laptop. This is a variation on LCD technology. LCD stands for Liquid Crystal Display, the earlier version of TFT screens. These screens take up less space but are more easily damaged than a CRT monitor.

Speakers: To output sound the computer must synthesize sound from the digital file and output sound waves through speakers. These might be small speakers integrated into the computer, phone or other device, or they may be stand-alone speakers that attach to the computer by a headphone jack, for example. Headphones are a type of speaker.



Inkjet printer: This is probably the type of printer you have at home. It works by spraying ink onto the page so if you have big shaded areas on a page the page will come out a bit damp and wrinkled. For normal text they produce good quality printing at a low cost. The ink is never sealed on the page so if you use a highlighter pen on notes from an inkjet the ink will smudge. They are also quite slow so they are not suitable for a large office or classroom.

Laser printer: This is the type of printer you probably have in the classroom as it is quicker and quieter than an inkjet. The ink is in the form of a toner powder that is sealed onto the page by heated rollers. That is why your printed pages come out warm. The ink goes onto the page dry so the quality is much better and it won't smudge. These are more expensive printers and the toner cartridges are more expensive than inkjet cartridges, though you will get more prints.





Dot-matrix printer: Before inkjet and laser printers we had dot-matrix printers. They worked a bit like a typewriter as a print head has to strike the paper through an ink ribbon to print text on a page. A dot matrix printer is termed an impact printer for this reason. They are still used today where companies have to use multi-part stationery. A laser printer would only print on the top page but an impact printer pushes through several layers that have carbon paper in between each sheet of paper. The paper is fed through the printer using the holes at the side. The holes are then torn off to give a printed page.

Braille Printer: An impact printer that creates raised dots to represent each letter to allow blind people to read.



Actuator: The actuator is the device that performs some sort of mechanical action based on inputs. For example, a voice sensor might trigger a motor to close the curtains. Often these are used with sensors but they can also be used with touch screens. Modern aircraft use many sensors to provide input to computers that control the flight. Actuators move the flaps on the wings and the tail automatically.



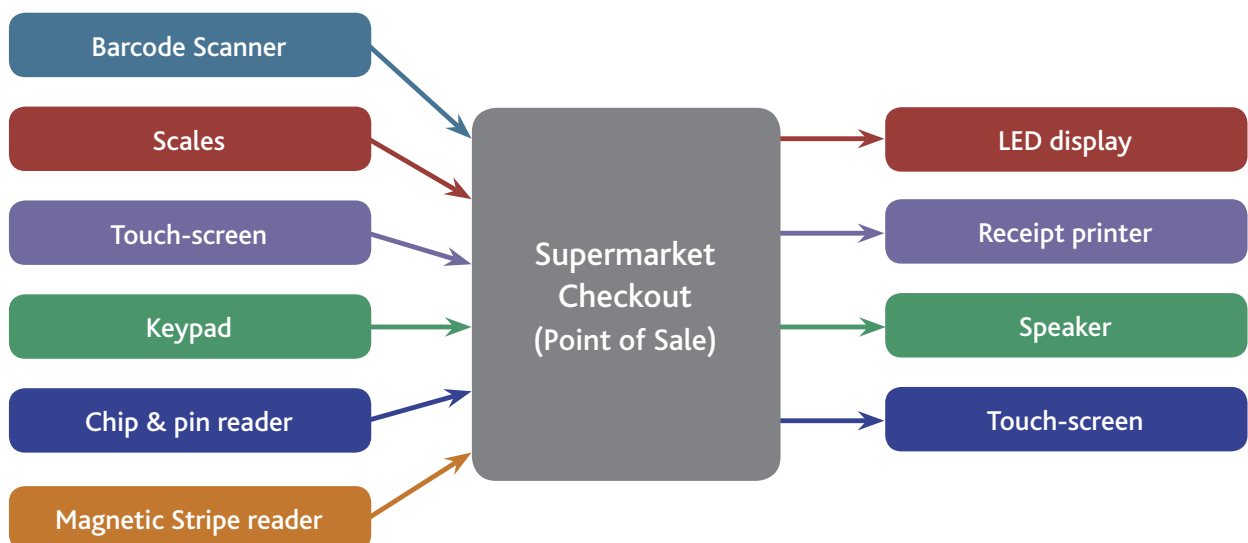
Lights (LEDs): Lights are used to show that Caps Lock is on, that the computer is in standby or that a laptop is connected to the mains. Light-Emitting Diodes (LEDs) are used to output different signals to the user.



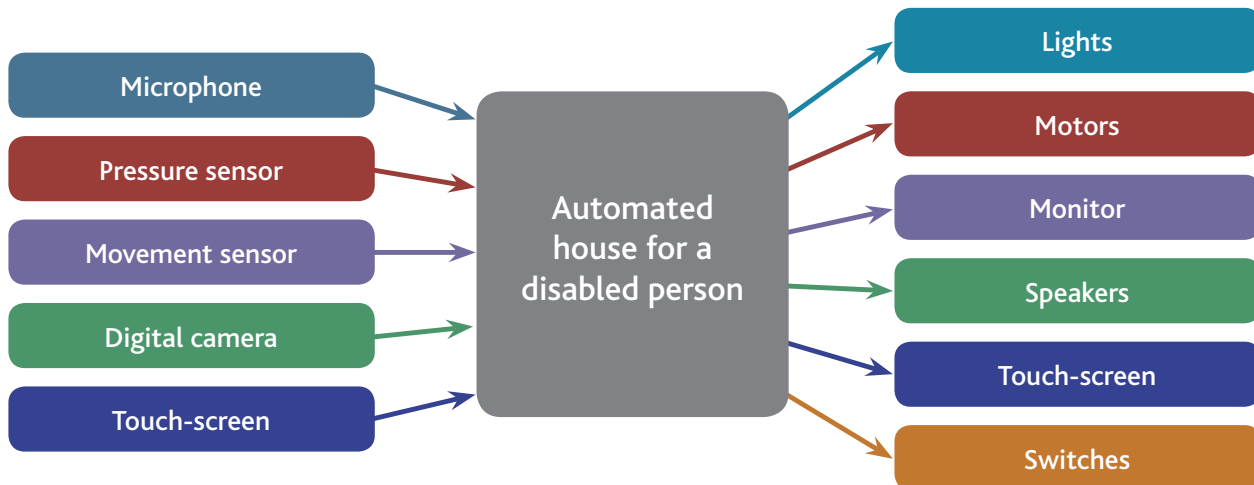
Appropriate I/O devices

In an exam you may be asked to identify suitable input and output (often abbreviated to I/O) devices for certain situations. We are all very familiar with a standard PC but think about how computers are used in different situations.

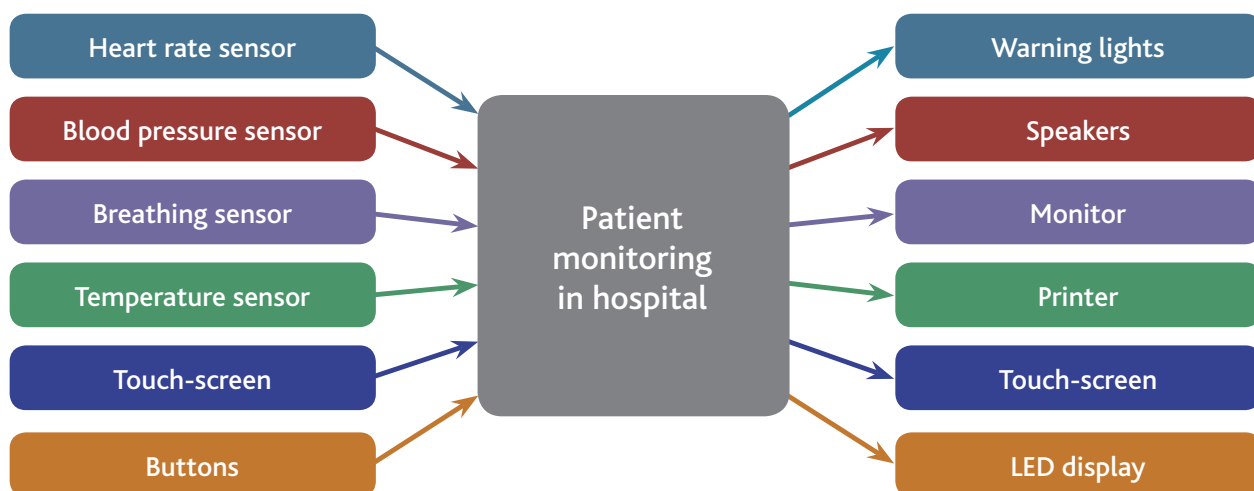
Supermarket Point of Sale (POS): The checkout at a supermarket has a wide range of input devices. The barcode scanner will get a product number from the item purchased, which will then enable the checkout to call up all the details from a database somewhere else in the store. The checkout has a weight sensor/scales for fruit and vegetables as well. The checkout operator must be able to type in product numbers when they don't scan and there will also be a touch-screen to quickly find produce such as apples that are being weighed. When the customer pays, the checkout will need to read credit card details and the checkout operator might swipe a loyalty card. Credit cards will have the card number stored on a chip but loyalty cards generally just have a magnetic stripe to store the customer ID.



Automated house for a disabled person: If you are in a wheelchair or find it hard to move around your home then simple tasks like closing the curtains or answering the door can be very difficult. Computers can be used to make these tasks easier to perform, so people can continue to live independent lives in their own homes. Input devices can take the form of a variety of sensors that trigger actions: for example, movement sensors that trigger lights and pressure sensors that open doors. A digital camera and a microphone at the front door can link to a monitor, speakers and microphone in the house so the person doesn't have to go to the door to see who is there. A touch screen can be used to control curtains, lights, heating, television and radio.



Patient monitoring in a hospital: Patients in a critical condition are linked up to computers that monitor their blood pressure, temperature, breathing rate and depth. These all use sensors of some kind. The output that nurses check will be displayed on a monitor showing the current readings, and there may also be speakers that produce alarms if readings fall outside certain limits and printers that produce a history of readings. To set the machine up the nurse will probably have a touch-screen and there may be some settings displayed in LEDs.



These are just a few ideas; you need to research other scenarios and suggest appropriate I/O devices to use for each one.

Binary Logic

The OCR Specification says that you should be able to:

- explain why data is represented in computer systems in binary form
- understand and produce simple logic diagrams using the operations NOT, AND and OR
- produce a truth table from a given logic diagram

Binary Logic in Programming

Understanding how binary logic works will help your programming. In your programs you often use complex Boolean expressions to control loops and selection statements – for example:

```
While NOT(EndOfFile) AND NOT (ItemFound).....
IF (X<=10) OR (CurrentCharNum>LengthOfString) THEN .....
```

You have probably programmed a repeat loop to carry on until the user typed an “N” or “n”. The loop would look something like this with the condition at the end:

```
Repeat
  |
  |
  |
Until (response = 'N') or (response = 'n')
```

A Boolean expression
 Another Boolean expression

*A condition in programming is made up of **Boolean Expressions** that are either true or false*

Each **Boolean expression** can be replaced with a letter which is called a **Boolean variable**.

```
Until (response = 'N') or (response = 'n')
```

Replace with the Boolean variable, X
 Replace with the Boolean variable, Y

```
Until X or Y
```

Just like the Boolean data type in programming, Boolean variables are either true or false. X and Y will be either true or false. We represent true with 1 and false with 0 to represent electronic circuits being open or closed, just like with binary.



Truth Tables

If you have more than one Boolean expression in a condition then you can use Truth Tables to show the outcomes.

$$Q = X \text{ AND } Y$$

X	Y	Q
0	0	0
0	1	0
1	0	0
1	1	1

For (X AND Y) to be true, X must be true and Y must be true so all three of these combinations are false.

(X AND Y) is true because X is true and Y is true

$$Q = X \text{ OR } Y$$

X	Y	Q
0	0	0
0	1	1
1	0	1
1	1	1

X is false and so is Y so (X OR Y) is false

Although X is false, Y is true so (X OR Y) is true

Although Y is false, X is true so (X OR Y) is true

(X OR Y) is true if either X or Y is true; here they are both true

$$Q = \text{NOT } X$$

X	Q
0	1
1	0

NOT X is the opposite of X so 0 becomes 1 in this case

NOT X is the opposite of X so 1 becomes 0 in this case

These basic gates can be combined into more complex expressions, for example:

$$Q = \text{NOT } (X \text{ AND } Y)$$

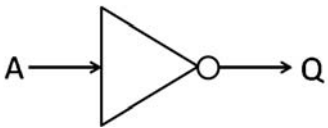
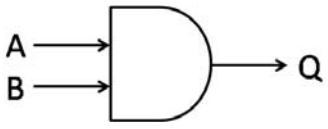
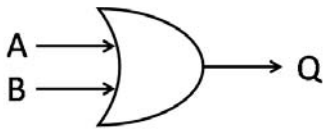
X	Y	X AND Y	Q
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Logic Diagrams

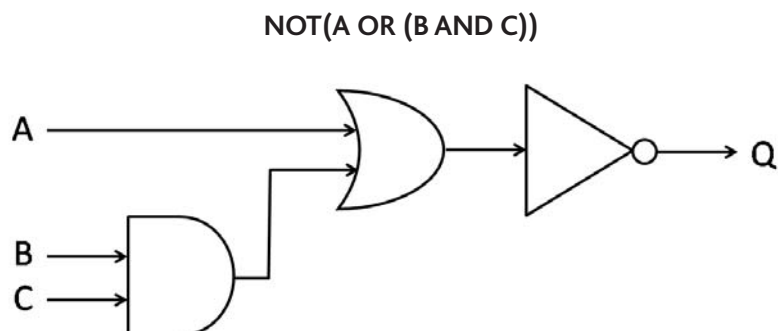
As we will see in *Chapter 4: Data Representation*, computers are based on electrical circuits where we can detect whether current is flowing or not. Binary uses base 2, so we have just two possible values 1 or 0. Representing data as binary values means we just have to detect these two values in electrical circuits.

Binary logic is about these most basic circuits. Circuits in computers are made up of many logic gates but at this level we are looking at just three of them: AND, OR and NOT. A given input will generate an output based on the logic gate in use.

We use specific symbols to represent the different logic gates; these are standard symbols. They can be used to represent Boolean expressions such as $Q = \text{NOT } A \text{ AND } (B \text{ OR } C)$.

NOT	
AND	
OR	

The logic gates can be joined up to make a circuit. For example:



Glossary of Terms

Units

CPU	The central processing unit that contains the processor, main memory and cache. (sometimes people say CPU when they mean processor so look for the context).
Main Memory/RAM	Also known as Immediate Access Store and Primary Memory The memory in the CPU that is used to temporarily store programs while they are running and the data used by these programs. The processor fetches instructions from main memory. Memory is made up of many addressable locations.
Processor	The component in the computer that fetches, decodes and executes instructions.
Cache	High speed memory in the CPU that is used to store a copy of frequently used instructions and data. Faster access speed than main memory. Used to improve CPU performance.
Clock Speed	Measured in hertz or cycles per second, the clock speed represents how many instructions per second the processor can execute. The higher the clock speed the faster the CPU can operate.
System Buses	The circuits/internal wiring that connect the processor and main memory.
Fetch-Execute Cycle	The process by which a program is run: instructions are stored in main memory, fetched by the processor one at a time, decoded and executed.
Dual-Core / Quad-Core	A CPU that contains multiple processor components (cores) that can operate independently to process more than one task at a time.

Memory

RAM	Random Access Memory: a type of memory that is read-write and volatile. Used for Main Memory.
ROM	Read Only Memory: memory that is hard-coded at the time of manufacture. Stores the startup program, called the bootstrap loader.
Bootstrap Loader	The first program that is loaded into main memory from ROM when a computer is switched on. This will load the operating system from secondary storage.
Volatile	Describes memory that loses its contents when the power is turned off, e.g. main memory
Non-Volatile	Describes memory that does not lose its contents when the power is turned off, e.g. hard disk.
Secondary Memory	Long term, non-volatile storage media such as hard disks, memory sticks, magnetic tapes, DVDs and CDs.
Virtual Memory	Part of the hard disk that is configured to behave as an extension to main memory.
Magnetic Media	Secondary storage such as hard disks, tape and floppy disks.
Optical Media	Secondary storage that is read using lasers such as CDs and DVDs.
Solid State / Flash Memory	Secondary storage that has no moving parts. Used in memory sticks, cameras and phones.
Pen Drive	Another term for a USB memory stick

Binary Logic

Boolean Expression	An expression that is either true or false, e.g. $X=10$
Truth Table	A table that shows all the possible combinations of inputs and their logical output value.
Logical Operators	AND, OR, NOT
Logic Diagram	A diagram of a circuit showing logic gates with inputs and the output these generate.

Past Exam Questions



Jan 2012, Question 8

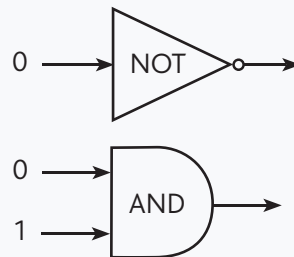
- 1 Mina's computer has 4 GB of RAM.
- (a) Describe the purpose of RAM in the computer. [2]
 - (b) The computer also uses virtual memory.
 - (i) Explain what is meant by virtual memory. [2]
 - (ii) State why virtual memory is needed. [1]
 - (iii) Mina upgrades the computer to 6 GB of RAM.
Explain how this upgrade will affect the performance of the computer. [2]

June 2012, Question 7

- 2 The CPU is the component which does most of the processing in a computer.
- (a) State two tasks which are carried out by the CPU when processing data. [2]
 - (b) Explain how the clock speed and the cache size of a CPU affect its performance. [4]

Jan 2013, Question 3

- 3 (a) State the output of each of the following logic circuits for the inputs given.



[2]

- (b) Fig. 1 is a circuit diagram. Complete the truth table for fig. 1.

[3]

p	q	(NOT p) AND q
0	0	0
1	0	0

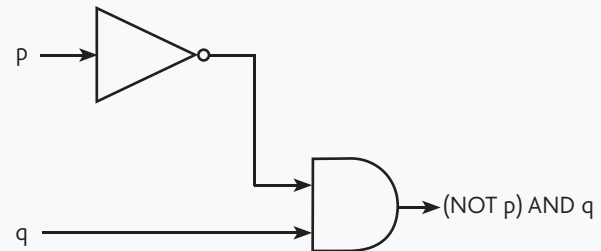


Fig. 1

Chapter 3: Software

Software is any program that runs on a computer. There are many different types of software including operating systems, games, word processing packages, virus checkers, spreadsheets, programming language translators (see *Chapter 7: Programming*) and many more. These are grouped into different categories. The two main categories are:

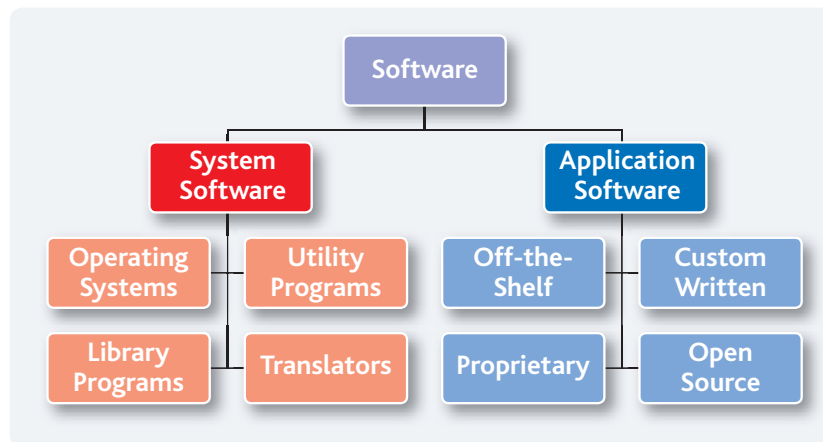
System software:

Programs that are needed to enable the computer to function, including the operating system, utilities, library routines and programming language translators.

Applications:

Programs that enable a user to perform a task: that is, something that the user needs to do with or without a computer such as writing a letter, keeping a set of class marks, recording attendance or processing orders.

To see the difference think, "Would I need to do this task if I didn't have a computer?" For example, if you had a computer you would probably use an email program or a word processing package to write a letter but if you did not have a computer, the task would still need to be done. The letter still needs to be written. On the other hand, if you did not have a computer, you would not need to do the tasks that an operating system, a programming language translator or a virus checker perform.



System software includes **operating systems** and **utilities**, both of which we look at in more detail later in this chapter. This category also includes the **translators** (compilers or interpreters) that convert the programs you write into machine code; each programming language has a compiler or interpreter, or both. **Library programs** are modules of code that can be used in your programs. For example, if you write a program to work in Windows you can build in standard functionality such as the Open File dialogue box rather than writing it yourself. In Delphi programs, statements like "Uses StringUtils" will import procedures that have already been written for you, in this case all the string handling functions. These are all library modules.

Application Software can be categorised in different ways. For GCSE Computing we will consider the different ways in which it can be sourced. (See Application Software on page 42.)

Operating Systems

The OCR Specification says that you should be able to:

- explain the need for the following functions of an operating system: user interface, memory management, peripheral management, multi-tasking and security

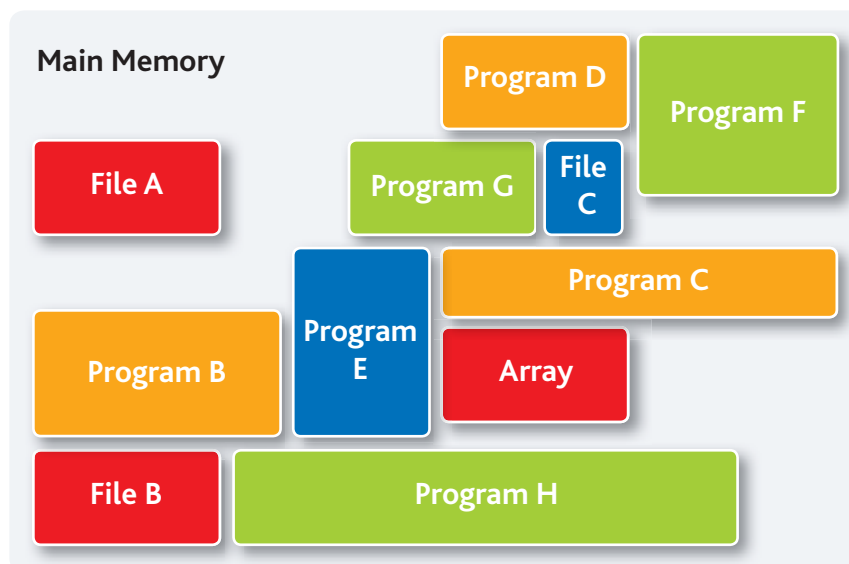
The operating system is system software. It is a group of programs that manages the computer's resources. These include the following functions:

- Memory management
- Peripheral management
- Multi-tasking
- Security
- Providing a user interface

Memory Management

When a program is running it must be in the computer's main memory, and the operating system must manage where in memory each program and the data it needs will go. Most computers are capable of holding several programs in memory at the same time, so that a user can switch from one application to another.

When you start up a program (say, a Python program or any application software such as Word or Excel), the memory manager allocates it adequate blocks of free space in main memory. It also allocates memory for any data file that you open such as the essay you are working on in Word. When the program finishes or no longer needs the data in previously allocated memory blocks, the memory management system frees up that space for reuse.



Multi-tasking / Process Management

A modern computer running many programs at once is **multi-tasking**. A user may have several documents open as well as various websites. The operating system will also be running background tasks just to manage the computer itself. The screenshot below shows that there were 10 applications open on my computer while I was typing this paragraph, but most of them were inactive. (A **process** is what we call a program when it is running in main memory.) There were also 48 background processes in memory.

Name	Status	12% CPU	63% Memory	0% Disk	0% Network
Apps (10)					
Adobe Acrobat (2)		0%	52.5 MB	0 MB/s	0 Mbps
Firefox		0%	121.5 MB	0 MB/s	0 Mbps
Google Chrome		0%	35.6 MB	0 MB/s	0 Mbps
Internet Explorer (2)		1.2%	274.3 MB	0 MB/s	0 Mbps
Microsoft Outlook		0%	62.5 MB	0 MB/s	0 Mbps
Microsoft Word		0%	104.9 MB	0 MB/s	0 Mbps
Skype		0.3%	52.3 MB	0 MB/s	0 Mbps
Spotify		0%	30.0 MB	0 MB/s	0 Mbps
Task Manager		0.6%	6.4 MB	0 MB/s	0 Mbps
Windows Explorer (4)		0.3%	22.5 MB	0 MB/s	0 Mbps
Background processes (48)					
AAM Updates Notifier Applicati...		0%	0.4 MB	0 MB/s	0 Mbps
Adobe Acrobat Update Service		0%	0.1 MB	0 MB/s	0 Mbps

A processor can only execute one instruction at a time (assuming it has a single-core processor). The operating system allocates a small amount of processor time (a few milliseconds) in turn to each active process in memory. However, if a running process stops to wait for user input, the processor immediately moves on to the next job.

All this happens so fast it appears as though all the processes are running at the same time. Some programs are clearly more urgent or important than others so there are priorities to manage as well. A program that deals with a hardware error, for example, will take priority over a word processing package trying to make text bold. While the processor is working on one job, other processes wait in main memory.

Peripheral Management

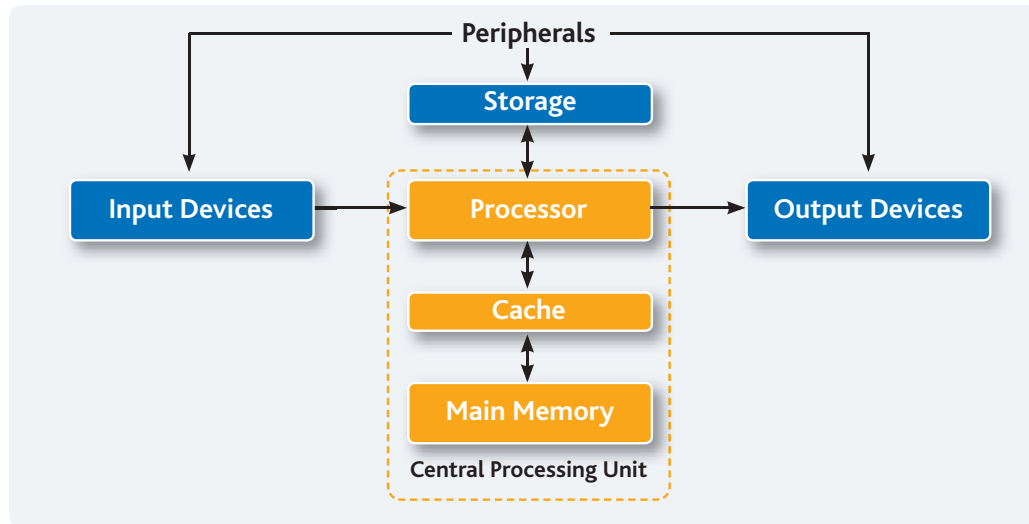
Peripherals are any computer hardware components that are not part of the CPU. This includes input, output and storage devices.



For you to do...

Type **Ctrl+Alt+Del** to look at the Task Manager on your computer and see how many processes are running.

For some of these, the term peripheral makes sense; the keyboard and monitor, for example, are outside the main computer casing, but storage is not so obvious. Although a hard disk is usually inside the computer casing, it is still considered a peripheral as it is outside the CPU (processor and main memory). Storage devices such as portable hard disks, memory sticks and CDs are also peripherals.



A function of the operating system is to manage these devices. When a user gives an instruction to print, the peripheral management function takes over and controls the sending of the data to be printed from memory to the device driver. (Each input or output device has its own driver – a small program that acts as an interface between the computer and the device. An HP printer, for example, will have different device drivers for a PC and a Mac, and the correct driver has to be installed so that the computer can communicate with the printer.) Meanwhile, the user can carry on editing their document in Word or whatever they were doing.

Security

Your phone, laptop, tablet or PC can all have passwords set to make them more secure. These can be set for different users on a PC but on a personal device like a phone the password is likely to be for the device. You can also password-protect individual files on the computer or set certain files as read-only for some users so they can view them but they cannot change them.

User name:

Password:

The type of security offered by the operating system will depend on the type of computer system. The operating system on a PC is very complex and has different types and levels of security to offer. The computer in a washing machine just has to make sure the “user” isn’t allowed to open the door when the machine is full of water.

Providing a User Interface

The user interface is the way in which we interact with computer hardware. We are all familiar with the way a PC works: clicking icons with the mouse, scrolling up and down pages, typing into forms, etc. This is called a **WIMP** user interface. WIMP stands for Windows, Icons, Menus and Pointers.

Mobile phones and tablet PCs have a slightly different user interface which allows you to move things by swiping the screen with your fingers and which can sense when you change the orientation of the device. Phones have controls that perform specific functions such as recording sound or taking a photograph.

Some computer systems are embedded in everyday machines such as cars and central heating controls. Users interact with these in different ways and the operating systems have to provide appropriate user interfaces.

Some computers have a **command-line interface**; no mouse or menus, just a text prompt where the user types a command. Before Windows came along with its **graphical user interface** (GUI) in the 1980s, PCs used this type of interface. Advanced computer users sometimes prefer a simple and direct means of controlling a program or operating system and MS Windows still provides access to a command line interface.



```
Command Prompt
C:\Users\Pat\Documents>cd database examples
C:\Users\Pat\Documents\database examples>dir
Volume in drive C is OS
Volume Serial Number is F8D8-3C60

Directory of C:\Users\Pat\Documents\database examples
13/09/2013  17:26    <DIR>          .
13/09/2013  17:26    <DIR>          ..
28/12/2010  10:08           143,360 appointments.mdb
28/12/2010  10:35           413,696 bookings.mdb
28/12/2010  10:33           229,376 DoctorAppointments.mdb
10/02/2011  22:21           589,824 Hairdresser_Database_2003.mdb
10/02/2011  22:01           753,664 Hairdresser_Database_2003_Backup.mdb
11/02/2011  17:05            33,280 Putting a command button on a form.doc
28/12/2010  10:52           573,440 room bookings.mdb
              7 File(s)          2,736,640 bytes
```

Whatever method is used for the user to communicate with a computer or computerised device, it is the operating system that provides these features.

Utility Programs

The OCR Specification says that you should be able to:

- describe the purpose and use of common utility programs for computer security (antivirus, spyware protection and firewalls), disk organisation (formatting, file transfer, and defragmentation), and system maintenance (system information and diagnosis, system cleanup tools, automatic updating)

Strictly speaking the operating system is the software that controls and manages the computer system but most operating systems also include programs called utilities. Utilities are not essential for the computer to work but either make it easier for the user to use in some way, or provide housekeeping functionality. We can categorise these utilities as follows:

- Security utilities that keep your computer safe from hackers and viruses.
- Disk organisation utilities that organise your files into folders and tidy up the disk.
- Maintenance utilities that perform system diagnostics and get software updates.

Security Utilities

Security is about keeping the computer system safe from hazards. Hazards come in many forms including viruses, hackers and spyware. To keep this in perspective though, the most common hazard a computer user has to face is their own carelessness in accidentally deleting files or putting them in the wrong place. The **backup utility** is probably the most important one of them all.

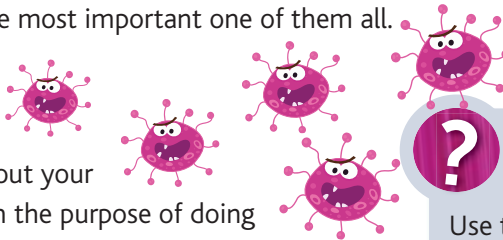
Antivirus software

A **virus** is a program that is installed on a computer without your knowledge or permission with the purpose of doing harm. It includes instructions to replicate automatically on a computer and between computers. Some viruses are just annoying and don't really do any damage but others will delete and/or change system files so that work files are corrupted or the computer becomes unusable.

Antivirus software will protect a computer in three ways:

- It prevents harmful programs from being installed on the computer
- It prevents important files, such as the operating system, from being changed or deleted
- If a virus does manage to install itself, the software will detect it when it performs regular scans. Any virus detected will be removed.

New viruses are created regularly so it is important that any antivirus software gets regular updates from the Internet.



For you to find out...

Use the internet to find out about the "I love you" virus, which wreaked havoc in 2000 and inspired a film in 2011.

How did it infect PCs around the world so quickly? Who did it affect?



Firewall

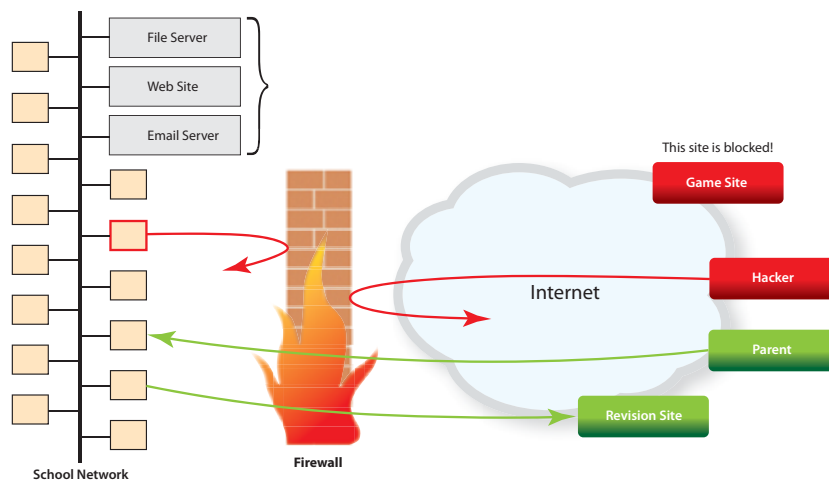
A computer connected to the Internet is potentially accessible to anyone else on the Internet. If a local area network, such as a school network, is connected to the Internet then all the file servers, the email server, the web server and all computers on the network are potentially accessible. Some people hack "just because they can" but often it is for identity theft or, for example, to get your bank account details so they can empty your account. Occasionally people hack with malicious intent to disrupt or destroy files or entire computer systems but this is less common.

A **firewall** is designed to prevent unauthorized access to or from a private network or intranet. All messages entering or leaving the intranet pass through the firewall, which examines each message and blocks those which do not meet specified security criteria.

Criteria may include for example:

- Where the access is from (the computer's address)
- The type of traffic (e.g. .exe files which may carry viruses)
- Specific web site addresses

A firewall doesn't just stop unwanted access from the outside world via the Internet; it can also stop computers on a network from accessing specific sites or categories of site on the network. This feature is used to stop staff in companies watching the cricket while they should be working, or from using social networking sites during working hours. In school you'll find that many sites have been blocked. Try going to a games website or getting to Facebook on a school computer and you will probably get a message saying that the site has been blocked. It is the firewall software that stops this traffic getting out of the local area network and onto the Internet.



Operating systems like MS Windows have firewall utilities included but you can also buy firewall software separately. Free firewall software can also be downloaded from the Internet and many banks provide free firewall software to customers using their Internet banking services.

Spyware protection

Programs that secretly record what you do on your computer are called spyware. The purpose of the software is to capture passwords, bank account details and keywords used in Internet shopping and banking. These details can then be used to make purchases on the Internet.

A spyware protection utility runs in the background on a PC. It detects spyware programs and prevents them from installing. It needs to regularly update itself from the Internet so that it can detect any new threats.

Some spyware is used for tracking and storing Internet users' activity on the web and then using this information to display pop-up advertisements on their screens next time they use the Internet. Parental control software may also use spyware-type programs.

Disk Organisation Utilities

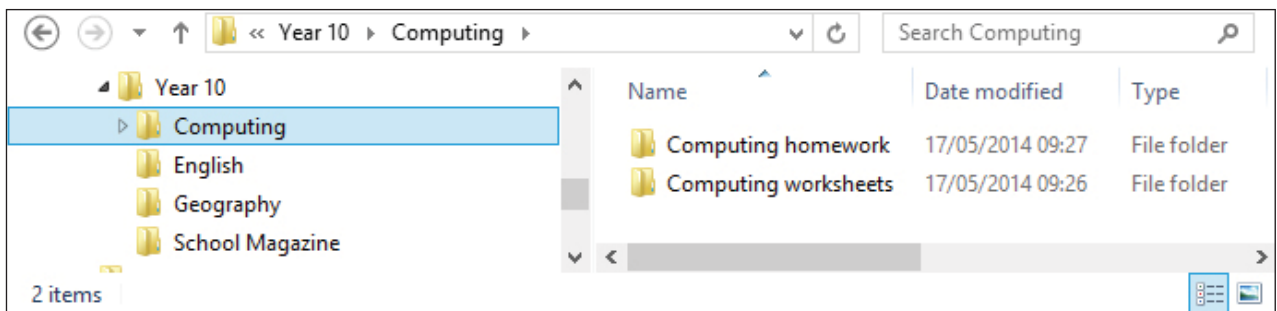
Formatting

All storage media (disks, memory sticks etc.) need to be formatted before they can be used by the operating system to store software and data. Basically this process involves marking the surface of the disk to indicate the start of each recording block. Blocks on a hard disk are now commonly 4096 bytes.

Portable devices are usually already formatted when you buy them so you don't have to format a floppy disk or USB memory stick, and hard disks in computers will be formatted by the operating system so that they are ready to store files in the way the operating system expects.

File transfer and file management

We take it for granted that we can organise our files into folders and easily move files around within the folder structure. This utility provides a logical view of how the files are organised to make it easier for the user. There are no little yellow folders actually on the disk, just lots of binary!



Disk defragmentation

The file management utility above makes the secondary storage look like a nicely organised filing cabinet but it doesn't really look like this. Files are stored on the hard disk in blocks wherever there is space. If you have a big file it might get split up into segments so it can be stored in the available gaps. This isn't very efficient because the operating system then has to keep track of where all the segments are. After a while thousands of files are stored in segments all over the disk. Files have become "fragmented".

The **disk defragmenter** moves the separate parts of the files around so that they can be stored together, which makes them quicker to access. The defragmenter also groups all the free disk space together so that new files can be stored in one place. This optimises disk performance.

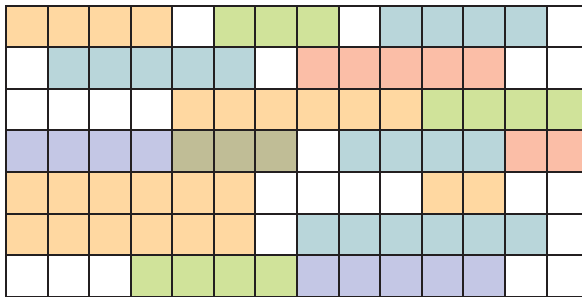


Interesting fact...

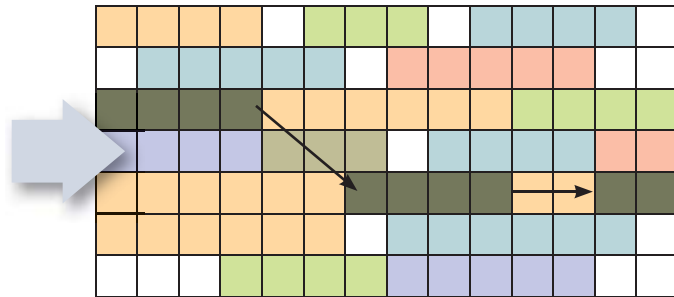
Reformatting a hard disk effectively deletes all the existing files from the disk.

However, until the space is overwritten by new files, some data will still be recoverable by specialists.

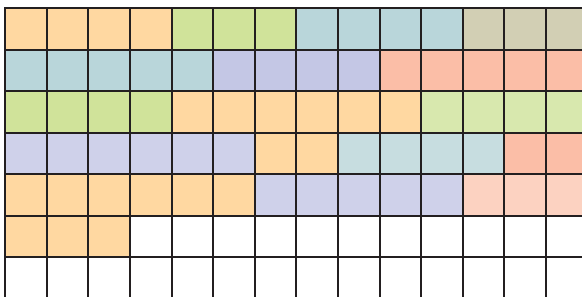
Before disk is defragmented, the disk contains lots of files, stored all over the disk:



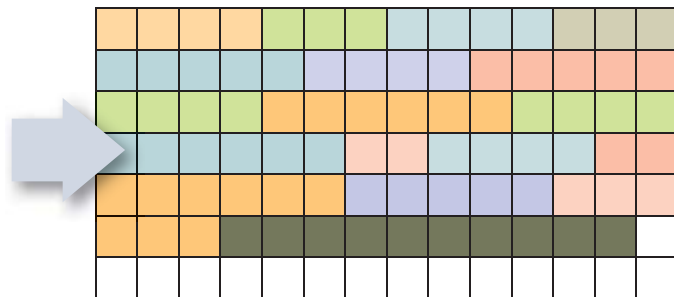
New file has to be saved in three different parts of the disk. Makes reading the file slower.



After defragmenting, the disk looks like this:



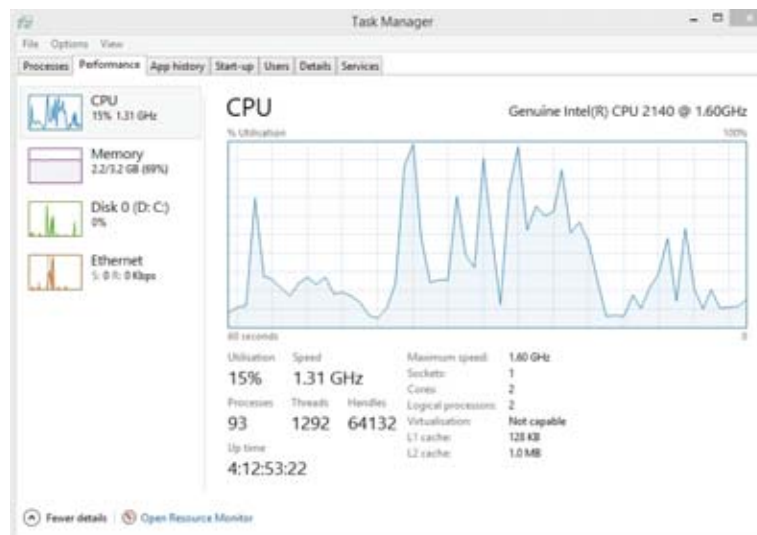
New file can be saved in one place so speeds up read access.



Maintenance Utilities

System information and diagnosis

This is a utility usually provided with the operating system. It gives information about the hardware, statistics about its use and information that will help diagnose any problems with the computer. Below is an example of a system information screen you might see:



This computer has not been switched off for 4 days, 12 hours, 53 minutes and 22 seconds.

System cleanup tools

The system cleanup utility searches for and deletes files that are no longer needed, for example, files that are used to install a software package. Once the package is installed these can be deleted. Also, many temporary files are created when files are moved around on the computer. All temporary files can be deleted to clean up the disk and improve performance.

Sometimes settings can be wrong or no longer needed. The system cleanup utility will correct errors in settings and delete ones that are no longer needed.

Using the cleanup tools regularly will

- free up disk space
- reduce the time it takes to boot up the system
- improve the performance of the computer

Automatic updating

The automatic update utility makes sure that any software installed on the computer is up-to-date. For any software already installed on the computer, the automatic update utility will regularly check the Internet for updates. These will be downloaded and installed if they are newer than the version already on the computer.

Firewalls and antivirus software must be updated regularly as new viruses and threats are constantly being devised and discovered.

Application software should also be updated as there will be bug fixes and improvements that become available to people with a licence for that package.

Stand-alone utility programs

As well as the utilities we take for granted on our PCs, there are utility programs that can be purchased separately to help manage the computer. The virus protection and firewall software that you use on your computer do not have to be the ones provided in the operating system. Many people use a separate package. There are also other utility programs such as compression software (e.g. WinZip) or software that converts different types of documents to other formats or enables you to read a PDF file (e.g. Adobe Acrobat).

Application Software

The OCR Specification says that you should be able to:

- discuss the relative merits of custom written, off the shelf, open source and proprietary software

This section looks at different ways of sourcing software. We can use software that has already been written for general use and there are different ways of accessing this. Some software is free and some software can be customised for our needs but a lot of software is quite expensive and there are usually limitations and restrictions on what we can do with it. If there isn't a package available to do exactly what we need, we can get software written specifically for our needs.

Open Source vs Proprietary

Open Source software is governed by the Open Source Initiative that says:

- Software is licensed for use but there is no charge for the licence. Anyone can use it.
- Open Source software must be distributed with the source code so anyone can modify it.
- Developers can sell the software they have created.
- Any new software created from Open Source software must also be "open". This means that it must be distributed or sold in a form that other people can read and also edit.

NB: This is different from **Freeware** (free software) which may be free to use but the user does not get access to the source code. Freeware usually has restrictions on its use as well.

Proprietary software is sold in the form of a licence to use it.

- There will be restrictions on how the software can be used, for example the licence may specify only one concurrent user, or it may permit up to say, 50 users on one site (site licence).
- The company or person who wrote the software will hold the copyright. The users will not have access to the source code and will not be allowed to modify the package and sell it to other people. This would infringe the copyright (Copyright, Designs and Patents Act).

The benefit of using proprietary software is the support available from the company. There will be regular updates available and technical support lines, training courses and a large user base. Open Source software tends to be more organic – it changes over time as developers modify source code and distribute new versions. There isn't a commercial organisation behind the software so there probably won't be a helpline or regular updates, just a community of enthusiastic developers.

Off-the-shelf vs Custom-written Software



Off-the-shelf software is exactly what it sounds like: you can go into a shop, pick it up off the shelf and buy it there and then. Off-the-shelf also applies to applications which you buy online as this is essentially the same procedure but without having to go to the shop. You can still get the software within a day or less. This is proprietary software, sold in the form of a licence to use it.



Interesting fact...

In 2014, there were 175 million Norton users.

As many people will buy the software you only have to make a tiny contribution towards the development costs. The company selling the software will make money because so many people buy it. Imagine how many people use Microsoft Word or have bought Norton Antivirus software.

Commonly used off-the-shelf applications are the everyday packages we use in school, for example, word-processing software, spreadsheets and email. Companies also buy off-the-shelf packages to perform common functions such as accounts, stock control, job costing and payroll.

Advantages of off-the-shelf software:

- You can buy it straight away
- Cheaper than custom-written software
- Lots of people use it so there will be plenty of support including help lines, books, user forums, Internet sites offering help, training courses
- Regular updates and bug fixes from the company

Disadvantages of off-the-shelf software:

- Made to suit many people so there may be many features that you never use, making the software more complicated than it needs to be
- May not do exactly what you want

Custom-written software is written for a customer to suit their specific requirements. It is not available to the general public. The company developing the software will analyse what the customer needs, design and make the software, and then deliver it. They must charge enough money to cover their development costs and make a profit so the customer will end up paying a lot more for custom software than buying an off-the-shelf package. This whole process takes time. You cannot order the software and expect it to arrive in the next couple of days. Custom software takes months or even years to develop.

Advantages of custom-written software:

- It does exactly what you want it to do. There might not be an off-the-shelf option for specialist requirements so custom-written software might be the only choice.
- No one else will have the software so it might mean your company can do something others can't or makes you more efficient than your competitors.

Disadvantages of custom-written software:

- You cannot have it straight away. The software may not be available for months or years.
- It is much more expensive than off-the-shelf software.
- No one else will be using this software so you can't go the local bookshop and buy a "How to..." guide.
- No regular updates and bug fixes to download – the company which wrote the software will charge a lot of money to upgrade the software.

Glossary of Terms

Software Categories

Software	Programs that run on the computer.
System Software	Programs that are used to run the computer, including the operating system, utilities, library routines and programming language translators.
Application Software	Programs that perform a task for a user, such as word processing, timetabling, accounts or payroll.
Operating System	Systems software that is necessary for the computer to function.
Utility Programs	Systems software that provides other useful functions for operating the computer or performing computer-related tasks such as anti-virus programs.
Library Software	Systems software modules that perform frequently required tasks. They can be built into or called from other programs.
Translators	System software that translates high-level programming languages into machine code. Includes compilers and interpreters. (An assembler translates low-level assembly code into machine code.)

Operating Systems

Memory Management	One of the main functions of the operating system – allocating space in main memory to all currently running programs and their associated data, and recycling the space when they have finished.
Peripheral	A device (hardware) that is connected to the CPU to provide input, output or storage
Peripheral Management	One of the main functions of the operating system – managing the input to and the output from the CPU.
Multi-tasking	One of the main functions of the operating system – managing how several tasks or programs, which are all running at the same time, share the processor.
Security	One of the main functions of the operating system – protecting the computer system from various hazards such as unauthorised users, viruses, hackers and accidental damage.
User Interface	The method of communication between the computer and its user. Sometimes called HCI (Human-Computer Interface) or MMI (Man-Machine Interface).
HCI	Human-Computer Interface: another term for user interface.

MMI	Man-Machine Interface: another term for user interface.
GUI	Graphical User Interface. A style of user interface, which is based on icons rather than text.
WIMP Interface	Stands for: Windows, Icons, Menus and Pointers. It describes a type of user interface where the user selects icons and menu items with a pointer of some kind (with a mouse, stylus or finger).
Command-Line Interface	A style of user interface that is only text-based. Commands are typed in at a text prompt.
Address	A numerical reference to a location in memory.
Process	A program that is running in main memory.

Utilities

Virus	A program that is installed on a computer without your knowledge or permission with the purpose of doing harm. It includes instructions to replicate automatically on a computer and between computers.
Antivirus Software	A utility program that prevents harmful programs being installed and important files being changed. The antivirus software scans all the files on the computer periodically and if a virus does install itself, it detects and removes it.
Firewall	A utility program that prevents unauthorised access to computers or a LAN from the Internet and controls what sites computers on the LAN can access.
Spyware	A program that secretly records the user's actions on the computer including passwords and personal details they type when accessing a secure site.
Disk Defragmenter	A utility program that optimises the use of the hard disk space by collecting together the separate segments of each file into contiguous blocks (i.e. blocks next to each other) on the disk as well as grouping together the blocks of free space so newly saved files do not have to be fragmented (split up).
System Cleanup	A utility program that deletes unnecessary files and settings to optimise the computer's performance.
Automatic Update	A utility program that, for any software already installed on the computer, will regularly check on the Internet for newer versions and updates, then download and install them.
System Information and Diagnosis	A utility program that presents information about the computer hardware and usage as well as information to help diagnose problems.

Formatting A utility program that formats secondary storage devices such as hard disks, preparing them for use.

File Transfer and File Management A utility program that allows the user to create a logical view of how their files are organised using folders. Allows the user to move files and folders, copy, paste, name and delete files and folders.

Applications

Open Source Software that is supplied with its source code. The source code can be modified and the software redistributed (as long as this too is open source).

Licence An agreement that defines the conditions for using the software.

Freeware Software that is provided free of charge under licence but without the source code. It is copyrighted and cannot be modified.

Proprietary Software Software that is copyrighted and the licence sold under a patented or trademarked name.

Off-the-Shelf Software Software that can be purchased from a High Street or online store. Not custom-written.

Custom-Written Software Software that is written for one customer's specific requirements.

Past Exam Questions



June 2011, Question 5

- 1 Describe the following types of common utility programs.
- (a) Antivirus [2]
 - (b) Disk defragmenter [2]

Jan 2012, Question 2

- 2 One of the functions of an operating system is multi-tasking.
- (a) Explain one reason why multi-tasking is needed in an operating system. [2]
 - (b) State two other functions of an operating system. [2]

June 2013, Question 9

- 3 A school uses off the shelf, proprietary software for managing pupil's attendance, and customised, open source software for managing pupils' examinations.
- (a) Describe the difference between off the shelf and custom software. [2]
 - (b) Describe the difference between proprietary and open source software. [2]

Chapter 4: Data Representation in Computers

This topic is all about how data is stored in a computer. In this chapter you will learn how computers store numbers, text, images, sounds and program instructions.

Units

The OCR Specification says that you should be able to:

- define the terms bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte
- understand that data needs to be converted into a binary format to be processed by a computer

Computers are made up of complicated hardware that stores and processes data. If you break a computer down into its most basic components you have millions of circuits that either allow electricity to flow, or not. Imagine a whole row of light switches that you can switch on and off in different combinations to mean different things. Each switch is either on or off. It only has two states, which can be represented by 1 or 0. This is called **binary**.



A single **1** or **0** is a **binary digit**, or a **bit** for short. A group of eight bits is called a **byte**. Imagine you've taken a small bite out of an apple, you might call that a nibble. So four bits, half a byte, is called a **nibble**!

Just as a kilometre is 1000 meters, we can group together 1024 bytes to make a **kilobyte**.

8 bits	=	1 byte
1024 bytes	=	1 kilobyte
1024 kilobytes	=	1 megabyte
1024 megabytes	=	1 gigabyte
1024 gigabytes	=	1 terabyte



For you to find out...

- 1 Why isn't a kilobyte 1000 bytes?
1024 seems a strange number. How does it relate to binary?
- 2 What comes after terabytes?

Numbers

The OCR Specification says that you should be able to:

- convert positive denary whole numbers (0-255) into 8-bit binary numbers and vice versa

Imagine you are back in primary school, learning to add again. $7+5 = 12$, so you write down the 2 units but carry the group of 10.

23 would be 2 groups of 10 and 3 units.

Counting in binary is the same except instead of digits 0 to 9 we only have two digits, 0 and 1, so we carry the group of 2. In Maths we call this Base 2. This is how we count to 10 in binary:



For your information...

Denary is our ordinary number system, which has 10 digits 0-9

denary	binary	
0	0	
1	1	
2	10	Notice that we now go to the second column – one group of 2, no units
3	11	One group of 2 plus one unit $2+1=3$
4	100	Now we go to the third column, 2 groups of previous column, so this is 4
5	101	
6	110	
7	111	
8	1000	Every time we go to the next column it is two times the previous column
9	1001	
10	1010	

Can you see the pattern? The column headings in a binary number double each time:

	$\times 2$	$\times 2$	$\times 2$	$\times 2$	$\times 2$	$\times 2$	$\times 2$	
	128	64	32	16	8	4	2	1
	$2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$	$2 \times 2 \times 2 \times 2 \times 2 \times 2$	$2 \times 2 \times 2 \times 2 \times 2$	$2 \times 2 \times 2 \times 2$	$2 \times 2 \times 2$	2×2	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

To convert the binary number 111001 into a denary (decimal or base 10) number use the column headings. The number below is $32+16+8+1 = 57$ (add up the column headings where there is a 1).

128	64	32	16	8	4	2	1
0	0	1	1	1	0	0	1

To convert a denary number to a binary number, use the column headings. You need to find the biggest column heading that you can take away from the number and start there:

Let's convert 57 into binary:

The biggest column heading we can take out of 57 is 32 (the next one is 64, which is too big).

Write a 1 under column heading 32. That leaves us with $57 - 32 = 25$.

Write a 1 under column heading 16 (because we can take 16 out of 25). $25 - 16 = 9$

9 is an 8 and a 1, so put 1 under each of these column headings

128	64	32	16	8	4	2	1
0	0	1	1	1	0	0	1

Always double check by adding the columns up at the end. They should give you the number you started with! $32+16+8+1 = 57$ ✓

Here are some more examples:

	Binary							
Denary	128	64	32	16	8	4	2	1
23	0	0	0	1	0	1	1	1
84	0	1	0	1	0	1	0	0
255	1	1	1	1	1	1	1	1

So far we have only looked at numbers using eight columns. This is an 8-bit number, or a byte.

00000011 is binary for 3 but so is 11. You do not need the leading zeros for it to be a valid number but we tend to write groups of 8 bits because computers usually store data in bytes.

Obviously numbers can be bigger than 255 but in computers we need to represent data in a specific number of bits.

For example, in some programming languages an integer is stored in 16 bits (2 bytes) whereas a real number is stored in 4 bytes.

At GCSE level you will only need to work with 8-bit numbers.

! For your information...

255 is a significant number in binary; it will come up a lot in other parts of the course!

What do you notice about the number 255 in binary?

The OCR Specification says that you should be able to:

- add two 8-bit binary integers and explain overflow errors which may occur

Adding binary works in exactly the same way as adding denary numbers except you carry groups of 2 instead of groups of 10:

Adding denary	Adding binary
$\begin{array}{r} 1\ 2\ 3\ 4\ 5 \\ \ 1\ 3\ 4\ + \\ \hline 1\ 2\ 4\ 7\ 9 \end{array}$	$\begin{array}{r} 1\ 0\ 0\ 1\ 0 \\ \ 1\ 0\ 1\ + \\ \hline 1\ 0\ 1\ 1\ 1 \end{array}$
$\begin{array}{r} 7\ 8\ 2\ 3\ 5 \\ \ 1\ 9\ 1\ 7\ + \\ \hline 7\ 8\ 3\ 3\ 2 \end{array}$	$\begin{array}{r} 1\ 0\ 0\ 1\ 1 \\ \ 1\ 1\ 1\ 1\ + \\ \hline 1\ 1\ 0\ 1\ 0 \end{array}$
<p><i>Notice that you carry 1 when you get to ten in a column so $5+7=12$, write 2 in that column but carry one group of ten.</i></p>	<p><i>Notice that you carry 1 when you get to two in a column, so $1+1=2$, write 0 in that column but carry one group of two.</i></p> <p><i>In the second column, $1+1+1=3$, carry one group of 2 to the third column.</i></p>

Some more examples:

$$\begin{array}{r} 10101100 \\ 00010001+ \\ \hline 10111101 \end{array} \quad \begin{array}{r} 00101101 \\ 10000101+ \\ \hline 10110010 \end{array} \quad \begin{array}{r} 00101101 \\ 10000111+ \\ \hline 10110100 \end{array}$$

The biggest number you can represent with 8 bits is 255 ($128+64+32+16+8+4+2+1$).

If you add two binary numbers together that result in a number bigger than 255, it will need 9 bits. A computer stores things in memory in a finite amount of space. If you cannot represent the number in that amount of space because it is too big, then we get **overflow**.

For example:

$$\begin{array}{r} (252) \quad 11111100 \\ (15) \quad 00001111+ \\ \hline (267) \quad 100001011 \end{array}$$

The computer would need 9 bits to represent 267 so this 9th bit doesn't fit in the byte allocated. This is what is meant by **overflow**.



For you to find out...

What is the biggest decimal number you can store using:

- 4 bits?
- 2 bytes?

The OCR Specification says that you should be able to:

- convert positive denary integers (0-255) into 2-digit hexadecimal numbers and vice versa
- convert between binary and hexadecimal equivalents of the same number
- explain the use of hexadecimal numbers to represent binary numbers

Which of these is easier to remember: **01011011** or **5B**? Humans are not very good at remembering long strings of numbers so, to make it easier, we can represent every group of 4 bits (a nibble) with a single digit.

The smallest value you can have with 4 bits is 0000. The largest value is 1111. This means that we need to represent the denary values 0 to 15 with a single digit. The trouble is, we only have numerical digits 0 to 9, so to get around this problem we use letters to represent the digits 10, 11, 12, 13, 14 and 15.

This is called Base 16 in Maths, or **hexadecimal** in Computing. We abbreviate this to **hex**.

This is how we count to 16 in denary, binary and hex:

denary	binary	Hex
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
255	1111 1111	FF



For you to find out...

How is hex used to code colours in HTML (web pages)?
For example, what colour is #FF00FF?

Search on "Colour palette hex" and see what you can find out.

A single hex digit replaces 4 bits.

15 is the biggest number you can have with 4 bits so 16 is one group of 16 and no units (just like we did before with binary).

255 is 15 groups of 16 + 15 units
i.e. $(15 \times 16) + 15$

To Convert a Binary Number to Hex:

In GCSE Computing you will only need to work with 8-bit binary numbers, which can be represented as two hex digits. The first hex digit represents groups of 16, the second hex digit represents the units.

The denary number 92 = $\frac{0101}{5}$ $\frac{1100}{12}$ = 5C in hex
 (groups of 16) (units) *(12 is replaced by C)*

To Convert a Denary Number to Hex:

To convert the denary number 182 into hex the first step is to work out how many groups of 16 there are in 182. Secondly work out how many units are left over.

$$182 / 16 = 11 \text{ remainder } 6$$

11 is B in hex. 6 is just 6

So 182 denary = B6 hex

Alternatively, you can convert the denary to binary first and then convert the binary to hex, as above.

Characters

The OCR Specification says that you should be able to:

- explain the use of binary codes to represent characters
- explain the term character set
- describe with examples (for example ASCII and Unicode) the relationship between the number of bits per character in a character set and the number of characters which can be represented

Every time a character is typed on a keyboard a code number is transmitted to the computer. The code numbers are stored in binary. Different sets of codes are available for different types of computer. PCs use a **character set** called **ASCII**, American Standard Code for Information Interchange. A character set is the group of characters that can be coded.

The next page shows a version of ASCII that uses 7 bits to code each character. The biggest number you can have with seven bits is 1111111 in binary (127 in denary). The smallest number you can have with seven bits is 0000000 (0 in denary). This means that you can have 128 different characters in the character set (using codes 0 to 127).

Other character encoding systems include:

- **Unicode:** An encoding system that typically has 16 or 32 bits per character so can code 2^{16} (65,536) or 2^{32} (4,294,967,296) different characters in its character set.
- **EBCDIC**(*pronounced eb-sid-ic*): Extended Binary Coded Decimal Interchange Code, an 8-bit encoding system that has 2^8 (256) different characters in its character set.

7-bit ASCII Table

ASCII	DEC	Binary	ASCII	DEC	Binary	ASCII	DEC	Binary	ASCII	DEC	Binary
NULL	000	000 0000	space	032	010 0000	@	064	100 0000	˘	096	110 0000
SOH	001	000 0001	!	033	010 0001	A	065	100 0001	a	097	110 0001
STX	002	000 0010	"	034	010 0010	B	066	100 0010	b	098	110 0010
ETX	003	000 0011	#	035	010 0011	C	067	100 0011	c	099	110 0011
EOT	004	000 0100	\$	036	010 0100	D	068	100 0100	d	100	110 0100
ENQ	005	000 0101	%	037	010 0101	E	069	100 0101	e	101	110 0101
ACK	006	000 0110	&	038	010 0110	F	070	100 0110	f	102	110 0110
BEL	007	000 0111	'	039	010 0111	G	071	100 0111	g	103	110 0111
BS	008	000 1000	(040	010 1000	H	072	100 1000	h	104	110 1000
HT	009	000 1001)	041	010 1001	I	073	100 1001	i	105	110 1001
LF	010	000 1010	*	042	010 1010	J	074	100 1010	j	106	110 1010
VT	011	000 1011	+	043	010 1011	K	075	100 1011	k	107	110 1011
FF	012	000 1100	,	044	010 1100	L	076	100 1100	l	108	110 1100
CR	013	000 1101	-	045	010 1101	M	077	100 1101	m	109	110 1101
SO	014	000 1110	.	046	010 1110	N	078	100 1110	n	110	110 1110
SI	015	000 1111	/	047	010 1111	O	079	100 1111	o	111	110 1111
DLE	016	001 0000	0	048	011 0000	P	080	101 0000	p	112	111 0000
DC1	017	001 0001	1	049	011 0001	Q	081	101 0001	q	113	111 0001
DC2	018	001 0010	2	050	011 0010	R	082	101 0010	r	114	111 0010
DC3	019	001 0011	3	051	011 0011	S	083	101 0011	s	115	111 0011
DC4	020	001 0100	4	052	011 0100	T	084	101 0100	t	116	111 0100
NAK	021	001 0101	5	053	011 0101	U	085	101 0101	u	117	111 0101
SYN	022	001 0110	6	054	011 0110	V	086	101 0110	v	118	111 0110
ETB	023	001 0111	7	055	011 0111	W	087	101 0111	w	119	111 0111
CAN	024	001 1000	8	056	011 1000	X	088	101 1000	x	120	111 1000
EM	025	001 1001	9	057	011 1001	Y	089	101 1001	y	121	111 1001
SUB	026	001 1010	:	058	011 1010	Z	090	101 1010	z	122	111 1010
ESC	027	001 1011	;	059	011 1011	[091	101 1011	{	123	111 1011
FS	028	001 1100	<	060	011 1100	\	092	101 1100		124	111 1100
GS	029	001 1101	=	061	011 1101]	093	101 1101	}	125	111 1101
RS	030	001 1110	>	062	011 1110	^	094	101 1110	~	126	111 1110
US	031	001 1111	?	063	011 1111	_	095	101 1111	DEL	127	111 1111

The characters are in numerical sequence, i.e. if "A" is 65 then "C" must be 67.

Also, they have an order so you can say that "7" < "9" and that "a" > "A".

Images

The OCR Specification says that you should be able to:

- explain the representation of an image as a series of pixels represented in binary
- explain the need for metadata to be included in the file such as height, width and colour depth
- discuss the effect of colour depth and resolution on the size of an image file

Images can be stored in different ways on a computer. A drawing that you create in PowerPoint is a **vector** graphic. It is made up of lines and shapes with specific properties such as line style, line colour, fill colour, start point and end point. The computer stores all of this data about each shape in binary.

When you take a photograph on a digital camera, the image is not made up of individual shapes. The picture somehow has to capture the continuously changing set of colours and shades that make up the real-life view. To store this type of image on a computer the image is broken down into very small elements called **pixels**. A pixel (short for picture element) is one specific colour. The whole image may be, for example, 600 pixels wide by 400 pixels deep. 600 x 400 is referred to as the picture's **resolution**. If the resolution of a picture is increased, then more pixels will need to be stored. This increases the size of the image file.

Making an image file

10	10	10	10	10	10	10	10	10
10	00	10	00	10	00	10	10	10
10	10	00	00	00	10	10	10	10
10	00	00	01	00	00	10	10	10
10	10	00	00	00	10	10	10	10
10	00	10	00	11	00	10	10	10
10	10	10	10	11	10	10	10	10
10	10	10	10	10	11	10	10	10
10	11	11	11	10	11	11	10	10
10	10	11	11	11	11	10	10	10
10	10	10	10	11	10	10	10	10

This image of a flower uses 4 colours. Therefore 2 bits are needed to record the colour of each pixel:

11	10	01	00
Green	White	Yellow	Pink

The number of bits used to store each pixel dictates how many colours an image can contain. 8 bits per pixel will give 256 possible colours. The number of bits per pixel is referred to as the **colour depth**.

If the colour depth is increased so more bits are used to represent each pixel, then the overall size of the file will increase.

If we record the value of each pixel in this image, starting from the top left-hand corner and going left to right across each row, we end up with the following data file:

```
10 10 10 10 10 10 10 10 10 00 10 00 10 00 10 10 10 10 00
00 00 10 10 10 10 00 00 01 00 00 10 10 10 10 00 00 00 10
10 10 10 00 10 00 11 00 10 10 10 10 10 11 10 10 10 etc
```

For the computer to interpret this file and rebuild the picture it must know some other things about the data file; for example, that the picture's resolution is 8x11 pixels and the colour depth is 2 bits per pixel. Data about the data file itself is called **metadata**.

Sound

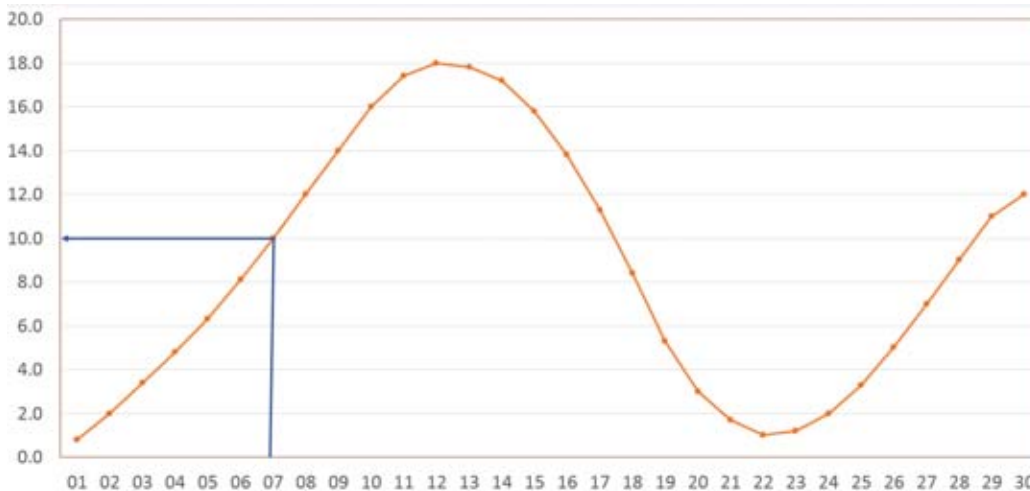
The OCR Specification says that you should be able to:

- explain how sound can be sampled and stored in digital form
- explain how sampling intervals and other considerations affect the size of a sound file and the quality of its playback

Sound waves are **analogue**, which means continuously changing. Anything stored on a computer has to be stored in a **digital** format as a series of binary numbers. To store sound on a computer we need to convert the waveform into a numerical representation. The device that takes real-world analogue signals and converts them to a digital representation is called an Analogue-to-Digital Converter (**ADC**).

For sound waves, the analogue signal is converted as follows:

- Measure the amplitude (height of wave) at regular intervals (sampling)
- Store the values as a series of binary numbers in a file as shown on the right



01	0.8
02	2.0
03	3.4
04	4.8
05	6.3
06	8.1
07	10.0
08	12.0
09	14.0
10	16.0
11	17.4
12	18.0
13	17.8
14	17.2
15	15.8
16	13.8
17	11.3
18	8.4
19	5.3
20	3.0
21	1.7
22	1.0
23	1.2
24	2.0
25	3.3
26	5.0
27	7.0
28	9.0
29	11.0
30	12.0

Sound Quality is affected by:

- **Sample Resolution:** The number of bits used to store each sample.
The more bits that are used, the better the accuracy of the sound file. In the graph of a sound wave above, the sample resolution on the Y axis is shown in denary to 1 decimal place. It could be more accurately represented with 2 or more decimal places. The same holds true in binary.
- **Sample Interval:** The time period between taking samples/measurements.
The more frequently the sound is sampled, the better the quality of playback. In the graph above, samples are taken every 1 unit of time – if we halved the sample interval, the wave would be more accurately represented.

Synthesis:

Sound synthesis is when the sound is recreated from this file by the computer, and played through speakers.

Instructions

The OCR Specification says that you should be able to:

- explain how instructions are coded as bit patterns
- explain how the computer distinguishes between instructions and data

Computer programs are made up of instructions. If you write a program in Delphi, for example, you use commands like `write` and `read`, or `Total := Mark1 + Mark2`. When you **compile** your program the computer converts the high level programming language into **machine code**, which is a binary representation of the instructions you typed. Each type of processor has its own type of machine code – a specific set of instructions that it can execute. This is called an **instruction set**.

Instruction Set

Imagine a simple game on a phone that involves getting a character to run around a maze and jump over obstacles on the way.

The **instruction set** for this game may have the following commands:

- RIGHT (Turn 90° to the right)
- MOVE (Move forward one step)
- LEFT (Turn 90° to the left)
- JUMP (Jump forward three steps)

RIGHT	00
MOVE	01
LEFT	10
JUMP	11

In machine code we replace each action (**operation**) with a unique numerical code. As this instruction set only contains four instructions, we can number them 0 to 3. This means that the computer could represent each command with a sequence of two bits as shown here.

The group of bits that represents the operation is called the **opcode** (short for operation code). If a particular processor had an instruction set that used 8 bits for the opcode, then there could be as many as 256 different instructions.

If the programming language allowed us to specify how many steps to move, or what angle to turn through, we could have instructions like this:

- RIGHT 90 (Turn 90° to the right)
- MOVE 2 (Move forward 2 steps)

In the program the number of degrees and the number of steps might be stored as variables in main memory so the instructions may look like this:

- RIGHT NumDegrees
- MOVE NumSteps

In this case the NumDegrees and NumSteps are variables, the data that the operation (e.g. RIGHT) will use. When the program is compiled, the compiler replaces the variable name in the machine code instruction with the actual memory address.

The instruction has two main parts to it:

- an **opcode**, which is the binary number representing the operation to be carried out
e.g. RIGHT, MOVE, ADD, LOAD, STORE
- the **operand** (data) that the operation will use. The data can be an actual value (e.g. 5) or an address in memory where the data can be found (a variable). Different opcodes will be used depending on whether the operand is an actual value or an address.

Instruction or data?

How does the computer know which memory locations hold instructions and which ones hold data?

Every location in memory has a unique address. When a program is compiled, the resulting machine code instructions might occupy, for example, 50 memory addresses 0 to 49 and all the variables used in the program will occupy adjacent memory locations following the instructions, for example locations 50 to 57 if eight variables are used.

The operating system will load the program into free space in memory (for example location 2,578,000 or some other location) and load the start address into the processor. The processor will continue executing instructions until it reaches a STOP or END statement in the program, or until some error or user intervention occurs. The operand in each instruction will specify where to look for the data, for example a variable in one of the relative addresses 50 to 57.

So, in summary, the operating system knows where it has loaded the program, and the program instructions specify where the data is held. Each time the program is run, it may have a different start address, but relative to this, its data will always be held in the same locations after the instructions.

Programs may use data files as well as variables. These files will be brought into main memory while the program is running. The operating system is responsible for managing main memory and will keep track of where each file is located.

Glossary of Terms

Units

Bit	A single binary digit: 1 or 0
Byte	8 bits
Nibble	4 bits
Kilobyte	1024 bytes / 2^{10} bytes
Megabyte	1024 kilobytes / 2^{20} bytes
Gigabyte	1024 megabytes / 2^{30} bytes
Terabyte	1024 gigabytes / 2^{40} bytes

Numbers

Binary	Base 2 number system, used by computers, uses the digits 0 and 1 only.
Denary/Decimal	Base 10 number system, how we normally count, uses digits 0 to 9.
Hexadecimal (hex)	Base 16 number system used by humans to represent groups of four bits at a time. Uses digits 0 to F.
Overflow	When the result of a numeric calculation is too large to be stored in the space reserved for that type of data.

Characters

Character set	The set of symbols that can be represented by a computer. The symbols are called characters and can be letters, digits, space, punctuation marks and some control characters such as "Escape". Each character is represented by a numerical code that is stored as a binary integer.
ASCII	American Standard Code for Information Interchange: a 7-bit character set used by PCs. (There is also an extended ASCII character set that uses 8 bits.)
EBCDIC (pr. eb-sid-ic)	Extended Binary Coded Decimal Interchange Code: an 8-bit character set used by older mainframes.
Unicode	A 16-or 32-bit character set that allows many more characters to be coded.

Images

Bitmap Image	An image that has been stored as a series of values per pixel. The colour of each individual pixel is stored in a file.
Vector Graphic	An image file that is made up of lines and shapes that have certain properties, for example, a line may have the following properties: start-point, end-point, line colour, line thickness, line style. The properties of each shape are stored in a file to make up the image.
Pixel	Short for picture element. It is the smallest component of a bitmapped image.
Colour Depth	The number of bits used to represent the colour of a single pixel in a bitmapped image. Higher colour depth gives a broader range of distinct colours. For example, an image stored as a .gif file uses 8 bits per pixel so the image could use 256 different colours.
Resolution	The number of pixels in an image expressed as: the-number-of-pixels-across x the-number-of-pixels-down e.g. 400 x 600. Effectively this describes the pixel density.
Metadata	Data about data. In the case of image files metadata is the data the computer needs to interpret the image data in the file, for example: resolution, colour depth and image dimensions.

Sound

Analogue	A continuously changing wave such as natural sound.
Digital	Data that is made up of separate values. How data is stored on a computer.
Sample Rate	The number of times per second that the sound wave is measured. The higher the rate the more accurately the sound wave is represented.
Sample Interval	The time gap between measurements of the sound wave being taken. Another way of expressing the sampling rate.
Sample Resolution	The number of bits used to store the value of each sample. The greater the number of bits the more accurately the value is stored.
ADC	Analogue to Digital converter: takes real-world analogue data and converts it to a binary representation that can be stored on a computer.

Instructions

Instruction Set	The group of instructions available for a specific processor to use. The number of instructions available will depend on the number of bits used. For example, with 4 bits there could potentially be 16 different instructions.
Opcode	The group of bits in a machine code (binary) instruction that represents the operation (instruction) such as EAT, MOVE or TURN.
Operand	A data value or an address that is part of a machine code instruction.
Compiler	Systems software that converts a program written in a high level programming language into machine code (binary).
Machine Code	Program instructions that have been converted into a form that the computer can execute. A machine code instruction typically has an opcode and an operand in binary.
High Level Programming Language	A programming language written in constructs using language we can understand. Languages include Delphi, Visual Basic, Java and C++.

Past Exam Questions



June 2013, Question 5

- 1 Numbers can be represented in denary, binary or hexadecimal.
- (a) (i) Convert the binary number 01101001 to denary, show your working. [2]
 (ii) Convert the denary number 154 to binary [2]
- (b) The security code for an alarm system is a long binary number which begins
 10001111100101111011 ...
- The technicians prefer to use hexadecimal to enter the security code.
- (i) When the number is converted into hexadecimal, the first two digits are 8F as shown below.
 Complete the gaps to show the next three digits.
- | | | | | | | |
|--------------|------|------|-------|-------|-------|-----|
| Binary: | 1000 | 1111 | 1001 | 0111 | 1011 | |
| Hexadecimal: | 8 | F | | | | [3] |
- (ii) Explain why the technicians prefer to use hexadecimal. [2]

June 2012, Question 6

- 2 (a) Convert the denary number 55 to an 8 bit binary number. [2]
 (b) Convert the denary number 55 to hexadecimal. [2]



Jan 2012, Question 5

3 Peter takes a picture of himself and his friends to put on a social networking site. The picture is converted into pixels and stored as a bitmap file.

(a) Tick one box in each row to show whether or not each of the following items must be included in the bitmap file.

	Must be included	Need not be included
The names of the people in the picture		
The width of the picture in pixels		
The number of bits used for each pixel		
The number of people in the picture		
The colour of each pixel		

(b) (i) What is meant by the resolution of the picture? [1]

(ii) How does the resolution affect the size of the bitmap file? [2]

Jan 2013, Question 6

4 When recording a sound file on a computer, the sound needs to be sampled.

(i) Describe how sampling is used when storing sound. [2]

(ii) Explain the effect of the sampling interval on the size and quality of the sound file recorded. [3]

Chapter 5: Databases

This topic looks at the basic concept of a database, the key features and benefits of a Database Management System (DBMS) and the basic theory behind relational databases such as Microsoft Access. Using Microsoft Access is a good way to get practical experience of relational database concepts and the features of a DBMS. This chapter does not provide instruction on how to use any one DBMS package.

The Database Concept

The OCR Specification says that you should be able to:

- describe a database as a persistent organised store of data
- explain the use of data handling software to create, maintain and interrogate a database

Many organisations keep large amounts of data. A company stores data about its customers and staff, schools store data about the students, supermarkets store data about stock levels and customer buying patterns. This data could be stored in books, card files or spreadsheets, depending on the volume and type of data. When there are large quantities of data an organisation is most likely to use a database.

A **database** is described as “**a persistent organised store of data**”. Let’s look at each part of that definition:

- **“Store of data”**
Clearly lots of data are being stored in a database.
- **“Organised”**
A database is more than just a store of data. A graphics file or collection of post-it notes on a wall is a “collection of data”. In a database, the data is organised into records. A school database, for example, will have a record for each student.
- **“Persistent”**
This means that a database is a non-volatile store of data on a secondary storage medium such as a hard disk. This is in comparison to an array, which could also be described as an organised store of data but only exists in memory while a program is running. All the data in a database is stored on a disk and can be accessed by programs which use and maintain it (i.e. keep it up to date).



For your information...

Data is strictly speaking a plural term so “...data are...” is correct grammar. The singular is “datum”, but this is seldom used so you will often see “data is...”

Data Handling Software

Data handling software can be created using an off-the-shelf product such as Microsoft Access or MySQL. Alternatively you can create the database itself in one of these products and then use a high level programming language such as Delphi or Visual Basic to create a customised, forms-based, front-end application that does exactly what is needed.

A database must first be created. The data must be gathered and organised in some way. Consider a company that wants a database for all its products. They will need a **record** for each product that holds all the relevant details about that product. The specific details will be stored as **fields**. Fields for a product database might include ProductID, Description, Supplier, Price, NumberInStock, etc.

A spreadsheet table is a basic database and could look like this:

	A	B	C	D	E
1	ProductID	Description	Supplier	Price	NumberInStock
2	0123	Humbugs, 250g bag	Smith & Sons	£1.45	200
3	0263	Toffee, small slab	Old Fashioned Toffee	£1.75	34
4	0462	Lemon sherbets, 500g bag	Smith & Sons	£2.50	128
5	1234	Lemon sherbets, 250g bag	Smith & Sons	£1.75	134



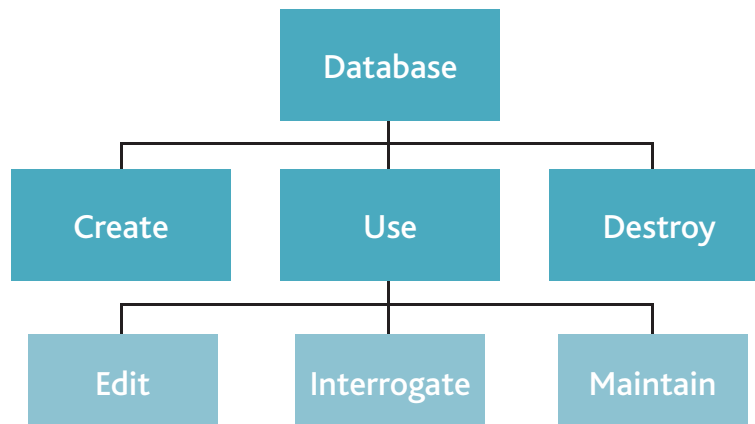
For your information...

Note the lack of spaces in these field names – just like in programming, it is considered good practice to use PascalCase to name fields (see Chapter 7: Programming).

This database contains the basic information that the company needs about product pricing and stock levels. The applications that use this data will **interrogate** the database. Interrogate means to ask questions about the data or to **query** it. For example, how many products have a stock level below 50 and need reordering? Or, how much does a bag of humbugs cost?

The quality of the information you get from a database application is only as good as the data you put in: GIGO is an acronym for Garbage In, Garbage Out. A database must be **maintained**. This means that when the data changes in the real world, the database must be updated to make sure the data stored is correct and up-to-date.

A database can be considered in terms of its life history and the operations that can be performed on it:



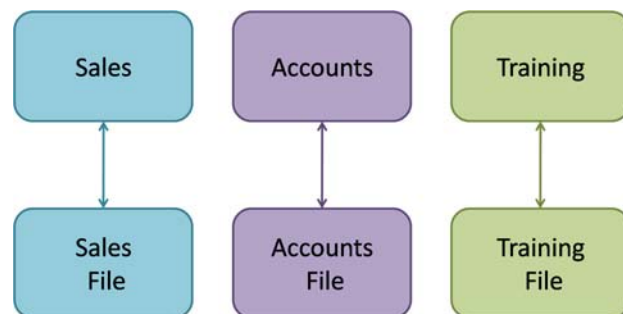
The DBMS

The OCR Specification says that you should be able to:

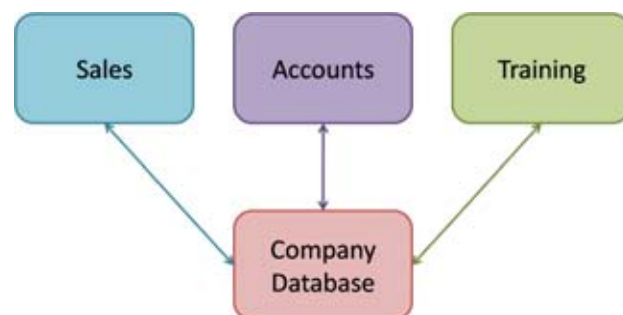
- describe how a DBMS allows the separation of data from applications and why this is desirable
- describe the principal features of a DBMS and how they can be used to create customised data handling applications

The traditional approach to storing data in a company was for everyone to store their own data in their own files.

A company might have a Sales department dealing with customers, an Accounts department that dealt with the same customers when they bought things and a Training department that trained the same customers on the products they'd just bought. Traditionally, all of these departments kept their own data files, rather than sharing one file of data.



The problem with this approach is that there are three copies of the same customer details. When a customer moves house he phones the sales person and assumes that the rest of company now has his new address. In practice, the Sales department had the up-to-date details but the rest of company didn't. This is called "data inconsistency" and is the main problem with duplicating data in more than one place. This is easily solved by everyone sharing one database instead of using their own files.

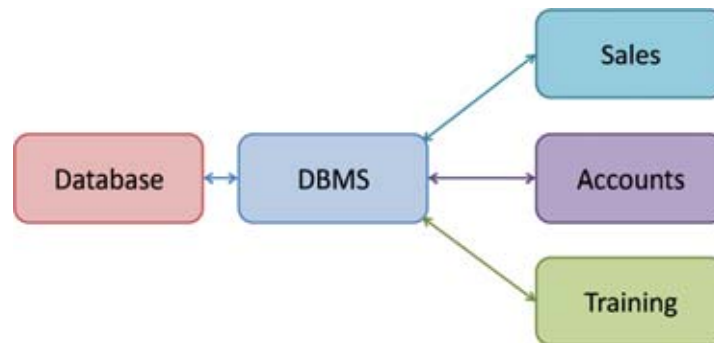


In computer systems we always try to avoid duplicating data in more than one place. **So remember:**

Data duplication leads to data inconsistency

However, having just one copy of the data led to a new problem. Each department wanted to do different things with the data. They all had their own departmental applications. Sharing the data meant that it was now consistent but introduced security issues and issues with the applications being closely linked to the data. If the Sales department changed something in the database then it affected the programs that other departments were using. The applications were too closely tied to the database. There needed to be some **separation** between the programs and the data they were using.

The separation of the programs from the data is achieved using a **database management system (DBMS)**. This is a software system that provides controlled access to the database. This separation is sometimes called **program-data independence**.



When a user logs in to the DBMS with their user name and password, this will give them access rights which enable them to read the parts of the data that they need. They may or may not be able to change or update the data. For example, the Training Department may be able to look at and change an employee's training record, and look at their full name and telephone extension but not change it. They will not be able to see the employee's salary details. The Personnel manager will have different access rights which enables him or her to see and update salary details. This is called having different **views** of the data.

Some key features of a database management system are that it:

- Provides separation between the applications and the database
- Allows multiple applications to use a single database
- Manages multiple applications trying to edit the same record at the same time (usually makes it read-only for second application)
- Provides appropriate views of the database to different users
- Provides security in terms of views and access rights
- Enables the creation of the relational database structure
- Allows applications to query/interrogate the database
- Enables the creation of reports based on queries
- Allows the applications with appropriate access rights to edit and maintain the database
- Can provide automatic backups if required

The DBMS that you are most likely to use is Microsoft Access. It provides all of the features above to create a customised data handling application.

The next section of this chapter discusses the components of a relational database and the DBMS functions used to create the application that uses it.



For you to find out...

Microsoft Access is very popular in schools and small businesses but what other Database Management Systems are there? Find three other examples of DBMS.

Relational Databases

The OCR Specification says that you should be able to:

- understand the relationship between entities and tables
- understand the components of a relational database, such as tables, forms, queries, reports and modules
- understand the use of logical operators in framing database queries
- explain the use of key fields to connect tables and avoid data redundancy
- describe methods of validating data as it is input

As we saw earlier in the chapter a database is a persistent organised store of data. The example of product details in a spreadsheet is a type of database. If you store all your data in a single file or worksheet it is called a **flat file**. This might be fine for simple data storage but as companies need to store more and more data, problems start to occur.

The spreadsheet (flat file) below stores student details. Every student belongs to a tutor group. Notice that all of the details about Mrs Robson's tutor group are repeated for every student in her tutor group. This is duplicated data and, as we said earlier, can lead to data inconsistency later. If Mrs Robson's tutor group moves rooms, then several students' records would need to be updated. It is too easy to miss one and end up with inconsistent data.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Candidate								Tutor				
2	Number	Name	Surname	Address1	Address2	Town	County	Postcode	Group	TutorName	Type	Room	
2	0001	Fred	Smith	23 The Road		Basingstoke	Hampshire	RG22 3FG	SJR	Mrs Robson	6th Form	203	
3	0002	Sally	Jones	32 Pleasant Str		Basingstoke	Hampshire	RG23 4SD	SLH	Miss Hickson	Year 9	120	
4	0003	George	Dotts	345 Boundary		Reading	Berkshire	RG1 4CV	SLH	Miss Hickson	Year 10	121	
5	0004	Joyce	Handel	2 Pickle Avenu		Reading	Berkshire	RG2 8NM	SJR	Mrs Robson	6th Form	203	
6	0005	Mabel	Pickles	The Old Post C		Guildford	Surrey	GU8 5GT	SJR	Mrs Robson	6th Form	203	
7	0006	Harry	Simpson	45 Foundary R		Basingstoke	Hampshire	RG24 6YU	PJT	Mr Thompson	Year 11	123	
8	0007	Xander	Erikson	Flat 3	Box Row	Reading	Berkshire	RG3 2SD	SLH	Miss Hickson	Year 10	121	
9	0008	Arnold	Holland	45 Reading Ro		Guildford	Surrey	GU1 5TY	PJT	Mr Thompson	Year 11	123	
10	0009	Lester	Jakes	567 Tower Stre		Basingstoke	Hampshire	RG23 4WQ	JKS	Mr Patel	Year 10	011	
11	0010	Jasmin	Flood	3a The Cottage	Dover Street	Basingstoke	Hampshire	RG24 7BN	JKS	Mr Patel	Year 11	012	

In computing we aim to store each piece of data just once. Ideally we should separate the data stored about tutor groups and the data stored about students. We only need to know which tutor group each student is in. We can refer to the other table to find out more details of that tutor group.

	A	B	C	D	E	F	G	H	I	J
	Candidate								Tutor	
1	Number	Name	Surname	Address1	Address2	Town	County	Postcode	Group	
2	0001	Fred	Smith	23 The Road		Basingstoke	Hampshire	RG22 3FG	SJR	
3	0002	Sally	Jones	32 Pleasant Str		Basingstoke	Hampshire	RG23 4SD	SLH	
4	0003	George	Dotts	345 Boundary		Reading	Berkshire	RG1 4CV	SLH	
5	0004	Joyce	Handel	2 Pickle Avenu		Reading	Berkshire	RG2 8NM	SJR	
6	0005	Mabel	Pickles	The Old Post C		Guildford	Surrey	GU8 5GT	SJR	
7	0006	Harry	Simpson	45 Foundary R		Basingstoke	Hampshire	RG24 6YU	PJT	
8	0007	Xander	Erikson	Flat 3	Box Row	Reading	Berkshire	RG3 2SD	SLH	
9	0008	Arnold	Holland	45 Reading Ro		Guildford	Surrey	GU1 5TY	PJT	
10	0009	Lester	Jakes	567 Tower Stre		Basingstoke	Hampshire			
11	0010	Jasmin	Flood	3a The Cottage	Dover Street	Basingstoke	Hampshire			
12										

	K	L	M	N
	TutorGroup	TutorName	Type	Room
	SJR	Mrs Robson	6th Form	203
	SLH	Miss Hickson	Year 9	120
	PJT	Mr Thompson	Year 11	123
	JKS	Mr Patel	Year 10	011

Reducing data duplication (data redundancy) has several benefits:

- reduces the risk of data inconsistency
- makes maintaining the database much easier
- reduces the size of the database

To understand relational databases there is no substitute for making one yourself. You can use Microsoft Access to create your own relational database and application (forms, queries and reports). The following sections look at the basic theory.

Creating a Relational Database

A relational database is a collection of data stored in related tables. In the real world there are people, objects and events (such as shows, holiday bookings or flights) that we store data about. These are called **entities**. In the spreadsheet example on the previous page we modelled two entities, *Student* and *Tutor Group*.

In a relational database we create a table for each entity, so a table is a collection of data about a specific entity. We use a database management system such as Microsoft Access to create these tables and to create logical links or **relationships** between them.

Here is the data from the spreadsheet example described earlier in the chapter, reorganised into tables in Access. There are two entities, *Student* and *Tutor Group*, so we create a table for each. This avoids storing the same data twice (i.e. avoids data redundancy). Notice that the tutor group initials appear against each student so we know which tutor group they are in. This same field also appears in the tblTutorGroup table so we can then reference the tutor group details. These fields have a special function in the database; they are used to create a **relationship** between the two tables.

Candic	First Na	Surnan	Address1	Address2	Town	Coun	Postcod	TutorGroup
0001	Fred	Smith	23 The Road		Basingstol	Hampsh	RG22 3FG	SJR
0002	Sally	Jones	32 Pleasant Str		Basingstol	Hampsh	RG23 4SD	JLH
0003	George	Dotts	345 Boundary I		Reading	Berkshir	RG1 4CV	JLH
0004	Joyce	Handel	2 Pickle Avenu		Reading	Berkshir	RG2 8NM	SJR
0005	Mabel	Pickles	The Old Post C	Church Lane	Guildford	Surrey	GU8 5GT	SJR
0006	Harry	Simpson	45 Foundary R		Basingstol	Hampsh	RG24 6YU	PLT

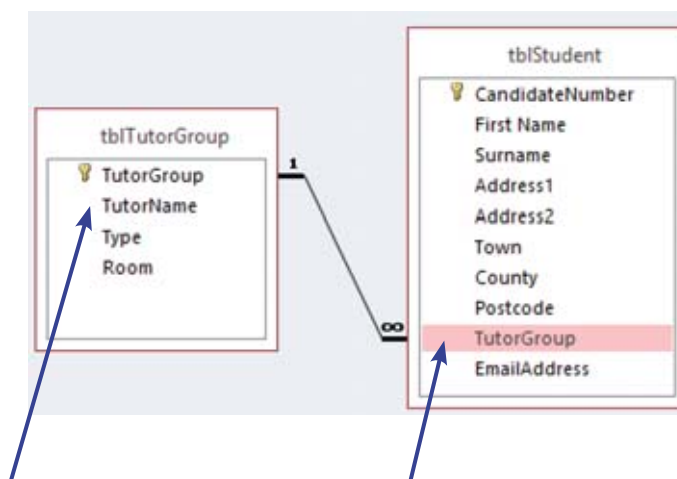
TutorGroup	TutorName	Type	Room
SJR	Susan Robson	6th Form	27
PLT	Paula Timms	Year 11	123
JLH	Jane Hobson	6th Form	120
JKS	Jonathan Simpson	Year 10	11

Both Sally and George are in tutor group "JLH". This field connects tblStudent to tblTutorGroup so the tutor group details can be accessed. From the two related tables we can see that Sally and George both go to classroom 120 for tutor time.

Each table contains a record for each student or each tutor group. The records are made up of fields, which are individual data items such as "surname" or "town". With potentially hundreds or thousands of records it is critical that the computer can tell which one is which so every record must have a unique identifier. The field that uniquely identifies each record in a table is called the **primary key**.

The primary key can be a number or text but every record must have a different value for this field. In the example above the primary key for the Student table tblStudent is "CandidateNumber". The primary key for the TutorGroup table is "TutorGroup" (the tutor's initials in this case).

In Access, the relationship between the two tables appears like this:



*TutorGroup is the primary key in
tblTutorGroup ...*

...and the foreign key in tblStudent

It shows the two tables and the fields they contain. It shows the primary key fields in each table with the key symbol. It also shows the relationship created between the two tables. In this case, it is a one-to-many

relationship: one tutor group has many students. Be aware that you can only create a relationship from the primary key in one table to the same field in the related table. In the related table, the field is called the **foreign key**.

Using Forms to Add/Edit Data

Having created the database itself (the collection of related tables) the data must be entered. This can be imported from another file such as a spreadsheet or typed into the table, one record at a time. Once the database is in use the user will need to be able to add new records, delete records and edit existing records.

In Microsoft Access the user can be given direct access to the tables but normally they would have a forms-based application interface to protect the database from user errors.

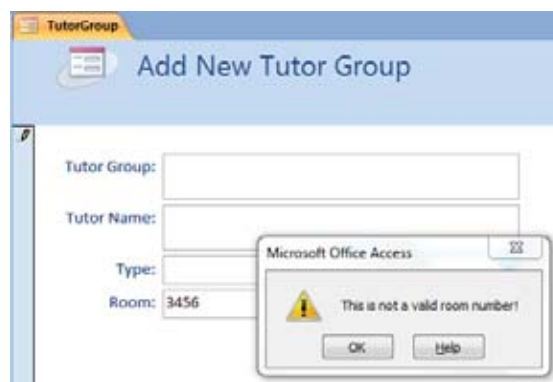
Forms can easily be created with a DBMS to allow controlled access to the underlying database. Remember that there may be several different applications using the same data and that each type of user will need different types of access. Some people will be allowed to see and edit all of the data but others may only be able to view it or access reports based on it. One of the strengths of a DBMS is that it can offer different types of user different views.

The form below is a typical form made in Microsoft Access to add new Tutor Groups to the database.



The screenshot shows a Microsoft Access form titled "Add New Tutor Group". The form has a light blue header with the title. Below the header, there are four input fields: "Tutor Group:", "Tutor Name:", "Type:" (with a dropdown arrow), and "Room:". The "Room:" field is currently empty.

Having created a form that allows users to easily input data, we can also add **validation checks** to ensure that the data being entered is reasonable. In this case we know that the school numbers its rooms with 3 digits so we can check that the room number is between 1 and 999 (a range check). If the user enters something else then they get an error message. This protects the integrity of the data.



The screenshot shows the same Microsoft Access form as above, but with an error message displayed. The "Room:" field now contains the value "3456". An error dialog box is overlaid on the form, titled "Microsoft Office Access", with a yellow warning icon and the message "This is not a valid room number!". The dialog box has "OK" and "Help" buttons.

The screenshot below shows another type of validation where the user is forced to pick options from a list so that they cannot enter any other variations by mistake:



The screenshot shows a web application window titled 'TutorGroup' with a sub-header 'Add New Tutor Group'. The form contains four fields: 'Tutor Group:' (text input), 'Tutor Name:' (text input), 'Type:' (dropdown menu), and 'Room:' (list box). The 'Room:' list box is open, showing four options: 'Year 9', 'Year 10', 'Year 11', and '6th Form'.

Validation checks are carried out by the computer software to check that the data entered is reasonable or sensible and conforms to rules specified by the programmer about the type of data allowed. There are several types of validation that are used when creating an application:

- **Range check:** a number or date is within a sensible/allowed range
- **Type check:** data is the right type such as an integer or a letter or text
- **Length check:** text entered is not too long or too short – for example, a password is greater than 8 characters, a product description is no longer than 25 characters
- **Existence check:** checks that a product or customer exists in the database – for example, if an order for a product is being entered, it checks that the product actually exists in the database
- **Presence check:** checks that some data has been entered, i.e. that the field has not been left blank
- **Format check:** checks that the format of, for example, a postcode or email address is appropriate

Validation can only check if a data item is reasonable. It cannot tell if it is correct. This is an important differentiation. If the "Student's Details" form prompts for Date of Birth the application can check that the data entered would be appropriate for that age group, but it cannot tell if you entered November instead of December by mistake.

Whilst validation ensures that the data entered are sensible, **verification** double-checks that it has been typed in correctly. Data is entered twice and the two versions are compared. If they are different the user can be prompted to try again. This is commonly used where email addresses and passwords are entered on forms.

Using Queries to Interrogate the Database

Stored data is only as useful as what you do with it. Once data has been stored in a database the users need to be able to find out things about it and generate reports.

Queries are used to interrogate the database. With a student database you may need to find a specific student's contact details or get a list of all the students in a particular class, for example. A DBMS will allow you to construct queries that find data matching specified criteria.

In Microsoft Access the Query By Example (QBE) grid makes it easy to find records that match criteria.

Here is a query to find all students who live in Hampshire:

The screenshot shows the Query Design View for a query named 'qryStudentsInHampshire'. The design grid is as follows:

Field:	CandidateNumber	First Name	Surname	Address1	Address2	Town	County	Postcode
Table:	tblStudent	tblStudent	tblStudent	tblStudent	tblStudent	tblStudent	tblStudent	tblStudent
Sort:								
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:							"Hampshire"	
or:								

A blue arrow points from the text "Records where County = 'Hampshire' are selected" to the 'Hampshire' entry in the Criteria row under the County field.

The resulting list:

Candir	First Name	Surname	Address1	Address2	Town	County	Postcod
0001	Fred	Smith	23 The Road		Basingstoke	Hampshire	RG22 3FG
0002	Sally	Jones	32 Pleasant Street		Basingstoke	Hampshire	RG23 4SD
0006	Harry	Simpson	45 Foundary Road		Basingstoke	Hampshire	RG24 6YU
0009	Lester	Jakes	567 Tower Street		Basingstoke	Hampshire	RG23 4WQ
0010	Jasmin	Flood	3a The Cottages	Dover Street	Basingstoke	Hampshire	RG24 7BN

In an exam you will need to be able to write criteria to select specific records. Some of these will include more than one part and some logical operators (NOT, AND, OR). Here are some examples:

`CandidateNumber="0010"` selects the details for that specific candidate

`(County="Hampshire") AND (TutorGroup="SJR")` selects students who live in Hampshire and are in tutor group SJR

Notice that each part of the criteria is in brackets where there is more than one part. Just like BIDMAS in mathematical calculations, there is also an order of precedence for logical operators. AND takes precedence over OR so you may need extra brackets if the OR parts need resolving first:

`(County="Hampshire") AND (TutorGroup="SJR") OR (TutorGroup="JLH")`

The criteria above will return students from tutor group SJR who also live in Hampshire and all the students in tutor group JLH.

`(County="Hampshire") AND ((TutorGroup="SJR") OR (TutorGroup="JLH"))`

This version, with the OR parts bracketed, will return students who are in either of these tutor groups and who also live in Hampshire.

Using Reports to Present Information

Data are just facts and figures with no context, but they can be processed into reports to convey useful information. A DBMS can take data from tables or the results of queries and present them in formatted reports. Here is an example of a report produced in Access:

rptStudentList						
Student Contact Details						13 August 2014
Surname	First Name	Address1	Address2	Town	County	Postcode
Dotts	George	345 Boundary Lane		Reading	Berkshire	RG1 4CV
Erikson	Xander	Flat 3	Box Row	Reading	Berkshire	RG3 2SD
Flood	Jasmin	3a The Cottages	Dover Street	Basingstoke	Hampshire	RG24 7BN
Handel	Joyce	2 Pickle Avenue		Reading	Berkshire	RG2 8NM
Holland	Arnold	45 Reading Road		Guildford	Surrey	GU1 5TY
Jakes	Lester	567 Tower Street		Basingstoke	Hampshire	RG23 4WQ
Jones	Sally	32 Pleasant Street		Basingstoke	Hampshire	RG23 4SD
Pickles	Mabel	The Old Post Office	Church Lane	Guildford	Surrey	GU8 5GT
Simpson	Harry	45 Foundary Road		Basingstoke	Hampshire	RG24 6YU
Smith	Fred	23 The Road		Basingstoke	Hampshire	RG22 3FG

Page 1 of 1

A good report should:

- be dated
- have a clear title; company reports will generally be done in a “house style” and include the company logo
- be laid out clearly with column headings
- have page numbers
- be sorted into a logical or useful sequence. For example, lists of people tend to be sorted alphabetically by surname, and a price list might be sorted by product ID number.

Sometimes data is grouped within a report and there may be subtotals and totals of numerical amounts.

Customising Your Application Using Code

However good the features of a DBMS are, there is always something you want to tailor to the specific requirements of the application. Most database management systems provide a programming interface so you can write code. The code is stored in **modules**.

Here is an example of a validation routine written in Visual Basic for Applications (VBA), the programming language within Microsoft Access. It checks that email addresses have an @ symbol and a dot in them:

```
Private Sub txtEmailAddress_Exit(Cancel As Integer)
    Dim AtPosn As Integer
    Dim DotPosn As Integer

    If Len(txtEmailAddress.Text) < 5 Then
        MsgBox "Email address is too short!"
    Else
        AtPosn = InStr(txtEmailAddress.Text, "@")
        If AtPosn < 2 Then
            MsgBox "Invalid email address, need characters before @"
        Else
            DotPosn = InStr(AtPosn, txtEmailAddress.Text, ".")
            If DotPosn = 0 Then
                MsgBox "Invalid email address, need at least 1 dot after @"
            End If
        End If
    End If
End Sub
```

This code tailors the form to validate the email address and generates appropriate error messages as follows:

The screenshot shows a form titled "Add Student Email Addresses" with the following fields:

- CandidateNumber: 0001
- First Name: Fred
- Surname: Smith
- EmailAddress: s@d

A Microsoft Access dialog box is displayed with the message "Email address is too short!" and an OK button.

The screenshot shows a form titled "Add Student Email Addresses" with the following fields:

- CandidateNumber: 0002
- First Name: Sally
- Surname: Jones
- EmailAddress: @sss.sss

A Microsoft Access dialog box is displayed with the message "Invalid email address, need characters before @" and an OK button.

The screenshot shows a form titled "Add Student Email Addresses" with the following fields:

- CandidateNumber: 0004
- First Name: Joyce
- Surname: Handel
- EmailAddress: ssss@dddd

A Microsoft Access dialog box is displayed with the message "Invalid email address, need at least 1 dot after @" and an OK button.

Glossary of Terms

Database Concepts

Database	A persistent organised store of data.
Persistent Storage	Non-volatile storage on a secondary storage medium such as a hard disk.
Data Duplication / Data Redundancy	Where the same data is stored more than once, unnecessarily.
Data Inconsistency	Where different versions of the same data have different values because duplicate versions have been stored and updated differently.
Program-Data Independence	Where the applications that use a shared database are separated from the actual data by a database management system. Changes can be made to one application without it affecting another.
DBMS	Stands for Database Management System, the system that separates the applications from the data and provides features that allow database systems to be created, interrogated and maintained.
Views	A feature of a DBMS that provides each application or user with specific access rights and views of the database.

Relational Databases

Flat File Database	A persistent organised store of data where data is stored in a single file organised into fields and records.
Relational Database	A persistent organised store of data where data is stored as a collection of related tables to minimise data redundancy.
Entity	A category of, for example, person (e.g. student, customer), object (e.g. classroom, stock item) or event (e.g. holiday booking, TV program) about which data is stored in a database, and which corresponds to a table in the relational database.
Table	A collection of data organised into records and fields within a relational database. A table represents a real world entity.
Record	Data stored about one instance of an entity: for example, one particular person or object
Field	One specific data item being stored such as surname or date.
Primary Key	A field in a table that uniquely identifies a record.
Foreign Key	A field in one table that is the primary key in another table and is used to create a relationship between those two tables.

Relationship	The logical connection created between two tables using a primary and foreign key pair. It allows related data about a record to be accessed from another table. A relationship between two tables can be one-to-one, one-to-many or many-to-many.
Form	Input. An interactive window used for data entry that usually includes validation routines, and uses controls such as combo boxes and radio buttons. Data input is saved to the database.
Validation	A check made by the computer to make sure the data is sensible.
Verification	A check to ensure that data has been input correctly. Sometimes this is done by prompting the user to read the data they have input and confirm it is correct. Other times the data has to be entered twice and one version is compared against the other to make sure it is the same.
Query	A feature of a DBMS that allows the database to be interrogated. It selects records from the database based on specified criteria.
Logical Operator	NOT, AND, OR. Used in complex criteria in queries.
Report	Output. A snapshot in time of the data from a database that can be printed. Data is formatted on a page and may be sorted or grouped. It may include totals.
Module	Section of code within a DBMS that allows the user interface to be tailored.

Past Exam Questions

June 2011, Question 2

1 A grocery shop uses a database with a DBMS to keep records of its stock.

(a) Explain what is meant by a DBMS. [3]

(b) The database uses forms and reports.

Describe each of these and give one example of how it would be used in the shop's database. [6]

Here is some data from the supermarket's database.

ProductID	Description	Supplier	Quantity Left	Reorder Level	Discontinued	Price
0001	6 eggs	Hill Farm	50	20	FALSE	£0.98
0002	2 litres of milk	Hill Farm	17	20	TRUE	£1.20
0003	1kg apples	Killey's	42	50	FALSE	£0.79
0004	250g butter	Hill Farm	12	25	FALSE	£0.49
0005	500g Moku Flakes	Moku Foods	0	10	TRUE	£0.99
0006	6 salad tomatoes	Killey's	30	30	FALSE	£0.89
0007	580g can baked beans	Moku Foods	27	30	FALSE	£0.42
0008	Family tomato ketchup	Moku Foods	41	0	FALSE	£1.45

(c) The shop runs queries using logical operators to select data for different purposes.

(i) State the ProductID of the products in the above sample which fit the following criteria.

- Supplier = Killey's
- Price > £1.00 OR Supplier = Hill Farm

[4]

(ii) Write the criteria which can be used to select all products which are not discontinued and where the QuantityLeft is lower than the ReorderLevel.

[3]



June 2012, Question 9

- 2 A DBMS is used to create customised data handling applications.
- (a) State what the initials DBMS stand for. [1]
- (b) Describe the features of a DBMS that can be used to create customised data handling applications and explain why using a DBMS is desirable.
- The quality of communication will be assessed in your answer to this question.

Jan 2013, Question 11

- 3 A social networking site uses a database to store the details of the people who have joined the site.
- (a) Describe what is meant by a database. [2]
- (b) When a person joins the website, they need to enter some personal data which is validated using rules. For example, the date of birth must be in the past.
- State one rule that could be used when validating each of the following:
- Email address
 - Gender
 - Password [3]
- (c) Each user can upload several pictures. Each picture has a date and a comment.
- The personal data of users is stored in a table called USER. The data about the pictures is stored in a separate table called PICTURE.
- Explain why the data about the pictures should be in a separate table, and how the tables can be linked. [4]

Chapter 6: Communications and Networking

This topic covers the basics of local and wide area networks. It looks at the Internet as an example of a wide area network and at some of the technologies associated with networks.

Local Area Networks (LANs)

A **Local Area Network** (LAN) is defined as a collection of computers and peripheral devices (such as printers) connected together on a single site. Notice that it is within a single site and not a single building. At school you probably have many different buildings within a campus. The school's LAN will connect together the computers in all these buildings.

The OCR Specification says that you should be able to:

- explain the advantages of networking stand-alone computers into a local area network

Consider how you work on a single laptop or PC at home that is not connected to a network. Compare this to how you work on computers at school, which are all on a LAN. A network in a large office building will provide all the same features that you have at school.

Benefits of Networking Computers

The benefits of networking the computers fall into these categories:

Sharing resources:

- Sharing folders and files so you can access files anywhere on the network from any computer and different people can access these files as needed
- Sharing peripheral devices such as printers and scanners
- Sharing an Internet connection

Communication:

- Using email to communicate with colleagues
- Using messaging systems to chat while you are working on other things
- Transferring files between computers

Centralised management:

- User profiles and security can all be managed centrally
- Software can be distributed across the network rather than having to install it on each individual computer
- Users can use any PC on the network but still see their own files
- Centralised backup of all files

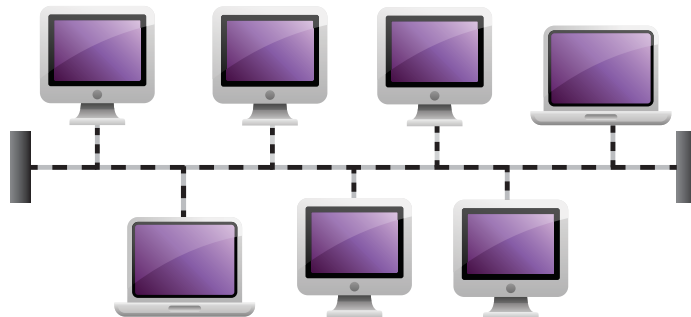
Network Topologies

The OCR Specification says that you should be able to:

- describe, using diagrams or otherwise, the ring, bus and star network topologies
- describe the hardware needed to connect stand-alone computers into a local area network, including hub/switches, wireless access points

Computers can be connected together in different layouts, or topologies. There are three basic topologies that are used but these may be combined in a large network.

Bus network

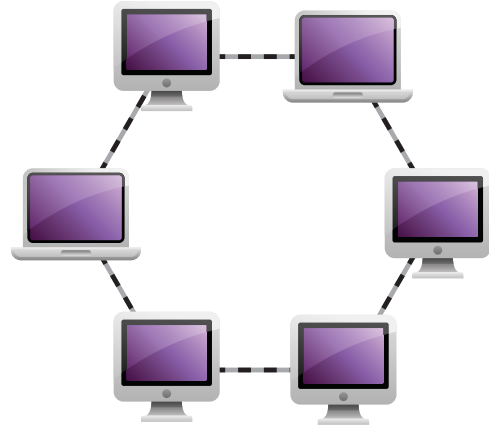


Computers are connected to a single backbone cable. The computers all share this cable to transmit to each other but only one computer can transmit at any one time. This is fine most of the time if the network is not too busy but if there is a lot of traffic then transmissions interfere with each other and computers have to retransmit.

Advantages:	Disadvantages:
1 Easy and inexpensive to install – less cabling than in a star network	1 If the main cable fails then the whole network goes down
2 Easy to add new computers	2 Cable failures are hard to isolate because all of the computers in the network are affected
	3 Performance slows down as the amount of traffic increases

Ring network

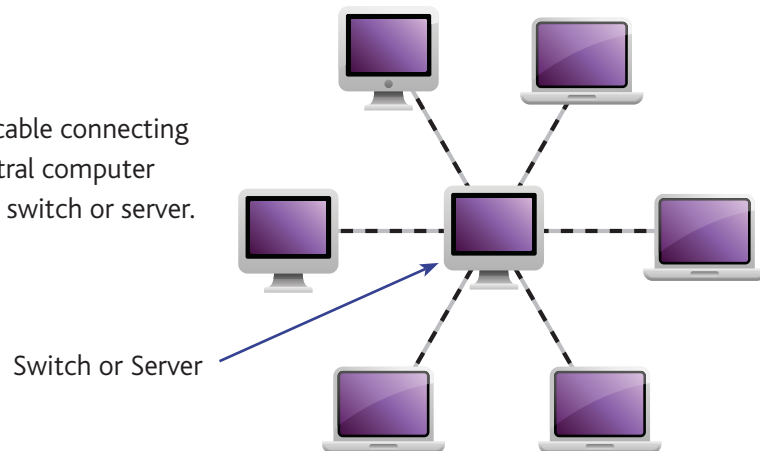
Computers are connected to adjacent computers in a ring. Computers take it in turns to transmit, controlled by passing a token around the ring. Computers can only transmit when they have the token.



Advantages:	Disadvantages:
1 Not dependent on a central computer like the star network	1 A single node or link failure disrupts the entire network
2 Token passing protocol is simple and therefore reliable	
3 Consistent performance even when there is a lot of traffic	

Star network

All of the computers have their own cable connecting them to a central computer. The central computer controls the network. This is usually a switch or server.



Advantages:	Disadvantages:
1 If one cable fails only one station is affected rather than the whole network	1 Can be costly to install because there is a lot of cabling
2 Consistent performance even when the network is heavily used	2 If the central computer fails then the whole network goes down
3 Easy to add new computers	
4 More secure – messages from a computer go directly to the centre	

Connecting Computers to the LAN

Each computer must have a network card so it can be connected to the LAN. The correct term for the network card is **Network Interface Card (NIC)**. The right type must be fitted for the kind of network connection required, either by a cable or wirelessly to the network.

In most classrooms or offices, computers tend to be star-wired to a hub or switch, which is either in the room or nearby. The room layout may look like a bus network because the cables all go around the edge of the room but in actual fact, each computer has its own wire running through the ducting to the central device.

Smaller star or bus networks in rooms/offices will then be connected to a backbone that could also be a star or bus topology. In a modern network there will be a mixture of topologies.

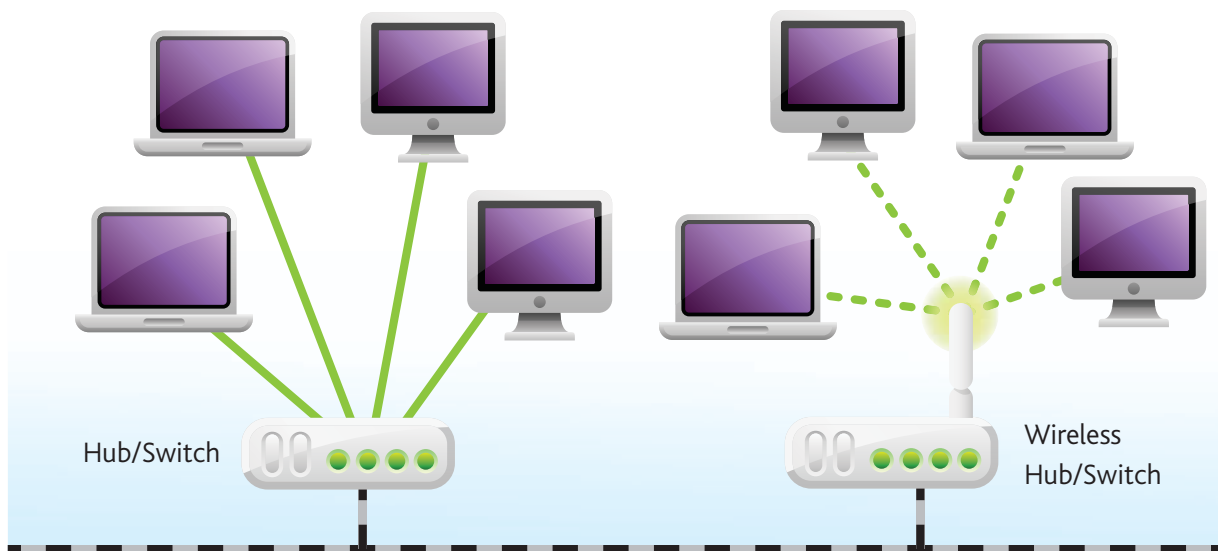
The central device in a small star network may be a server. In an office or classroom on a bigger LAN, the central device in a star will be a switch.

Star topology used to connect computers in an office or classroom

cabled in a star

or

a wireless star



Bus topology used to connect classrooms

Peer-to-Peer or Client-Server

The OCR Specification says that you should be able to:

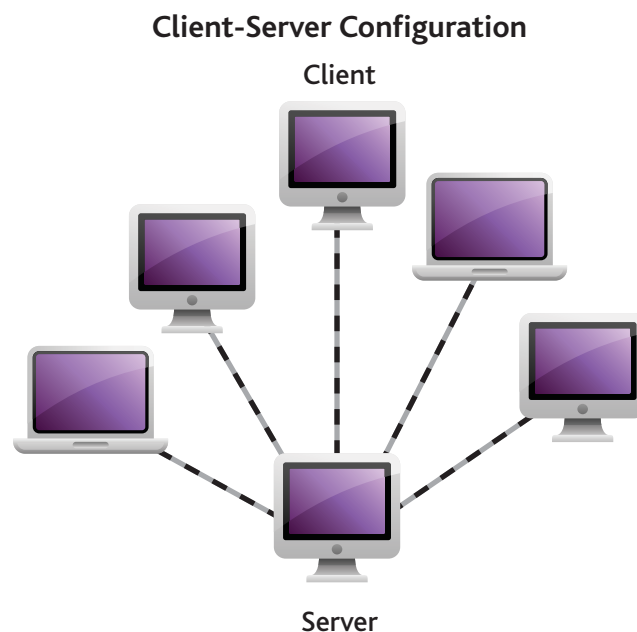
- explain the different roles of computers in a client-server and a peer-to-peer network

In your school you will be able to use your files on computers in different buildings. This is because they are not stored on the computer that you are using but are on a file server somewhere else in school. The **server** is a specialised computer with a different role from the normal PC. There will probably be a web server to host the school's external website and an email server which receives all emails and distributes them to network users. It may detect and block incoming emails that it thinks are spam.

In a large network it is common to have shared files and resources on centralised servers. The computers that you use around school are referred to as clients.

In a client-server network:

- the network is centrally managed by a powerful computer called the server
- client computers communicate via a central server
- there may be other computers which act as email server, print server and web server

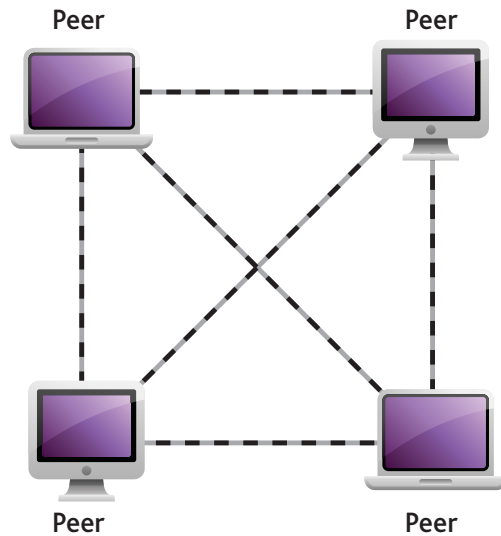


In smaller offices this is not practical. If you run a small company with a handful of computers it is not cost-effective to have a specialised server or someone to run the network. In a small office the computers will simply be cabled together. Each computer is configured so it will share specified files and folders with other peer computers on the LAN. PCs on the LAN can only access files on another computer if access rights have been granted. This is called a peer-to-peer network because all of the computers have equal status and the same role in the network.

In a peer-to-peer network:

- peer computers communicate directly with each other
- files are stored on individual computers but can be shared with others

Peer-to-Peer Configuration



Peer-to-Peer	Client-Server
All computers have equal status	Specialised roles: computers tend to be a client or a server
Easy to set up and maintain	Needs a network manager to run the network
No centralised management	Centralised security and management
Each computer must be backed up separately	Backup done from central server
No dependency on a server	Dependent on central server

Wide Area Networks (WAN)

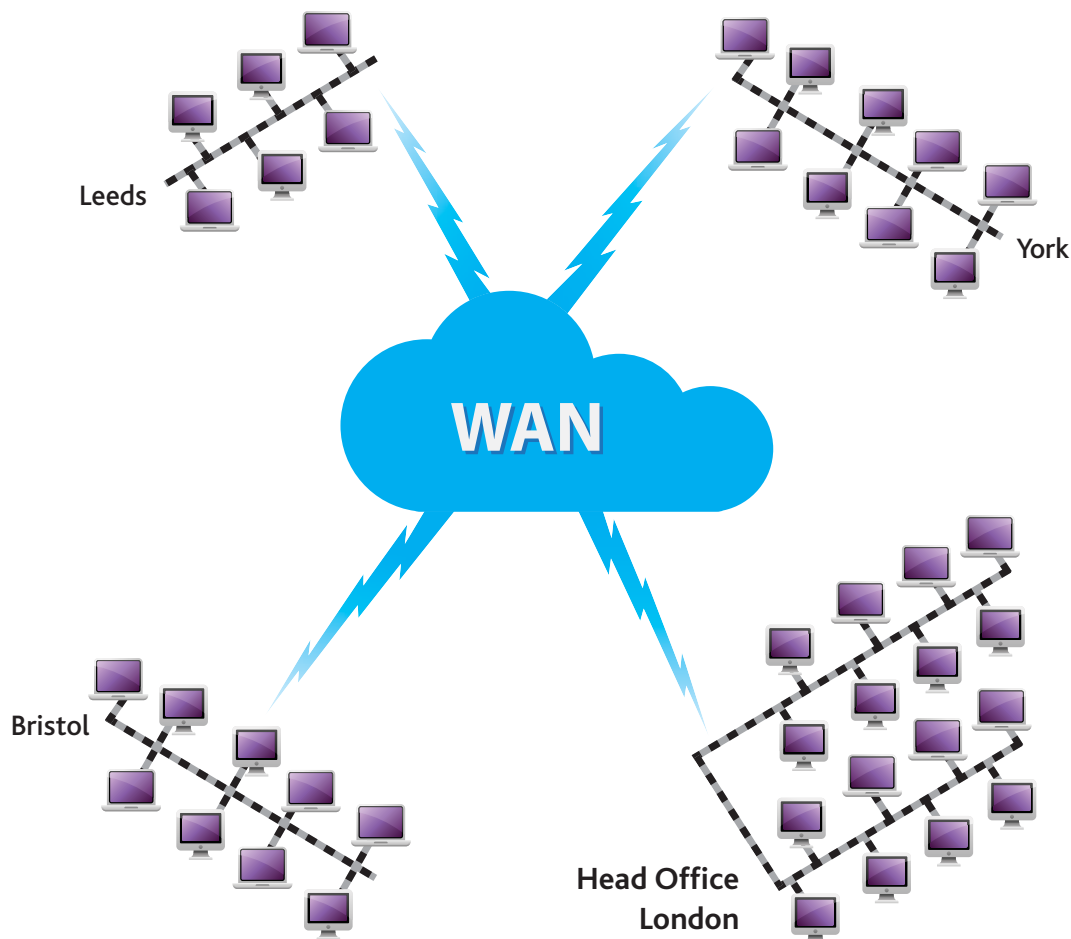
The OCR Specification says that you should be able to:

- describe the differences between a local area network and a wide area network such as the internet

We said in the last section that a LAN was a collection of computers connected together within a single site. A Wide Area Network, or WAN, is a collection of computers and networks connected together over a geographically remote area.

The term “geographically remote” is a confusing one. It does not mean that networks have to be miles apart, although they may be. Geographical remoteness is more about what separates sites than the distance involved. Your campus may be quite large but is still a LAN. If you have a split campus with a public highway or some other buildings in between the two campuses, then these will need to be connected by a WAN. The Internet is a WAN and this is worldwide.

WANs use hired infrastructure to connect the LANs together; the school or business cannot install its own cables between the two sites. A business with offices in London, Leeds, Bristol and York will lease connections from a network service provider to connect the four office LANs together.



How Devices Communicate on a Network

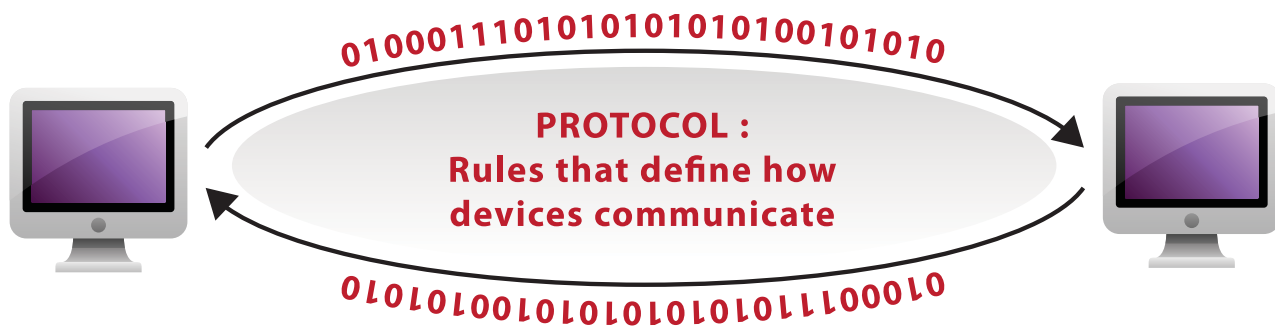
The OCR Specification says that you should be able to:

- explain the terms IP addressing, MAC addressing, packet and protocols

Consider how you communicate as a human being. *Buna ziua, ce mai faci azi?* Sorry, don't you speak Romanian? When we go abroad we find it hard to communicate because we do not understand the language. Even if you can speak French you probably struggle in France because they talk too fast for you to understand. Computer communication has similar problems. Computers need to be speaking the same language and at the same speed in order to communicate.

If one computer transmits a stream of binary to another computer, the receiving end needs to know what the rules are. This is called a **protocol**. A protocol is the set of rules that define how devices communicate. A protocol will cover:

- how the communication will start, getting the attention of the other computer ("Oi, you!")
- the transmission speed
- the significance of the bits being transmitted (like the language)
- how the bits will be delivered (one at a time or in groups of 8 for example)
- error-checking procedures being used (this involves some extra bits, like punctuation).



Internet Protocol (IP)

One protocol that you may have heard of is IP. This is the protocol used on the Internet. All computers and servers connected to the Internet, as well as the routers that make up the Internet, communicate using IP.

Internet Protocol will be discussed in more detail in the following section about the Internet.

MAC Addresses

If devices are going to communicate they must have a unique reference number. This is called an address. It is the same principal as addressing a letter; you need to put a unique address on the front so the postman knows where to deliver it. Within a LAN each device must have a Network Interface Card (NIC) to connect it to the network. This card will have a **MAC address**. The MAC address is hard-coded into the NIC when it is manufactured; it cannot be configured using software. Every networked device will have a MAC address. It is a 48-bit address that is written as twelve hex digits to make it easier for humans to work with. For example:

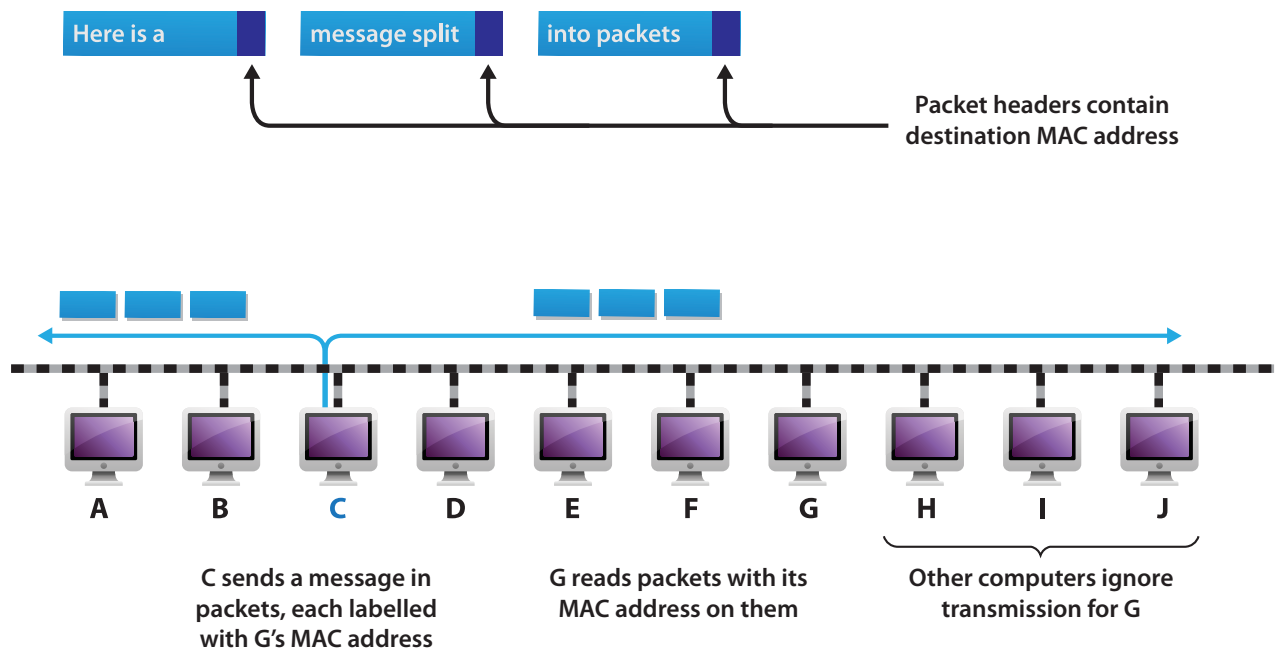
In hex: 00-09-7C-F1-F7-85

In binary: 000000000000100101111100111100011111011110000101

The MAC address is used to transmit between devices within a LAN.

Packets on a LAN

When two devices want to communicate across a LAN one device will send a **message** to the other. This message is broken into smaller chunks called **packets**. These packets are broadcast onto the LAN with the MAC address of the destination device. The destination device will see all of the traffic on the LAN but will only pick up the packets with its own MAC address on them. It's a bit like getting your luggage back after a flight – you watch all the bags go around on the conveyer belt but you only pick up the one with your name on the label.



When devices communicate over a WAN, such as the Internet, they also use packets. For example, when you request a webpage in your browser, that message is sent to the web server as a series of packets. This is discussed further in the following section on the Internet.

The Internet

The OCR Specification says that you should be able to:

- describe the nature of the Internet as a worldwide collection of computer networks
- explain the need for IP addressing of resources on the Internet and how this can be facilitated by the role of DNS servers
- explain the terms IP addressing...packet and protocols

The Internet is a wide area network. It is a worldwide collection of computers and networks that uses Internet Protocol (IP) to communicate. It isn't owned or managed by any one group of people and anyone can access it.

The network itself is made up of network devices called **routers**. These are much bigger, higher performance routers than the ones at school or at home. They form the backbone of the network. IP is the protocol used between these routers.



Internet Protocol (IP)

Each device on the Internet must have a unique IP address so that it can communicate over the wide area. This is in addition to the hardware MAC address, which is only used to address devices inside a LAN. The IP address is made up of four numbers separated by three dots, for example: 193.127.30.23

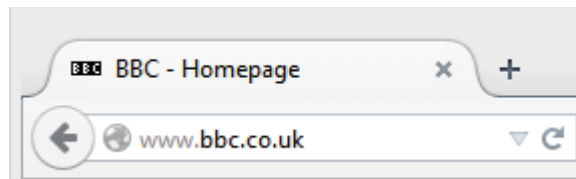
The four numbers each represent a byte so each one can only be a number between 0 and 255. The computers will be sending binary, not decimal integers, but we write the addresses as four separate numbers because it is easier for a person to deal with. (The dots are not transmitted; they are just separators to make it easier for us to read.)

```
11000001  01111111  00011110  00010111
    193      127      30       23
```

Some IP addresses have special significance and are not used for devices. For example, 0.0.0.0 and 255.255.255.255 are never used to address a device.

Accessing Websites on the Internet

Websites are stored on web servers connected to the Internet. The site will have an IP address so people can access the pages using their browser software. However, when you want to access a site you don't type the IP address, you type in a **domain name** such as www.bbc.co.uk. This is because humans are quite bad at remembering numbers and typing them in correctly so the domain name is a text reference to a site that can be translated into the numerical IP address.



When you type the domain name, "www.bbc.co.uk" into the browser the web page request is sent to a **Domain Name System (DNS) server** in the Internet. The DNS server has a database of domain names and IP addresses so it can translate the domain name into an IP address.

There is a large number of DNS servers in the Internet and these communicate with each other so the DNS servers regularly update each other. If your local DNS server does not have the domain name listed, the page request can be forwarded to another DNS server.



The advantages of using DNS servers to translate domain names into IP addresses are:

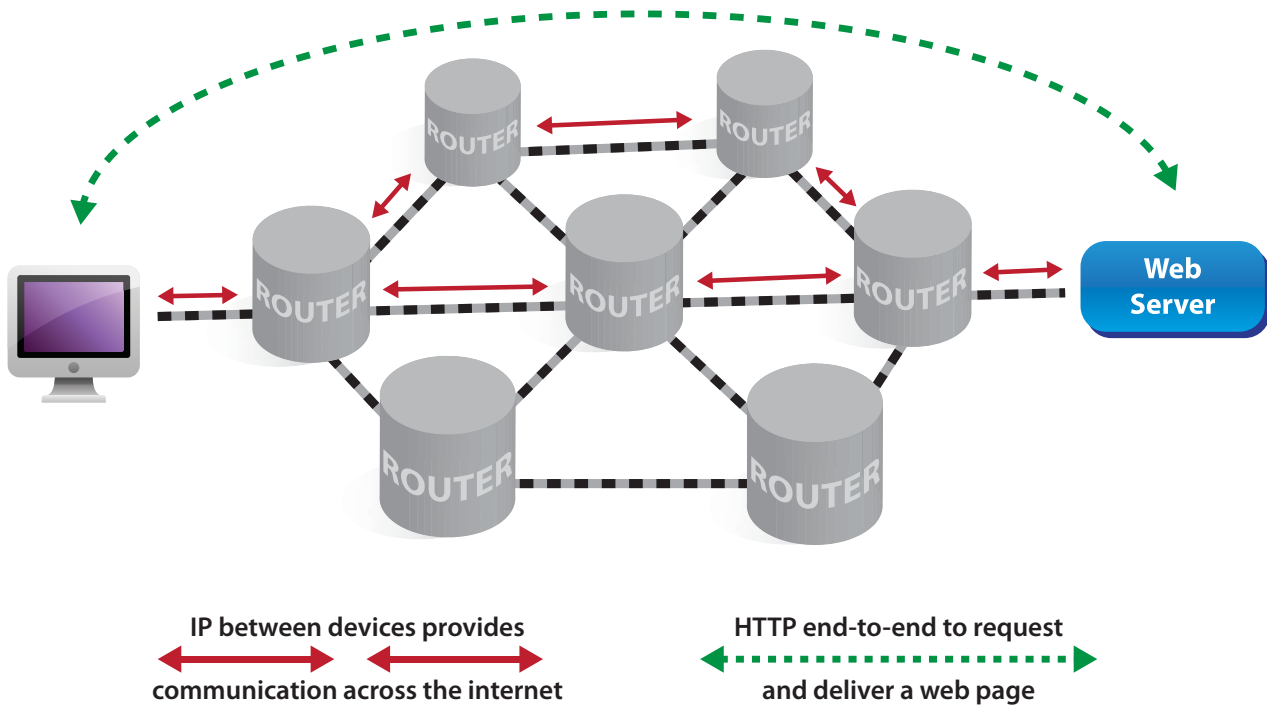
- Humans do not have to remember or type in numerical addresses
- If the IP addresses change at some point the DNS servers can update their databases, and the users can continue using the same domain names
- Many distributed DNS servers mean that everyone has access to all addresses from their local DNS server

In most browsers you can also type the IP address of a site into the address bar to get to a site. We don't tend to do this but it shows that essentially the domain name and IP address both address the same place. This IP address gets you to the BBC home page:



You may also have noticed that your browser adds "http://" in front of the domain name you typed. This is another example of a protocol. **HTTP** stands for **Hyper Text Transfer Protocol** and is the protocol used to request and deliver web pages.

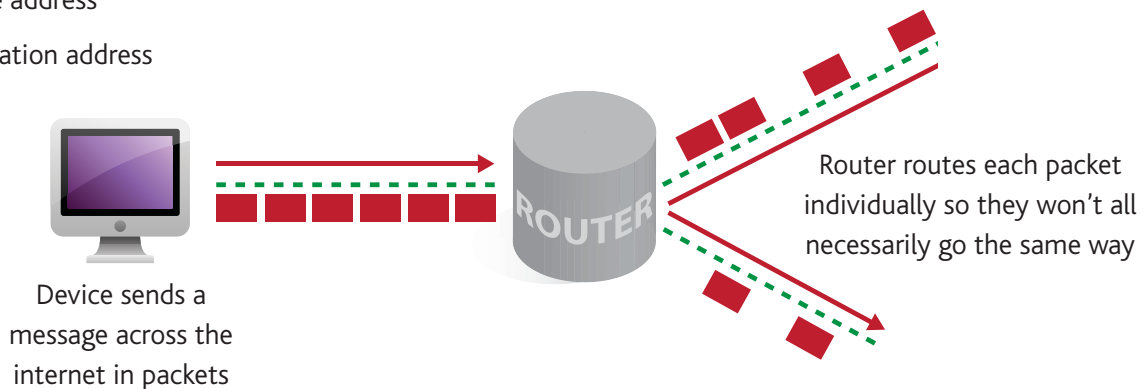
This protocol operates over IP. IP is the protocol used between routers and devices. HTTP is an end-to-end protocol between the PC and the web server. In computing, protocols often operate in layers. Imagine the process of sending a letter to a friend. The Post Office operates the lower level protocol between towns to physically deliver your letter (like IP between the routers and LANs). The letter is the higher level protocol that communicates end-to-end between you and your friend (like HTTP requesting a page from a web server and the server sending the page back).



Packet switching on the Internet

The messages being sent between devices on the Internet will be split up into smaller chunks called **packets**. The IP protocol is then responsible for delivering these packets from one device (the source) to another device (the **destination**), for example from a PC to a web server. At each router, the IP protocol decides which way to send each packet. Each router needs to know where the packet came from (source address) and where it's going to (destination address), so it can make the appropriate routing decision. Each packet could potentially go via a different route to get to its endpoint, depending on traffic conditions at the moment IP makes its routing decision. For this reason every packet must have a **sequence number** on it. The destination device can then put the packets back together in the correct order. Each packet being sent across the Internet must therefore have a packet header on it, which contains:

- a sequence number
- a source address
- a destination address



Connecting to the internet

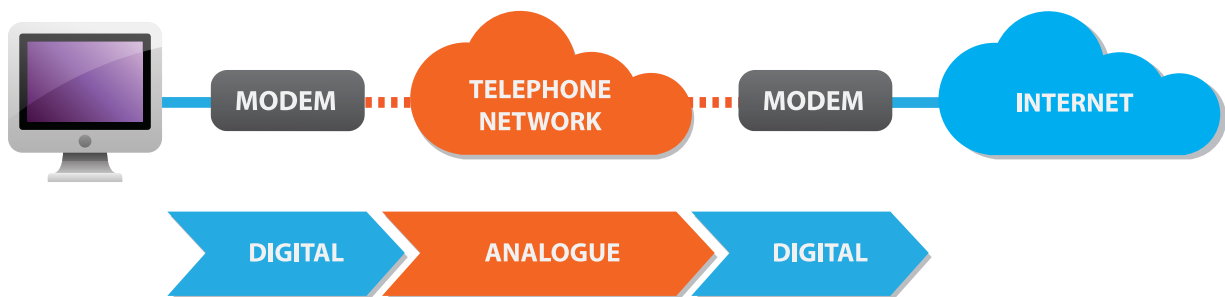
The OCR Specification says that you should be able to:

- describe the hardware needed to connect to the Internet including modems, routers, etc.

There are many ways to connect to the Internet:

Modem:

This is the cheapest but slowest type of connection, only 56kb/s (kilobits per second). It uses the telephone network to create a connection from your computer to the Internet. Your computer and the Internet are both **digital**, designed for computer traffic. The telephone network is **analogue**, designed for voice traffic. The **modem (modulator-demodulator)** is a device that converts the digital signal from the computer into an analogue signal, and vice versa at the other end. As well as being slow, this method means that you can only use the phone line for one transmission at a time; if you're on the Internet then the phone cannot be used at the same time.

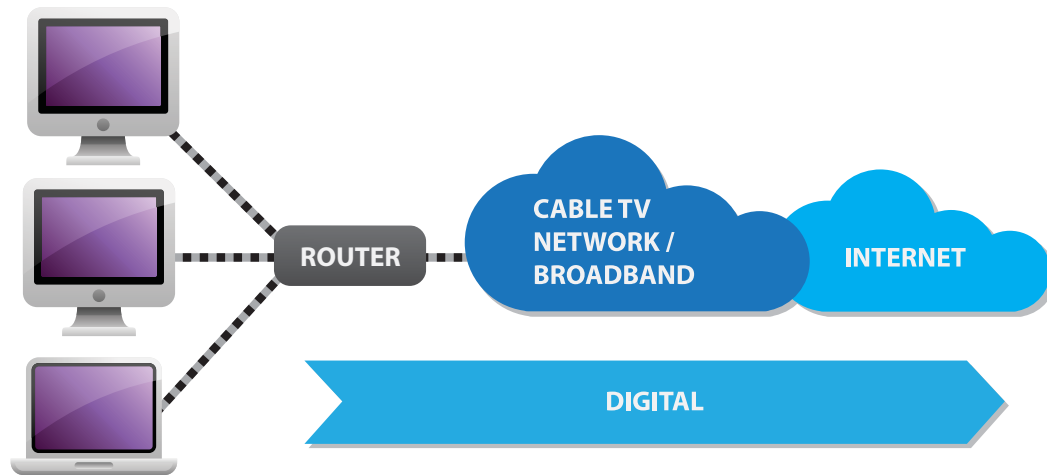


Local router:

Very few home PCs are connected to the Internet using a modem these days. Many of us will have more than one computer at home and will have a wireless router somewhere in the house. The computers all connect to the router to get an Internet connection and can also communicate with each other as they now form a small LAN.

The router can be connected to the Internet using:

- The fibre-optic cable that delivers your cable TV service
- The part of the phone line that connects your house to the local exchange using **broadband** technology. This is not the same as using a modem to transmit over the telephone network itself, but it does use the local cable. It is digital transmission all the way and allows the use of the computers and phone line at the same time, unlike a modem.



Broadband and cable TV services offer speeds in excess of 2Mb/s (megabits per second) and in places services up to 50Mb/s are being advertised (Aug 2014). The modem running at 56kb/s is just 0.05Mb/s (56/1024 to convert kilobits per second into megabits per second).

If your computer is connected to a LAN then there will be an Internet connection provided from a router connected to the LAN. The principal is the same as a home router but it will be more powerful.

Creating web pages in HTML

The OCR Specification says that you should be able to:

- explain the importance of HTML and its derivatives as a standard for the creation of web pages

Web pages are written in a programming language called **HTML**, HyperText Markup Language. HTML is used to describe the page content. It is used with a Cascading Style Sheet (**CSS**) which defines how the content is styled. Some styling can be coded with HTML but it is considered good practice to use CSS for styling and HTML for content.



See HTML and CSS in action...

Check out the Zen Garden website to see how the same HTML content can look with different Cascading Style Sheets.
www.csszengarden.com

```
<div class="summary" id="zen-summary" role="article">
  <p>A demonstration of what can be accomplished through <abbr title="Cascading Style Sheets">CSS</abbr>-based
  design. Select any style sheet from the list to load it into this page.</p>
  <p>Download the example <a href="/examples/index" title="This page's source HTML code, not to be modified.">html
  file</a> and <a href="/examples/style.css" title="This page's sample CSS, the file you may modify.">css file</a></p>
</div>

<div class="preamble" id="zen-preamble" role="article">
  <h3>The Road to Enlightenment</h3>
  <p>Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible <abbr
  title="Document Object Model">DOM</abbr>s, broken <abbr title="Cascading Style Sheets">CSS</abbr> support, and abandoned browsers.</p>
  <p>We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk
  like the <abbr title="World Wide Web Consortium">W3C</abbr>, <abbr title="Web Standards Project">WaSP</abbr>, and the major browser
  creators.</p>
  <p>The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with
  clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.</p>
</div>
</section>
```



A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example  HTML FILE and  CSS FILE

THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WASP, and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see

HTML is coded using **tags** that define the page structure. You can write HTML in a basic text editor and save the file as "filename.html". You can then run the page in your browser. Here is a very basic webpage in HTML:

```
<html>
<head>
  <title>Seahorses in the UK</title>
</head>

<body>
  <h1> Dorset Wildlife </h1>
  <h2> Studland Bay Seahorses </h2>
  <p> Studland Bay is a breeding site for both spiny and short-nosed
seahorses.</p>
  <h2> Habitat </h2>
  <p> The seabed in Studland Bay is mostly sandy with sea grasses
and rocky outcrops, providing protection for the seahorses and hard
surfaces for other inhabitants such as the native oyster to cling to.</
p>
  <p>More on <a href=http://www.theseahorsetrust.org/seahorse-facts.
aspx>
seahorses</a> here.
  <br>
  <br>
   </p>
</body>
</html>
```

Tag	What it does
<head> </head>	Start and end tags for information <i>about</i> this page, not text that appears <i>on</i> the page.
<title> </title>	The text between these tags appears in the blue bar at the top of the browser window.
<body> </body>	The HTML between the body tags defines the page content.
<p> </p>	Paragraph tags that start and end a paragraph.
<h1> </h1> <h2> </h2>	Heading tags will go either end of heading text. The CSS will define how these look but normally heading text is bigger than normal page text.
	The image source tag references a picture file and defines basic parameters such as size.
<a>	An anchor tag which defines a hyperlink using an additional href attribute to set the link's destination.

The web page it produces look like this:

Dorset Wildlife

Studland Bay Seahorses

Studland Bay is a breeding site for both spiny and short-nosed seahorses.

Habitat

The seabed in Studland Bay is mostly sandy with sea grasses and rocky outcrops, providing protection for the seahorses and hard surfaces for other inhabitants such as the native oyster to cling to.

More on [seahorses](#) here.



When you create a page using HTML it is not a website that other people can get to yet; you can only see the page in your browser because you have access to the file. A website must be published to the Internet so other people can access it. This can be done by uploading it to your own web server or you can upload it to someone else's web server. If you use someone else's server this is called "hosting". Internet Service Providers (ISPs) offer this as a service, sometimes free for small sites if you are already a customer.

Writing a website using HTML would be time-consuming so there are application packages available to help design a website and create the HTML code for you. One example of this is Dreamweaver.



Useful website...

An excellent site for details on HTML code is:

www.htmldog.com

Security

The OCR Specification says that you should be able to:

- explain the need for security measures in networks, such as user access levels, suitable passwords and encryption techniques
- describe and justify network policies such as acceptable use, disaster recovery, failover, backup, archiving

Security is about keeping your computer and the files stored on it safe from hazards. These hazards come in the form of viruses, hackers, hardware failures, software faults and natural disasters such as floods and fires, but the biggest hazard to networked computers with shared resources is other users!

Network security measures cover three important areas:

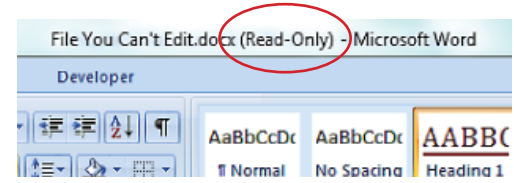
<p>1 Preventive measures</p>	<p>Aimed at stopping a hazardous event from occurring</p>	<ul style="list-style-type: none"> • Access rights • Firewalls • Physical security such as locking rooms • Passwords • Encryption • Acceptable use policy
<p>2 Detective measures</p>	<p>Aimed at detecting when data has been corrupted or systems have been compromised</p>	<ul style="list-style-type: none"> • Virus checking software • Firewall software • Fire alarms • Audit trails
<p>3 Corrective measures</p>	<p>Aimed at correcting or restoring the system after problems have occurred</p>	<ul style="list-style-type: none"> • Backup and restore procedures • Redundant hardware / Failover • Disaster recovery procedures

Chapter 3: Software covers security software for a standalone computer. Anti-virus software, firewall and spyware protection software are all equally applicable in a network.

Security Precautions

Access Rights

User access rights should be set for disks, folders and files so users can only access what they need to. At school you can probably read files on a shared area but not edit them; this is Read-Only access. The teacher will have Read-Write access to these folders. Some folders you won't even be able to see.



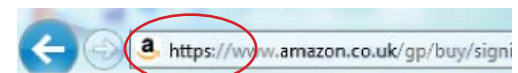
In a work environment, the Accounts staff will have access to payroll details but other departments will not. The Data Protection Act says that employers must keep personal data secure so setting appropriate access rights is a legal responsibility as well as a good idea.

Encryption

There are devices that can read network transmissions from the cables just by scanning the emissions; they don't even have to be plugged into the network. Also anything transmitted over a network can be intercepted and read. This takes place without leaving any trace so nobody would know that it had happened.

One way of stopping this unauthorised access to data is to encrypt anything sent on a network. Encryption changes the data before it is transmitted so it can only be deciphered by someone with the appropriate key. To anyone intercepting the message it would be unintelligible.

When you buy something on the Internet or use Internet banking you may have noticed that instead of HTTP in front of the domain name it changes to HTTPS. It works in the same way as HTTP but is encrypted so your payment details are kept secure.



Password Protection

In a networked environment such as a school or a company, many of the computers are used by more than one person. Even if employees have their own computer it may be in an open plan office. The easiest way to stop unauthorised access to your computer or your files is to use a combination of username and password.

The password should never be shared with friends or stuck on a post-it note under the keyboard (yes, people really do!). Also, the password should be "strong". This means that it is not easy to guess, it probably contains letters, numbers and symbols and is at least 6 characters long. Some companies make employees change their password every month but this doesn't really work because people usually just add the month number on the end because it is easy to remember.

User name:	<input type="text"/>
Password:	<input type="password"/>
<input type="button" value="Sign in"/>	

For additional security against people trying lots of different passwords to get into someone else's account, the account can be locked after a certain number of failed attempts.

Network Policies

As well as configuration and software precautions there are procedural precautions a company can take to protect its data. These procedures and policies include:

Backup and Restore Procedures

The staff that manage a network will back up the servers regularly. A backup is a copy of all the users' files, which can be restored in the event of files getting corrupted or deleted. Backup copies must be made regularly; how often will depend on the nature of the system to a certain extent. In some businesses a daily backup may be sufficient but in others, files may be backed up every hour. Backups are normally made using a removable hard disk or cassette tape. The medium has to be high capacity and portable so it can be stored in a fire-proof safe or off site.

Archiving

Often there is a large amount of data stored on a computer system that is no longer needed on a regular basis. However, you cannot delete it just in case it is needed again or because a company is legally required to keep some records for a number of years (for example tax returns). Archiving is when the data is taken off the main system and stored, usually on magnetic tape as it is cheap. It can be loaded back onto the system if it is needed again. It is not a copy like a backup. The point is to free up space on the main computer system.

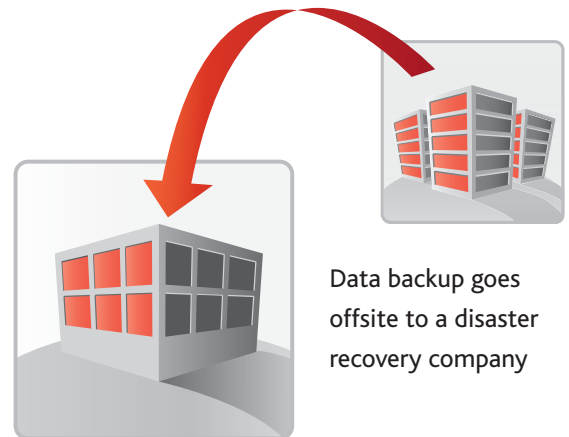
Disaster Recovery

Some companies would not be able to operate at all if their computer systems went down. These companies will have a **disaster recovery plan** that would enable them to keep on doing business even in the event of something catastrophic happening. Catastrophic events that destroy a whole building and all of the servers include things like fire, a tidal wave or a bomb.

A disaster recovery plan includes the ability to replicate the computer system in a very short time. This would involve:

- Data being backed up regularly
- Duplicate hardware systems being available very quickly
- The backup data being restored on the new hardware so the company could carry on as normal

Companies can either buy their own redundant hardware just in case but they are more likely to pay a disaster recovery company for the service. There are companies that specialise in providing hardware at short notice to companies.



Failover

When a computer system is mission-critical to a company it cannot be offline at all. Obviously hardware failures still happen but when they do the computer system will swap over to a spare component straight away. Spare components built into a computer system for this purpose are referred to as **redundant**.

A system that has lots of redundancy built into it is described as **fault-tolerant** because when faults happen the system copes with it. **Failover** is the process of swapping to the spare/redundant component. Failover happens automatically and transparently (without the user noticing).

Acceptable Use Policy

When you started at your school you may have had to sign an Acceptable Use Policy before you were given a username and password. This policy probably said you must not use other people's accounts, access pornography, play games or do anything else that is not related to your school work.

Employees will all sign a similar agreement. This is a contract between you and the school/company saying you agree to use the network only for certain things. In school you probably get away with playing games now and then but at work a company can sack you for going against the agreement you signed.

The Acceptable Use Policy makes it clear to all network users what is acceptable and what is not.



Hazard	Security Precaution
Accidental data deletion/corruption by users.	Backup and restore procedures
Unauthorised access to files/folders by employees	Passwords, firewall, access rights Acceptable use policy
Hardware failure or software faults	Backup and restore procedures Failover
Hackers	Passwords, firewall, access rights
Natural disasters: fire, flood, lightning strike etc	Backup and restore procedures Disaster recovery
Bomb	Backup and restore procedures Disaster recovery
Curious computing students "just seeing what happens when you click..."!!	Access rights Backup and restore Acceptable use policy (!)

File types and file compression

The OCR Specification says that you should be able to:

- explain the importance of compressing files that are transmitted via the Internet
- describe the differences between lossy and lossless compression
- describe common file standards associated with the Internet such as JPG, GIF, PDF, MP3, MPEG

When data is transmitted across the Internet it will go through many different physical links between routers. The connection from a computer or a LAN into the Internet is likely to be the slowest part of this route, as you probably know from experience. At home you may have quite a slow network connection and it may take a while for web pages to load.

One way of speeding up how quickly files can be transmitted across the Internet is to compress them to make them smaller. Smaller files take less time to transmit over a network.

Understanding how compression affects files is important as the type of compression selected will affect how the image looks or the audio track sounds. The final use of the file will dictate how much you can compress the files and still have a file that is useable.

Compression can be considered in two categories:

- **Lossy compression:** a data encoding method where files are compressed by removing some of the detail. For example, photographs can be stored using fewer colours so fewer bits are needed per pixel (see *Chapter 4: Data Representation*). Lossy compression is used to compress multimedia data such as pictures, audio files and video files.
- **Lossless compression:** a data encoding method where files are compressed but no data is lost. Essential for text and data files. For example, bank records must keep all of the data; you cannot transmit a bank statement and miss out a few zeros because they don't matter too much!

Still image file types

Still images are photographs and vector graphics (such as clipart). As we saw in *Chapter 4: Data Representation*, the way we store images will affect the size. The colour depth is one factor. If you use 8 bits to represent each pixel then you can use up to 256 different colours in the picture. "Truecolor" is the name given to a picture representation using 24 bits per pixel where the colours are a mix of Red, Green and Blue (RGB).

To compress an image you can use fewer bits per pixel. A bitmap image (.bmp) or portable network graphic (.png) file is a lossless version of the picture. If you save the same photograph as a JPEG file then it is still a high quality image with a colour depth of 24 bits but some of the data is lost where it is unlikely to be noticed. Digital cameras mostly store pictures as JPEG files. If you save the same picture as a GIF file then you make the file much smaller as you only use 8 bits per pixel instead of 24 bits. The human eye can tell the difference at this stage. You will see solid blocks of colour instead of gradual transitions in the photograph. However, for small pictures on websites that will only be viewed as a thumbnail, GIF files are fine and take less time to load on a webpage.

Here is a section of a photograph blown up so you can see the difference:

JPEG version



GIF version



Video file types

Video files are mostly stored as MPEG format. There are two versions for different quality requirements though: MPEG-1 is great for low resolution sequences on a website but if you want high resolution full screen video then you need MPEG-2.

Audio file types

Audio files can also be compressed. The demand for music downloads drove the need for a better compression method and MP3 became the dominant compression type. We can download MP3 files from the Internet very quickly and their size means that we can store hundreds of tracks on an MP3 player. Most people cannot tell the difference in quality between an MP3 track and a track from a CD, and you can get about 120 tracks on a CD, if they are in MP3 format.



Document file types

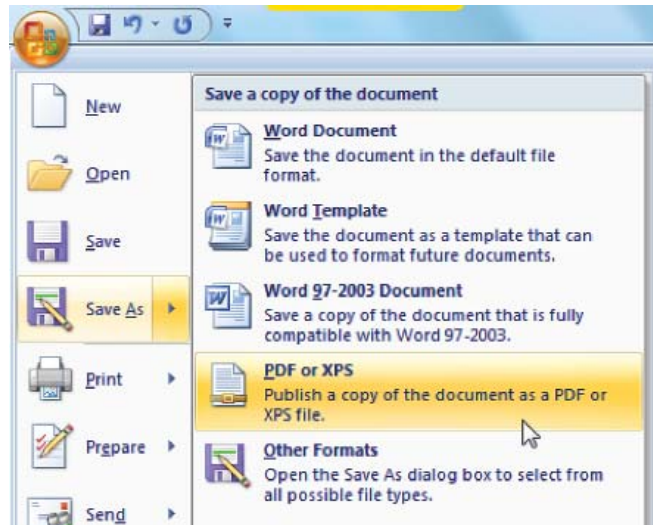
You can create documents in any word-processing or desktop publishing package. Each package will have its own file format. For example, if you create a document in Microsoft Word it will be saved as a .docx file. This is fine when we are working on our own documents but when we are sharing documents across the Internet this format might not suit everyone.

PDF (Portable Document Format) is an open file format, capable of displaying a document on any hardware or any operating system. This means that PDF files can be downloaded from websites anywhere in the world and read on a Windows PC, Apple Mac, tablet, book reader, smart phone or whatever the latest gadget is.

When a PDF document is created it captures all the elements of a printed page as an electronic image that you can view, navigate, print or forward to someone else. When the PDF file is created you can also set options that allow the reader to use copy and paste or block this feature.

Word-processing packages and desktop publishers are also able to write PDF files as well as their own format. The device on which the document is read will need PDF reader software, which is freely available and already installed on many devices.

PDF files are not compressed but you can use a lossless compression algorithm such as ZIP to reduce their size.



Summary of File Types

Type	File suffix	Compression Type	Explanation
Bitmap	.bmp	-	Uncompressed still image file
Portable Network Graphic	.png	Lossless	Colour depth = 24 bits, RGB, 16.7 million different colours
JPEG	.jpg	Lossy	Good for photographs. Colour depth = 24 bits, RGB, 16.7 million different colours
Graphic Interchange Format	.gif	Lossless	Colour depth = 8 bits (only 256 colours) Good for images with large areas of solid colour Ideal for web graphics Although this is a lossless compression, images with higher colour depths are often converted into GIFs to make them smaller (this process is called "quantisation" but is not in the GCSE specification!)
QuickTime	.mov	Lossless	Audio files
Windows Media Player	.wmv .wav	Lossy	Audio files: Files are not as small as MP3
MP3	.mp3	Lossy	Audio files: Designed for downloading music from the Internet. In MP3 format you could fit 120 songs on a CD.
MPEG -1	.mpg	Lossy	Video files: Suitable for small low-resolution sequences on CD
MPEG-2	.mp2	Lossy	Video files: Suitable for full-screen, high resolution video on DVD
Portable Document Format	.pdf	-	An uncompressed document format that is universally accessible.

Glossary of Terms

Networking

LAN	A collection of computers and peripheral devices connected together within a single site.
WAN	A collection of computers and LANs connected together over a geographically remote area, using leased infrastructure.
Topology	A description of how devices are connected together. Does not necessarily represent physical layout.
Bus	A topology where each device is connected to a main cable, referred to as the bus. Any device can transmit at any time but only one transmission can occur on the main bus at any one time.
Ring	A topology where each device is connected to the next in a loop. Uses a token-passing protocol to allow transmission by one device at a time.
Star	A topology where each device has its own cable connecting it to a central device, which can be a switch or a server.
Peer-to-Peer	A method of organising devices in a network where devices are all of equal status rather than having specialised roles. Each computer can access resources on another computer, assuming access rights have been granted by the other computer.
Client-Server	A method of organising devices in a network where some computers have specialised roles: servers. The servers provide resources and services to the other computers, known as clients. Management of the network and shared resources/files is centralised at the server.
Hub	A hardware device that provides connectivity to a LAN cable. A multiport box that has a connection to the LAN from one side and several computers on the other. Can be wireless or cabled.
Switch	A hardware device that is similar to a hub but it has built-in intelligence to direct traffic to the right place. Computers connected to a switch form a star topology LAN.
Wireless Access Point	The device to which a computer connects wirelessly. Can be a wireless hub or a wireless switch.
NIC	Network Interface Card: the card that plugs into a computer to provide a connection to a LAN. Can be wireless or cabled. Holds the MAC address.

MAC Address	A unique hardware number allocated to every NIC. It is a 48-bit address, usually written in hex, e.g. 00-09-7C-F1-F7-85
Message	A communication between devices. Split into packets for sending over a network and put back together again at the other end.
Packet	A fixed size chunk of a message created to send a message over a network. It has its own header containing data such as the destination address and packet number (so the message can be put back together in the right order).
Protocol	A set of rules that defines how devices communicate. E.g. IP, HTTP, HTTPS

Internet

Internet	A public worldwide network where computers and networks in geographically separate locations are connected together using a variety of communication links. Devices communicate using Internet Protocol (IP).
Routers	The hardware devices that make up the backbone of the Internet as well as (smaller ones) providing connectivity from a LAN to the Internet. Use Internet Protocol to communicate with each other.
Modem	The hardware device used to convert the digital transmission from a computer into an analogue signal that can be carried over the analogue telephone network. A method of accessing the Internet.
Digital	A transmission signal that is made up of separate values (numbers), as opposed to the continuously changing signal in analogue transmissions.
Analogue	A transmission signal that is continuously changing, as opposed to being made up of separate values (numbers). Sound in the real world is analogue.
Broadband	A digital method of connecting to the Internet that allows more than one transmission at the same time, e.g: phone and computers. It may use the site's normal phone line or a fibre optic cable to carry transmissions.
WWW	World-Wide Web: a collection of pages distributed on servers connected to the Internet. Uses HTTP to request and send pages to browsers.
HTTP	HyperText Transfer Protocol: the protocol used by a browser to send page requests to a server and also by the server to send back the required page.
HTTPS	A secure version of HTTP where transmissions are encrypted.
IP Addressing	A method of labeling any device connected to the network with a unique numerical value. Uses four bytes usually expressed in this notation: 123.123.003.243
Domain Name	The text label for a website in the Internet: www.bbc.co.uk It corresponds to an IP address for that site.

DNS Servers	Domain Name System server: a database of domain names and associated IP addresses stored on servers. There are many DNS servers distributed across the Internet, which communicate with each other.
HTML	HyperText Markup Language: the programming language used to define the layout and content of a webpage. Uses tags in conjunction with a CSS to control how content is displayed.
CSS	Cascading Style Sheet: defines the formatting and layout of the content defined by the HTML code. E.g. <H1> may be 32pt Arial in Green.
Tags	Labels that go around the content (text, pictures etc) to define the page layout. Eg: <H1> A heading </H1>

Security

User Access Levels	A network policy that defines which users can see which folders and files and the type of access they have to them. Eg: Read-Only or Read-Write.
Encryption	Where the data is changed, using a key, before it is transmitted so that it can only be deciphered by another device with the appropriate key. To anyone intercepting the message it would be unintelligible.
Acceptable Use Policy	An agreement that computer users will sign/agree to before being allowed access to a computer or the network.
Failover	When a hardware component fails, the computer switches over to a redundant component without the service to the user being interrupted.
Redundant	Spare, ready to be used if another component fails. Relates to spare hardware components in fault-tolerant systems that use failover.
Fault-Tolerant	A system that has been designed to cope with hardware failures. Uses redundant hardware and failover usually.
Backup	A copy of data is taken from a live computer system as a precaution against system failure or corruption/deletion of individual files/folders. To be restored in the event of data loss.
Archiving	Files are removed from the main computer system but kept in long-term storage, just in case they are needed in the future or because the law requires they be kept. Creates space on main system.
Disaster Recovery	A collection of precautions that ensures the computer system can be reestablished very quickly after a catastrophe. Includes backup policy, complete hardware system available offsite at short notice and policies to restore data and applications on the replacement hardware.

Compression

Compression	Making files smaller for quicker transmission over a network.
Lossless compression	File is compressed with no loss of essential data.
Lossy compression	Files are compressed by removing some data that is less essential for the purpose. For example, using fewer colours in a picture (reduce colour depth).

Past Exam Questions



June 2011, Question 3

1 A rock band uses an internet website to advertise its music.

(a) The website uses HTML.

(i) Describe HTML. [2]

(ii) Explain the importance of HTML in the creation of web pages. [2]

(b) A list of file extensions for common file standards used on the internet is shown below.

JPG PDF MP3 MPEG ZIP

The rock band allows some files to be downloaded by fans.

Complete the table below to show which file format from the list given above may be used for each of the following files. [5]

File	File Format
A high resolution image of the band to use as a desktop background.	
Sheet music of their songs ready to be printed in the correct format for guitar players.	
A short video extract from their latest concert tour	
A compressed collection of 200 plain text files containing the lyrics of all their songs.	
An audio recording of a song from their album	

(c) Some of the file formats use compression.

(i) Explain the importance of compressing files when transmitting them via the internet. [2]

(ii) Describe the difference between lossy and lossless compression and give an example where each would be used. [4]



Jan 2012, Question 4

2 (a) The table below contains some statements about the internet.

Tick one box in each row to show whether each statement is true or false.

[3]

	True	False
The internet is the same as the World Wide Web		
The internet is a Local Area Network		
The internet is a network between many networks		

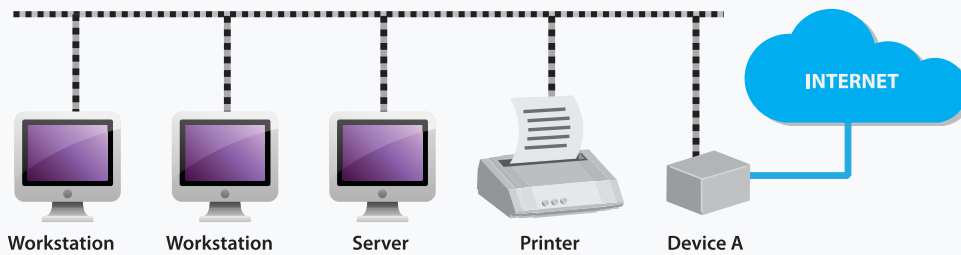
(b) A user types the address www.ocr.org.uk into a web browser.

Describe how a DNS server is used to access this website and explain the advantages of using DNS servers.

The quality of written communication will be assessed in your answer to this question. [6]

June 2012, Question 2

3 The following diagram shows how the computers in Mr Singh’s office are connected to each other to form a LAN.



(a) State the correct name for this network topology. [1]

(b) State the name of the Device A which connects the server to the internet. [1]

(c) Give three functions of the server in this network. [3]

(d) In his home, Mr Singh has a peer-to-peer network.
Explain what is meant by a peer-to-peer network. [2]

Chapter 7: Programming

As part of this GCSE you will have learnt a **high level programming language** such as Python, Delphi or Visual Basic. This topic is all about designing **algorithms**, writing code using best practice and testing the programs to make sure they meet the requirements.

In the A453 unit you will do a controlled assessment programming task, which will use this programming knowledge in a very practical way.

In the A451 theory unit you will also be examined on your understanding of algorithms, coding and testing.

This chapter does not teach you how to program in a specific programming language but addresses the theory that you will need to understand for the A451 exam.

Algorithms

The OCR Specification says that you should be able to:

- understand algorithms (written in pseudocode or flow diagram), explain what they do, and correct or complete them
- produce algorithms in pseudocode or flow diagrams to solve problems

In computing we write programs or create computer systems to “solve a problem”. The **problem** is the need or requirement we have to meet. The solution could be a simple program but is more likely to be a complex suite of hardware and software in a real-world scenario.

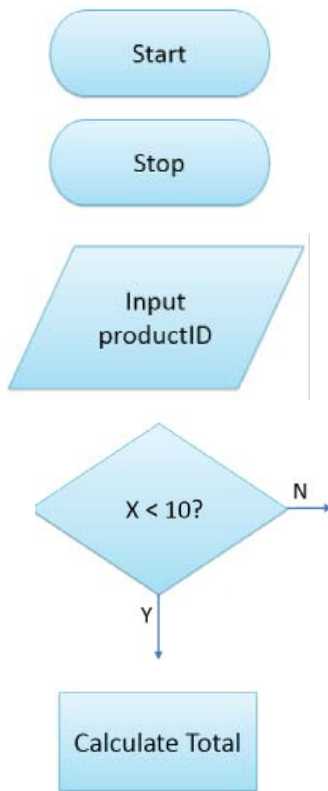
Understanding how to solve the problem is important. You cannot just start coding at line 1 and hope to get a working solution straight away.

The outline of the solution is called an **algorithm**. An algorithm is defined as, “a series of steps to solve a problem”. Algorithms can be expressed in several ways. This chapter will look at creating an outline algorithm with a **flow diagram** (also called a **flowchart**) and then creating a more detailed algorithm from this using **pseudocode**.



Flow Diagrams

A flow diagram or flowchart is an industry-standard design tool, which is used to show an algorithm diagrammatically. There are standard symbols that you will need to understand:



Terminators: Show where the algorithm starts and finishes

Shows an **input** to, or an **output** from, the system. E.g. **Input** ProductID or **Print** ProductDescription and ProductCost

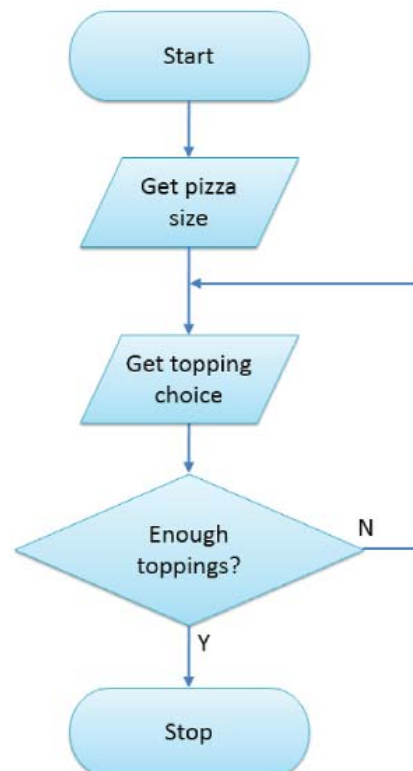
Decision box: Asks a question and has different routes out depending on the answer. In this case if the number X is less than 10 you go down, if it isn't you go right

Process: An action such as a calculation. Always includes a verb. For example, **Add** 10 to X, **Calculate** Total

This very simple example shows the algorithm for taking a pizza order. The program must get the pizza size first and then get one or more toppings.

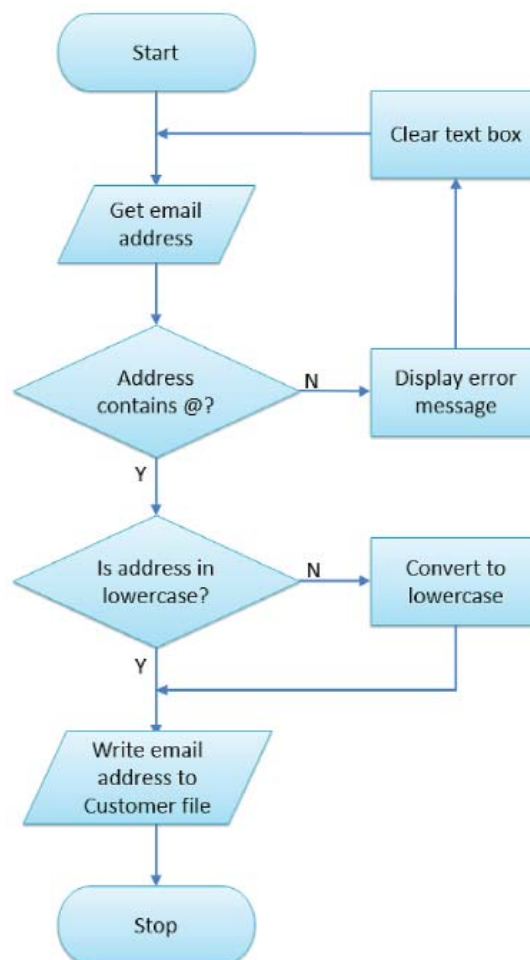
Notice the following:

- There is a loop to get more than one topping. This loop is controlled by the decision box that asks, "Enough toppings?". If the answer is "Yes" to this question the program stops, otherwise it keeps asking for the next topping.
- The line that loops around joins the middle of a line, it does not come into a box.



Here is a more complex problem.

- A computerised form prompts a user to enter their email address.
- The validation rules check if the address has an @ symbol in it. If it doesn't, an error message is displayed, the text box is cleared and the system asks the user to enter the email address again. This continues until an appropriate address is entered.
- The system then checks that the email address has been typed in lowercase, if not it converts it to lowercase.
- Once the email address is ok it is stored in the customer database.
- The flowchart for this could be as follows:



Pseudocode

Pseudocode is used to write an algorithm in programming-style constructs but not in an actual programming language. You do not need to worry about the detailed syntax or be precise about how the code will do something; you just describe the steps you will need in your program. As we said before, an algorithm is a series of steps to solve a problem, so that is all you are doing at this stage.

Once you have fairly detailed pseudocode you can use it as comments in your program and write the actual code under each step.

Control flow in imperative languages

The OCR Specification says that you should be able to:

- understand and use sequence in an algorithm
- understand and use selection in an algorithm (IF and CASE statements)
- understand and use iteration in an algorithm (FOR, WHILE and REPEAT loops)

There are three constructs used to write algorithms in pseudocode (and in actual code):

Sequence

Sequence is just a matter of writing the steps down in the order they need to happen. For example:

```
Input the product price
Input the quantity
Total = quantity x price
Output "Total price is " + total
```

Selection

There are two basic selection constructs that you will learn when you program. IF...THEN...ELSE allows you to choose between two options. By nesting these or having several in a row you can choose between several options but this is more efficiently achieved by the CASE statement. Both constructs are shown below with the key words in bold:

```
IF X <= 10 THEN
    Z = Z + 10
ELSE
    Z = Z-10
END IF ←
```

Not all programming languages use "END IF" but we normally use it in algorithms just to be really clear.

Still indent as you would in your programs though!

The following selections show three different ways of coding a menu system where the user can choose between three options:

Method 1: Using multiple IF statements

```

IF MenuChoice=1 THEN
    Do this thing
END IF
IF MenuChoice=2 THEN
    Do the other thing
END IF
IF MenuChoice=3 THEN
    Self-destruct
END IF

```

These are three separate IF...THEN statements so the computer will have to execute the second and third IF statements, even if the user selected '1' from the menu.

This is not efficient coding but it will work.

Method 2: Using multiple nested IF statements

```

IF MenuChoice=1 THEN
    Do this thing
ELSE
    IF MenuChoice=2 THEN
        Do the other thing
    ELSE
        IF MenuChoice=3 THEN
            Self-destruct
        END IF
    END IF
END IF

```

This still uses three separate IF...THEN statements but they are nested so that the computer will only execute the IF statement tests until it finds one that is true.

This is more efficient than method 1 but for several choices it is difficult for a programmer to follow.

```

CASE MenuChoice OF
    1: Do this thing
    2: Do the other thing
    3: Self-destruct
ELSE
    Write "You must choose 1, 2 or 3"
END CASE

```

The CASE statement is designed for coding multiple choices in a program, such as a menu of several options where the user will enter one choice.

Can you see how much clearer this chunk of code is than the nested-IF version above?

Iteration

There are three basic iteration (loop) constructs that you will learn when you program.

The **FOR** loop allows you to execute a group of steps a specified number of times. It is controlled by a loop counter, which is automatically incremented each time around the loop.

```
FOR count = 1 to 10 DO
  Write count x 3
NEXT count
```

Not all programming languages use "next nnn" but we normally use it in algorithms just to be really clear that **count** is incremented each time around the loop.

A **REPEAT** loop is controlled by a condition **at the end** of the loop. It will, therefore, always execute the following steps **at least once**. Here is an example of an algorithm that uses a REPEAT loop:

```
count = 1
REPEAT
  Write count x 3
  count = count + 1
UNTIL count = 10
```

A **WHILE** loop is controlled by a condition **at the start** of the loop. It will, therefore, execute the following steps **zero or more** times. This is important when you read from a file, for example, when you need to make sure the file is not empty **before** you try to read from it. Here is an example of an algorithm that uses a WHILE loop:

```
WHILE not eof(StudentFile) DO
  Read (StudentFile)
  Write surname, firstname, grade
END WHILE
```

eof means **End Of File**

eof(StudentFile) checks if the file pointer is at the end of the file called *StudentFile*.

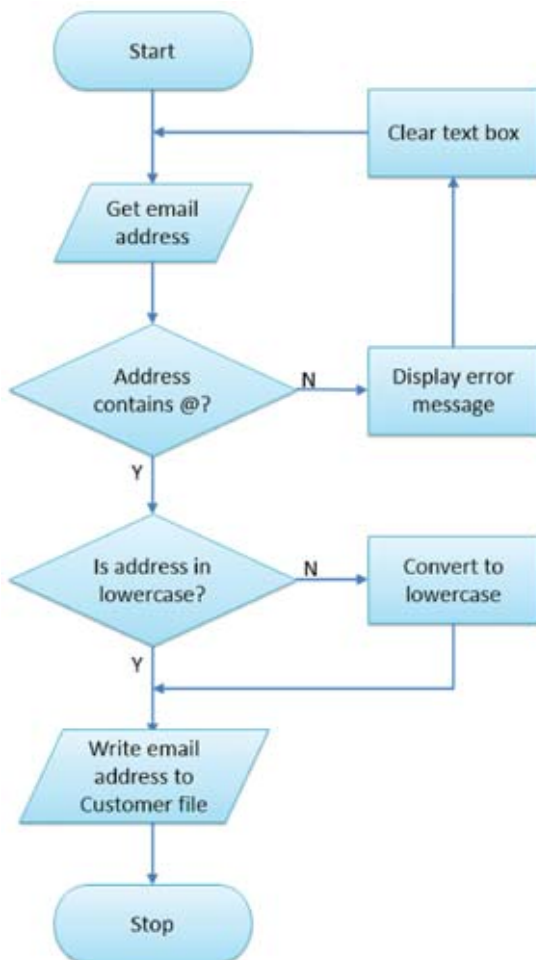
Not all programming languages use "end while" but we normally use it in algorithms to be clear where the loop ends.



For your information...

A **condition** is a **boolean expression** that will be true or false at a particular point in the program. Conditions are used to control iteration and selection statements.

If we convert the flowchart from the previous section into pseudocode it would look something like this:



REPEAT

Input EmailAddress

IF EmailAddress contains "@" **THEN**
HasAtSign = true

ELSE

HasAtSign = False

Output error message

Clear text box

END IF

UNTIL HasAtSign

IF EmailAddress is not lowercase **THEN**
Convert to lowercase

END IF

Write EmailAddress to
Customer file

END of Program

In pseudocode you can use phrases like `IF EmailAddress does not contain "@" THEN...`. At this stage a general statement is fine. When you write the actual code you will need to use the appropriate string-handling function, for example:

```
If InString(variable, '@')=0 THEN (returns 0 if @ not found in variable)
```

Notice that the pseudocode is indented like a real program. This is good practice and makes your algorithm easier to follow. Also, get into the habit of naming items without spaces. For example, here we've used "EmailAddress". The **PascalCase** format (all words capitalised but no spaces) makes it easy to read and means you don't need to use spaces or underscores. You cannot use spaces in identifiers in any programming language so don't use them in your design.



For you to find out...

Investigate PascalCase, camelCase and Hungarian notation. Which of these did you use in the database unit when you named components on your forms?

Programming languages

The OCR Specification says that you should be able to:

- explain the difference between high level code and machine code
- explain the need for translators to convert high level code to machine code
- describe the characteristics of an assembler, a compiler and an interpreter

In *Chapter 4: Data Representation*, we looked at how computers store instructions and data as binary. Program code in binary is referred to as **machine code** and is known as the first generation of programming languages.

To write code at the processor level we use **assembly language**, which is known as second generation language. There are lots of different assembly languages; one for each different processor architecture. The code is written using **mnemonics**, abbreviated text commands such as LDA (LOAD), STO (STORE), ADD.

Human beings find it easier to write programs in languages that are suited to the type of problem they are trying to solve and that look more like normal languages. Third generation languages, otherwise known as **high level languages**, were invented for this. There are lots of different programming languages to suit different types of problem. For example, you might use Delphi to write a forms-based data processing application but you might use Java to code web-based applets.

Independent of hardware (portable)	3rd Generation	High Level Languages	In statements: Rate:=3.02 Used:=5672 BillAmount:=Rate*Used
Language is processor specific	2nd Generation	Assembly Language	In mnemonics: LOAD #34 ADD &4F3A STORE &39FC
	1st Generation	Machine Code	In binary: 1010100011010101 0100100101010101

Hardware

Comparing high level languages and machine code

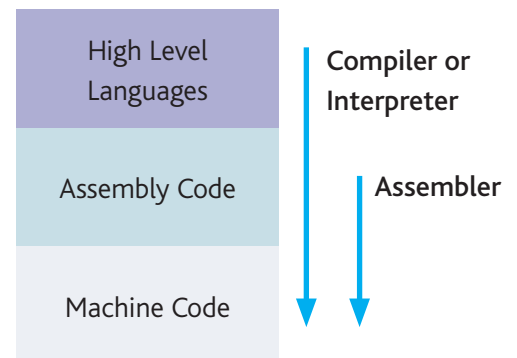
Features of the two types of languages can be compared as follows:

Machine Code	High Level Languages
Processor specific: one instruction set per processor architecture, which won't work on another machine	Portable: will work on different types of machine/processor
Designed with hardware in mind	Designed with a type of problem in mind
Machine code operations relate directly to an assembly language command; one-to-one relationship	One high level program instruction will be translated into several machine code instructions

Translating programs into machine code

Whatever language a program is written in, it must be translated into machine code so it can run on the processor. Translators are a type of **system software** (see *Chapter 3: Software*). There are three types of **translator** program that do this:

- Assemblers:**
 Convert assembly language into machine code. A simple conversion as every assembly language instruction is translated into a single machine code instruction.
- Compilers and Interpreters:**
 Convert a high level programming language into machine code. This is a more complex translation as a single instruction can result in many machine code instructions.



A compiler translates a high level programming language into machine code. The machine code version is then run by the computer. This compiled version doesn't need the original **source code** or the compiler in order to run. When you buy commercial software you are buying an executable version called **object code**.



The benefit of using a compiler is that object code can be distributed to customers easily, without giving them your valuable source code. If you distributed source code, the market could be flooded with illegally copied and amended versions within days. (Some people are happy to share their code and deliberately release their software as Open Source, see *Chapter 3: Software*).

The disadvantage is that any software problems that occur in the object code version are harder to diagnose.

An interpreter is an alternative way of translating high level programs. It translates and executes one line at a time so it uses far less memory than a compiler. The interpreter is often used when software is being developed as it is easy to pinpoint where there is an error in the code.

The disadvantage of interpretation is the speed of execution. Programs run much slower as every line has to be translated first. No object code is created; the interpreter translates the program every time it is run. If, for example, you have a loop in the program where instructions are executed several times, the interpreter will translate them from scratch every time around the loop.

The table below compares compilers and interpreters:

Compiler	Interpreter
Translates the whole program to produce the executable object code	Translates and executes one line at a time
The object code is the version that the computer runs	The source code is the version that the computer runs
Faster run time because the program already in machine code	Slower run time because the program is translated every time it is run
Customers cannot see the actual code you wrote when you distribute the program	If you distributed source code with an interpreter then customers would have your actual code
Used for distributed software	Used in development

Handling data in algorithms

The OCR Specification says that you should be able to:

- define the terms variable and constant as used in an imperative language
- use variables and constants
- describe the data types integer, real, Boolean, character and string
- select and justify appropriate data types for a given program

Constants

In a program there are certain values that remain the same (constant) while the program runs. The programmer could use the actual value in the code each time but it is considered good practice to give the value a unique name (an **identifier**) and then use that name throughout the program. We declare **constants** at the start of our programs and then refer to them as needed in the code. For example:

At the start of the program: `CONST`

`VATRate = 0.2;`

Later in the program code: `SellPrice := NetPrice * VATRate + NetPrice;`



Please note, though, this does not mean that the constant's value will never change during the lifetime of the system! For example, the VAT rate will stay the same (constant) while the program runs in the shop each day but, after the next budget, it may change. The value allocated to that constant will then need to be edited in the program.

The two main benefits of declaring a constant are:

- When its value changes, you only have to edit it in one place rather than looking for every place in the program where you used that number.
- The code will be easier to read and understand because the constant's identifier will be used instead of a number. This makes your code easier to debug and, later on, maintain.

Variables and data types

When a program runs it processes data, which must be stored in memory while the program is running. We need to name these locations so we can refer to them in the program. As the program runs, the values in these locations might change so they are called variables. For example: a variable called "total" might change several times as many numbers are added to it. A variable called "surname" will hold different values as the program processes a list of customer orders.

Each variable has an **identifier** (unique name) that refers to a location in memory where the data item will be stored. Each variable also has a **data type**, which defines:

- the type of data that will be stored at the memory location and therefore the operations that can be performed on this data item (described later in this chapter)
- how much space will be needed for that data item

We normally declare variables at the start of a program so when the program runs, the appropriate memory can be reserved to store the data. The following is an example from a Delphi program:

```
Var
    num1, num2    :integer;
    total        :real;
    choice       :char;
    username     :string;
    found        :Boolean;
```

The word “var” tells the computer that variable declarations follow. The word to the left of the colon is the variable’s identifier/name. The word to the right of the colon is the **data type** (see below).

The table below shows a list of data types and the typical amount of memory that each would need. Note that the amount varies for different programming languages so an integer may need 2 bytes in one language but 4 bytes in another.

Data Type	Type of Data	Typical Amount of Memory
Integer	A whole number, such as 3, 45, -453	2 bytes
Real	A number with a fractional part such as 34.456, -9.234	4 bytes
Char	A single character, where a character can be any letter, digit, punctuation mark or symbol that can be typed	1 byte
String	Zero or more characters. A string can be null (empty), just one character or several characters	1 byte per character in the string
Boolean	A Boolean variable has the value True or False	1 byte*

*High level programming languages don’t store anything in less than a byte.

Operations on common data types

The OCR Specification says that you should be able to:

- perform common operations on numeric and Boolean data

Operations are things you can do to specific types of data. For example, you can perform arithmetic operations on numbers and you can perform string-handling operations on text.

Numerical data types:

Operations that can be performed on numerical data types are as follows:

Arithmetic operations

(give a numerical result)

eg: $25 + 3 = 28$

- + (addition)
- - (subtraction)
- * (multiplication)
- / (division)
- DIV (integer division)
- MOD (modulus)

Comparison operations

(give a boolean result: true or false)

eg: $456 > 34$ is true

- < (less than)
- > (greater than)
- <= (less than or equal to)
- >= (greater than or equal to)
- <> (not equal to)
- = (equal to)

When performing operations on data items you need to consider the data types used. For example, in a simple calculation where two whole numbers are added together, variables could be defined as follows:

```
Var
  Num1, num2, total : integer;
```

But if the calculation involves division, then the answer variable must be declared as a real number:

```
Var
  Num1, num2 : integer;
  answer      : real;
```

The arithmetic operators DIV and MOD can only be performed on whole numbers (integers).

DIV is integer division. It works like normal division but returns the whole number of times one number goes into the other. Here are some examples:

$13 \text{ DIV } 3 = 4$ $300 \text{ DIV } 30 = 10$ $305 \text{ DIV } 30 = 10$

MOD gives the remainder of integer division as follows:

$13 \text{ MOD } 3 = 1$ $300 \text{ MOD } 30 = 0$ $305 \text{ MOD } 30 = 5$

Boolean data type:

Boolean variables are either true or false. It makes no sense to perform mathematical operations on them or to compare them to see which is greater. With boolean variables we use **logical operators** to create **boolean expressions**. (Boolean expressions were used earlier to control loops, where we called them conditions.) Logical operations you should know are:

NOT AND OR

Consider an estate agent's program that searches through a file of house details to find ones that match a customer's requirements. In this case the customer wants a house or flat, but it must have more than three bedrooms:

```
IF (NumberOfRooms>3) AND ((type="House") OR (type="Flat")) THEN
    Output details
END IF
```

Notice the extra set of brackets around the second half of the expression. AND takes precedence over OR so without the extra brackets the program would return all the houses with more than three bedrooms as well as any flats, whether they have more than three bedrooms or not.

The table below summarises the operations that can be performed on each data type.

Integer	Real	Boolean
Arithmetic operations Comparison operations	Arithmetic operations (but not DIV and MOD, which only apply to whole numbers) Comparison operations	Logical operations

Arrays

The OCR Specification says that you should be able to:

- use one-dimensional arrays

Earlier in this chapter we talked about variables. If we were processing one or two specific data items, then we would have a variable for each of these. For example, a program that adds two numbers together might use variables called `num1`, `num2` and `total`, all of type integer.

Often a program will process a number of data items of the same type; for example, a program processing the results for 5,000 people who have run a marathon race, may need to sort and print these in order of their race times. All the records need to be held in memory while the sorting is done.

We could use variables called ID1, Result1, ID2, Result2, ID3, Result3... ID5000, Result5000 to store the IDs and times of each runner but most programming languages allow you to use an **array** to make processing groups of data easier to code. An array is a group of data items of the same data type, which is stored under one identifier (name) in contiguous (one after another) memory locations.

Simple (1-dimensional) array

This program processes 12 numbers using a simple array called "numbers". Imagine a table with 1 row of 12 boxes:

Numbers:

1	2	3	4	5	6	7	8	9	10	11	12

Each box in the table can contain an integer. Each box has a numerical reference called a **subscript** that is used to refer to that individual data item. For example, the third box in this array is referred to as:

```
numbers [3]
```

At the start of a program the array will be defined, just as you would a variable. In Delphi the array declaration for the array shown above would look like this:

```
numbers : array [1..12] of integer;
```

The individual boxes in the array can be used just like variables:

- **Assign** values to them: `Numbers [4] =27`
- **Input** values to them from the keyboard or a file: `Input numbers [4]`
- **Output** the value stored in a box to the screen or a file:

```
Output "The fourth value is " + numbers[4]
```

The benefits of using arrays are:

- Code is easier to follow and therefore easier to debug and maintain.
- A group of data items can be easily processed using a FOR loop.

When you process a group of data in an array you typically do the same thing to each data item, so having them stored in numbered locations makes this much easier and quicker to code.

The following **algorithm** gets 12 numbers from the user, adds them up and outputs the total.



For your information...

Some languages, such as Java and C++, start numbering arrays at zero rather than 1.

Some languages use round brackets such as `numbers (3)`.

```

total=0
for loop = 1 to 12 do
    output "Enter number "+loop
    input numbers [loop]
    total = total + numbers [loop]
next loop
output "Total is "+total

```

Initialise the variable **total** to zero at the start

Each number the user types is stored in the numbered location in the array, based on the value of **loop** each time; loop=1 the first time, 2 the second time etc.

Refers to the current value of **loop** to add a specific number to the total

Errors

The OCR Specification says that you should be able to:

- describe syntax errors and logic errors which may occur while developing a program
- understand and identify syntax and logic errors

When you write a program in a high level programming language, a translator (compiler or interpreter, see earlier in chapter) will scan each line of code and convert it into machine code. As you will already have found out, programming is not as easy as it looks!

- Firstly, it is very easy to make mistakes typing in the code, for example typing "prnt" instead of "print". These are **syntax errors**.
- Secondly, once you have corrected all the syntax errors, the code may run but not do what you want. This means there are **logic errors** in your program.

Syntax errors

The translator expects commands to have a certain format, called **syntax**, just like a sentence in English has grammar rules. Syntax is a set of rules which define the format of each command. For example, in Delphi you would define variables in the format:

```
VariableName : Datatype;
```

The compiler knows that this statement must have an identifier followed by a colon, followed by a recognised data type and a semi-colon at the end.

In Visual Basic an IF statement must have this format:

```

IF x=10 THEN
    Grade="Pass"
ELSE
    Grade="Fail"
END IF

```



For your information...

If you spell identifiers, such as variable names, incorrectly this is not a problem, as long as you always spell them wrong in the same way.

Without the correct key words the compiler will not be able to translate it into machine code and will give a syntax error.

Some common syntax errors include:

- Mistyping a key word: `WRIET` instead of `WRITE`
- Missing key words out of constructs such as starting a `REPEAT` loop but not writing `UNTIL` anywhere
- Opening brackets but not closing them
- Not having the right number of parameters inside brackets for functions, for example:
`Answer = round(TheNumber)` will give a syntax error if the language expects another parameter to state the number of decimal places: `Answer = round(TheNumber, 2)`

A program will not compile or run if there are syntax errors.

Logic errors

Once the code is written correctly, with no syntax errors, the program will compile. The program can then be run – but just because the program will run does not mean that it is working correctly. Often when the program is run it doesn't do quite what was expected. This is called a **logic error**.

Typical logic errors that you have probably coded already include:

- Missing brackets from mathematical calculations:
`NetPay = GrossPay - TaxFreePart x TaxRate`
Is not the same as:
`NetPay = (GrossPay - TaxFreePart) x TaxRate`
- Loops that do not execute the correct number of times because a condition states `X>10` instead of `X>=10`.
- Variables that have not been initialised, or have been initialised in the wrong place (often incorrectly initialised inside the loop instead of just before it).
- Flawed algorithms that just don't do what they were intended to do. Capturing all of the complexities of real-life scenarios in code is difficult and users always manage to do something to the input that you didn't cater for!

Often these logic errors are hard to spot. You should always do a visual check of output to check it isn't ridiculous but you also need to do some systematic testing to make sure the program really does behave as expected (see next section).



For your information...

Remember BIDMAS?

Brackets first, then multiply and divide and then add and subtract!

Testing

The OCR Specification says that you should be able to:

- select and justify test data for a program, stating the expected outcome of each test

How do you know if your program really works? It may run and produce output but you need to check that the output is as expected. If you made a spreadsheet that multiplied pounds by a conversion rate to get dollars you would probably type in an easy number like £10 to see if the number of dollars looked right. We test programs in a similar way. We decide on some sensible test values to put into the program to test that the outputs are what we expect.

Planning how to test a program

A good way of planning your testing is to write down what sample data you will use to make sure the program works correctly, whatever the input. This is a "Test Plan". It is important that you think about this before you write the program so you don't move the goalposts later. Remember, the aim of testing is to try and identify, for example, errors in calculations or user input which the program cannot handle, not to prove that the program usually works correctly.


You can formalise your plan by using a table with the following headings:

Test purpose	Test data	Expected outcome	Actual outcome

The first three columns are part of the design and planning stage and the final column is completed when you test the finished program.

If numbers or dates are being entered by the user then you should use test data that checks both ends of the allowed range as well as data that should not be allowed, just outside this range. This is known as testing **boundary data** or **extremes**.

✗	✓	✓	✗
0	1 to 10		11



Allowed range of
input values

! For your information...

The term "extremes" doesn't mean extremely big or extremely small but refers to the extremes of the range, the smallest and largest values allowed.

The test plan for testing this validation would look like the one shown over page.

Test purpose	Test data	Expected outcome
1 Check user can only enter number between 1 and 10	1 (lowest valid number)	Input is accepted
2 Check user can only enter number between 1 and 10	10 (highest valid number)	Input is accepted
3 Check user can only enter number between 1 and 10	0 (not allowed)	Error message is displayed and user is asked to enter number again
4 Check user can only enter number between 1 and 10	11 (not allowed)	Error message is displayed and user is asked to enter number again
5 Check user can only enter number between 1 and 10	5 (typical number)	Input accepted
6 Check user can only enter number between 1 and 10	? (non-numerical entries not allowed)	Error message is displayed and user is asked to enter number again

If text is being entered, check what happens when you enter nothing or a string that is too long, as well as the text you would expect to be accepted.

Test purpose	Test data	Expected outcome
7 Check user password works	"frogs" (correct password)	User allowed to continue
8 Check user password works	"cats" (correct password)	Input is accepted
9 Check user password works	"" (no password)	Error message is displayed and user is asked to enter number again

IDE tools

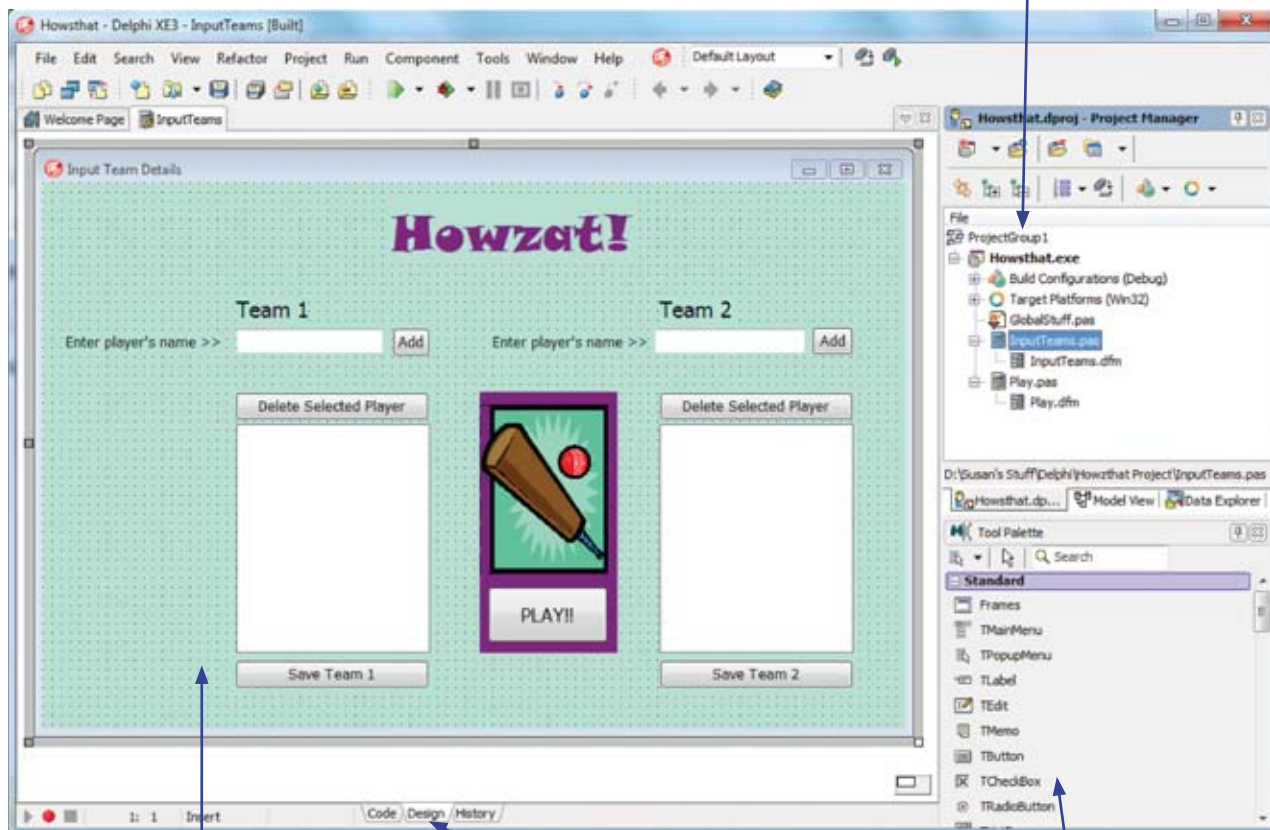
The OCR Specification says that you should be able to:

- describe common tools and facilities available in an integrated development environment (IDE): editors, error diagnostics, run-time environment, translators, auto-documentation

When you create a program you will be using a software package that helps you write the code more easily. This is called an **Integrated Development Environment** or **IDE**.

The screenshot below shows the form design view in the Delphi 2007 IDE. The key features labeled are common to most IDEs.

An overview of the program files that make up a "project". A project may consist of many different forms, all connected together.



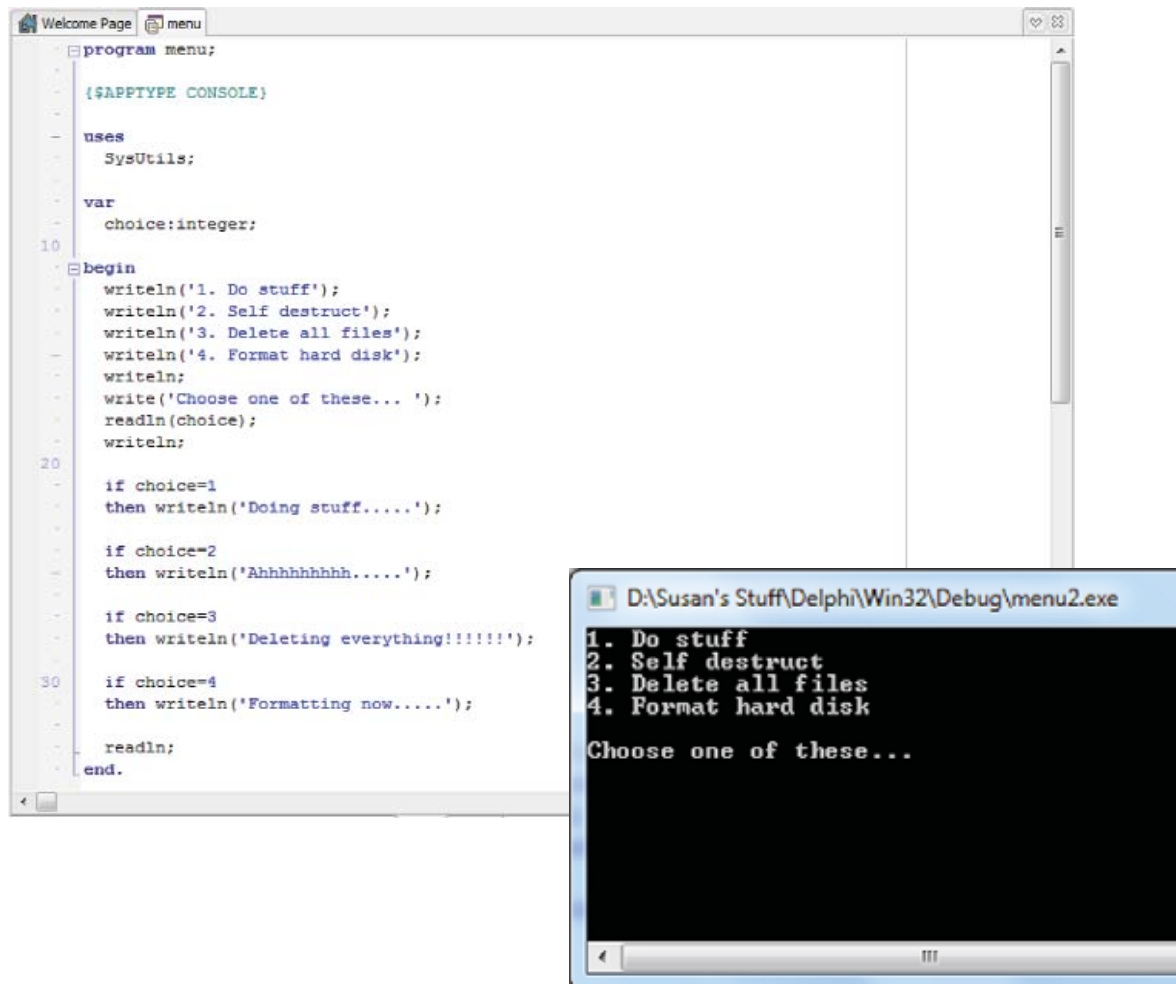
A WYSIWYG view of the form being designed. The developer can drag and drop components onto this form.

Tabs to toggle this view and the code view.
(Code view on next page)

A palette of components that can be put on the form such as text boxes and buttons.

Editing environment

When you write the code the IDE will provide helpful features in the editing environment such as colour-coding key words, numbering the lines and even auto-completing constructs for you.



The line numbers make it easy to see which bit of code an error message relates to. The auto-completion of constructs reduces errors such as missing "ends" or starting a "repeat" loop and forgetting the "until" part.

Run-time environment

When you run the program the IDE will provide the translator needed to run the program, in this case a compiler. The screen and the feedback you see when you try to run the program are provided by the **run-time environment**.

When you try to compile the program the IDE will generate error messages in a diagnostics window. Notice how the IDE has highlighted the line in red. This is the line causing the error and the message below that says, **Undeclared identifier 'Out'** is because the words **Not Out** should be in quote marks in the code.

```

procedure TFormTeamInput.btnT2SaveClick(Sender: TObject);
begin
  reset(team1);
  lstT2Players.ItemIndex:=0;
  while lstT2Players.ItemIndex+1<lstT2Players.Items.Count do
  begin
    with aPlayer do
    begin
      PlayerID:=lstT2Players.ItemIndex+1;
      PlayerName:=lstT2Players.Items.Text;
      NumRuns:=0;
155   HowOut:=Not Out;
    end; // of building one player record
    write(team2,aPlayer);
    lstT2Players.ItemIndex:=lstT2Players.ItemIndex+1;
    end; // of adding one player to the file
160 end;

```

Messages

```

[dcc32 Error] InputTeams.pas(155): E2003 Undeclared identifier: 'Out'
[dcc32 Fatal Error] Howthat.dpr(5): F2063 Could not compile used unit 'InputTeams.pas'
Failed

```

As well as assisting the programmer with error messages there are other debugging tools that allow the programmer to stop the program running at a certain line (called a **breakpoint**) and allows them to check the values of certain variables. This is a very useful debugging tool.

Breakpoint on line 21 (red dot).
 When the program runs it will stop here.

At line 23 the programmer can see that the variable "choice" has the value 3

This is where execution has reached

```

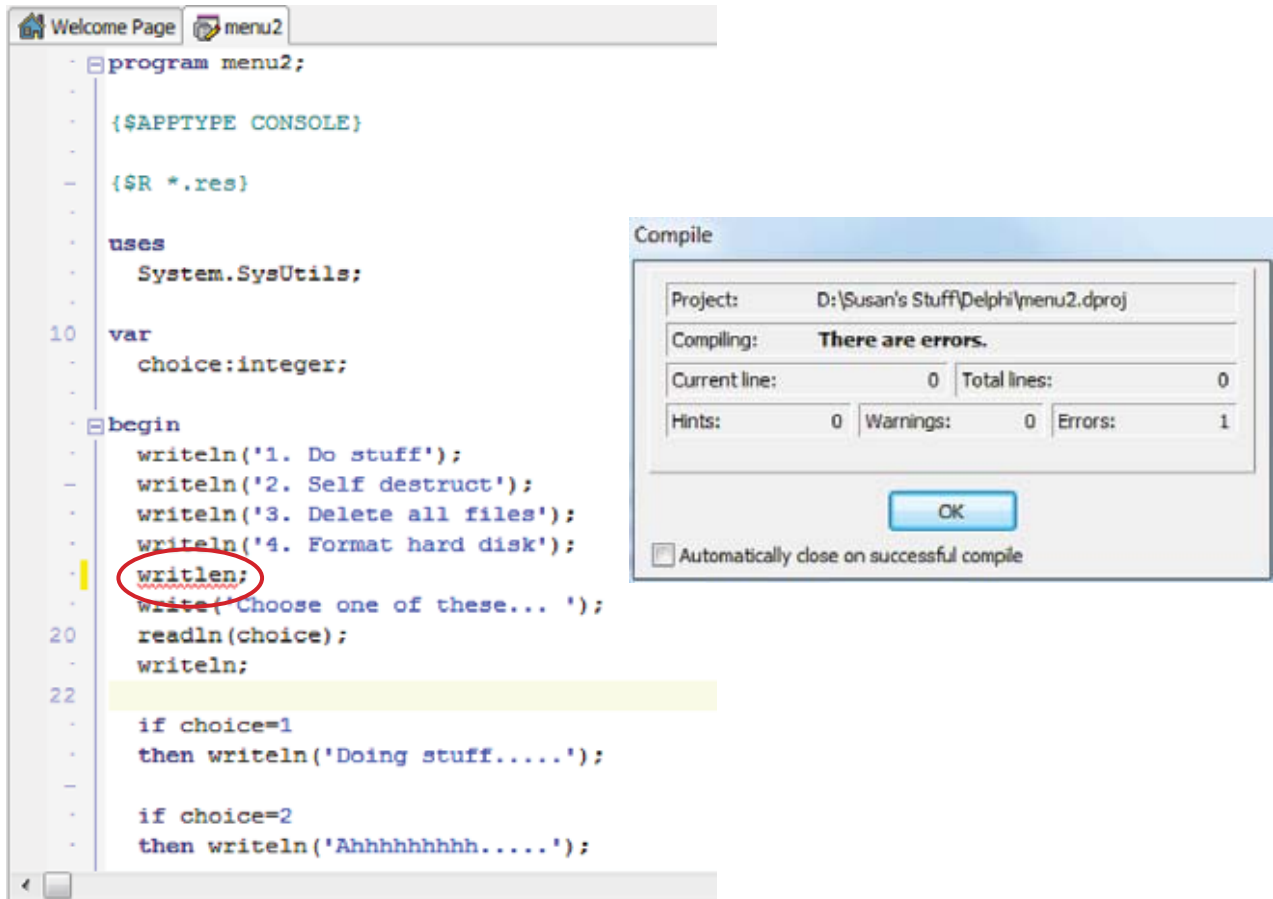
begin
  writeln('1. Do stuff');
  writeln('2. Self destruct');
  writeln('3. Delete all files');
  writeln('4. Format hard disk');
  writeln;
  write('Choose one of these... ');
  readln(choice);
  writeln;
  if choice=1
  then writeln('Doing stuff.....');
  if choice=2
  then writeln('Ahhhhhhhhh.....');
  if choice=3
  then writeln('Deleting everything!!!!!!');
  if choice=4
  then writeln('Formatting now.....');

```

In this case there is a breakpoint at line 21 so the programmer can then step through the program one line at a time. The programmer can hover over a variable to see what value it has. Here you can see that "choice" has the value 3. Sometimes the value isn't what you'd expect.

Translators

When the code has been written it must be translated into machine code before it can run. An IDE includes the translators that will either compile or interpret the code, as required (see earlier in chapter, for an explanation of the different translators). The compiler will check the syntax and report any errors. The program will compile fully once all the syntax errors have been removed, and convert the high-level programming code into machine code. The program shown on the next page will not run because there is a syntax error on line 18.



Auto-documentation

When a developer writes a program, it is good practice to use comments in the code. The compiler recognises comment lines because they start with a special symbol. For example:

```

// This is a comment in a Delphi program
' This is a comment in a Visual Basic program
<!-- This is a comment in HTML -->

```

These comments can then be used to create program documentation automatically.

The IDE takes a note of all the variables, modules and sub-routines as the project is developed. The compiler uses this data along with the developer's comments to generate a text file that can be the basis of the programmer's technical documentation. Technical documentation is important for program maintenance.

Glossary of Terms

Algorithms

Algorithm	A series of steps to solve a problem. Can be expressed in structured English, pseudocode or as a flowchart (also called flow diagram).
Flowchart/Flow Diagram	A diagram using commonly defined symbols to express an algorithm.
Structured English	A way of writing an algorithm in natural language using some basic programming constructs such as IF...THEN...ELSE and loops. More structured than just natural language/prose.
Pseudocode	A way of writing an algorithm that is close to actual programming language, using coding-style constructs such as IF...THEN...ELSE, loops and array notation as appropriate.
Hungarian Notation	The convention of prefixing identifiers to indicate what type of object they are. Commonly used with forms where, for example the prefix txt might indicate a textbox and lst might prefix a list box. The prefix is conventionally in lowercase.
camelCase	The use of capital letters in an identifier to make them more readable. With camelCase the first word is not capitalised: <code>txtCustomer, lstCounty, frmAddCustomer, btnSubmit</code>
PascalCase	The use of capital letters in an identifier to make it more readable. For example, variable and procedure identifiers: <code>StudentNumber, ProductID, StartPoint, CalcTotal</code>
Sequence	Where instructions are executed one after another in series.
Selection	Where the program will execute certain instructions based on conditions. Selection statements include: IF...THEN...ELSE and CASE...OF to select which commands to execute.
Iteration	Where a program will execute a group of instructions zero or more times based on a condition. FOR loops will execute instructions a specific number of times, REPEAT...UNTIL loops one or more times and WHILE...DO loops zero or more times.
Condition	A boolean expression that controls an iteration or selection statement. For example, REPEAT...UNTIL <code>X=10</code> , where <code>X=10</code> is the condition.
Boolean Expression	An expression that is true or false. For example: <code>continue="Y"</code> Expressions can be more complex, containing several parts: <code>((continue="Y") or (continue="y"))and (tries<10)</code>

Programming languages

High Level Programming Language	A programming language where programming constructs are written in a way that is close to natural language instead of in mnemonics or machine code. For example, Delphi, Pascal, Visual Basic, Java, C++.
Imperative Language	Programming language such as Python or Delphi which uses a sequence of statements to determine how to reach a certain goal or solve a problem.
Assembly Language	Second generation programming language where instructions are in the form of mnemonics.
Mnemonics	Abbreviations representing commands used in assembly language programming. For example, LDA, STO, ADX.
Machine Code	First generation code; binary instructions where some bits are used to define the operation (called the "opcode") and some bits define the data to be used.
Translator	The piece of systems software used to convert different programming languages into machine code. Three types: assembler, interpreter and compiler.
Interpreter	A translator that converts high level languages into machine code one line at a time, checking syntax, converting to machine code and executing the code.
Compiler	A translator that converts high level languages into machine code. Works through the whole program (source code) checking the syntax, then converting to machine code and creating the executable object code, which can be saved. The object code is executed, not the source code.
Source Code	The original high level program.
Object Code	The executable version of the program after it has been compiled.
Assembler	The translator that converts assembly language programs into machine code.

Handling Data in Algorithms

Identifier	A unique name for something (variable, constant, program, procedure etc.) within the program.
Constant	A named value within a program that has a specific value. Its value does not change while the program is running.
Variable	An identifier associated with a particular memory location, used to store data. Its value may change as the program is run and new values are assigned to it.
Data Type	A formal description of the type of data being stored in a variable. It defines the amount of memory required and the type of operations that can be performed on that variable. Typical data types include Integer, real, string, boolean
Integer	Data type for whole numbers. Typically uses 2 bytes.
Real	Data type for fractional numbers. Typically uses 4 bytes.

Char	Data type for a single character. Typically uses 1 byte.
String	Data type for text, zero or more characters. Typically uses 1 byte per character.
Boolean	True or false. Typically uses 1 byte.
Operations	The actions that can be performed on a variable.
Arithmetic Operations	Add, subtract, multiply, divide, integer division (DIV) and modulus (MOD)
Comparison Operations	= < > <= >= <>
Logical Operators	NOT AND OR
Boolean Expressions	Expressions that resolve to true or false such as $X \leq 20$
Array	A group of data items of the same data type that use a single identifier. Individual data items are accessed using a subscript.

Errors and testing

Syntax	A set of rules that defines how program statements must be written in order for the translator to understand them.
Syntax Errors	An error in the format of the program statements such as missing semicolons or keywords spelt incorrectly.
Logic Errors	An error in the algorithm that means the outcome is not as expected, even though the program will run.
Valid Data	Data that should be allowed by the program. For example, if a range of 1 to 10 is allowed, then valid data will be any number between 1 and 10 inclusive.
Invalid Data	Data that is not valid and should be rejected by the program. For example, if a range of 1 to 10 is allowed, then invalid data will include abc, -251, 0, and 11
Boundary Data	Data either side of the range extremes. For example in a range of 1 to 10 boundary data will include 0, 1, 10 and 11

Past Exam Questions



June 2013, Question 7

- 1 Julie is writing a computer game that simulates a 100m race. Each time the space bar is pressed, the position of the player moves up by 1. When the position reaches 100, the player has won.

Here is Julie's algorithm for the program.

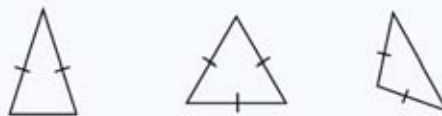
```

CONST PlayerKey = " "
Position = 0
REPEAT
    INPUT KeyPressed
    IF KeyPressed = PlayerKey THEN
        Position = Position + 1
    END IF
UNTIL Position = 100
  
```

- (a) State an example of a constant and a variable in the algorithm above. [2]
- (b) State what is meant by selection and iteration using examples from Julie's algorithm. [4]
- (c) To make the game more interesting, Julie changes the rules. Each time the spacebar is pressed, the position of the player will now move up by a random number.
- State **two** changes that need to be made to include this new rule. Justify each change. [4]

June 2013, Question 10

- 2 An isosceles triangle is a triangle that has at least two equal length sides. The diagram below shows examples of isosceles triangles. In each diagram the marked sides are equal.



Write an algorithm for a computer program that determines whether a triangle is an isosceles triangle.

- The user inputs the lengths of the three sides as Length1, Length2 and Length3
- If any two sides have the same length the program outputs "Isosceles"
- Otherwise the program outputs "Not Isosceles"

[5]



Jan 2013, Question 9

3 Charley is writing a program for music students. To make sure that there are no logic errors in the program, Charley uses a test plan.

- (a) Describe what is meant by a logic error. [2]
- (b) The program uses the letters in the following list to represent musical notes.

C D E F G A B

When the user inputs a letter from this list, the program outputs the next three notes in the list. If it gets to the end of the list, it starts again from the beginning, so the next note after B is C.

Complete the test plan below by stating, for each input data, the expected outcome and a reason for the test. [6]

Input Data	Expected outcome	Reason for test
C		
A		
H		

Appendix: Answers to Exam Questions

These answers are good responses to the question that cover the mark scheme but probably give more detail than absolutely necessary – you want to make sure you get the marks so don't be too minimalist. Hints and tips on how to approach answers are in the grey boxes like this:

*

When you answer the questions you don't know what the mark scheme is going to say – you have to give all the information that is relevant in clear and concise phrases. Notice how many marks there are; if it is a 2 mark question try to make three specific points in response

Answering Longer Exam Questions

With questions worth more than one mark it is important to make sure you answer the question fully.

This is a method that may help you. It's called **BUMP**.

- **Box** the keywords that say what you have to do, e.g. **State** ..., **Describe** ..., **Explain** ...
- **Underline** the bit you have to do it to, e.g. **Describe** advantages of using a computer system...
- **Make sure** there isn't another bit to the question, e.g. and the need for this system to be reliable.
- **Plan** your answer, don't just start writing.

And remember, in Computing you can use **headings, bullet points, diagrams, tables**, etc. You don't have to write complete sentences in paragraphs like an essay. Pick the best method to communicate effectively. For example, for an algorithm this may be pseudocode or a flowchart.

Chapter 1: Fundamentals Of Computer Systems

June 2011, Question 1 (part question)

- 1 (a) Storage device: Hardware that is used to store files/data long term, non-volatile, data stored in binary. E.g. Hard disk

Input device: Hardware that is used to enter data into the computer, takes real-world data and converts it into a digital form that can be stored on a computer. E.g. keyboard, sensor

Output device: Hardware that presents the results of processing to the user or actuators that perform a task automatically. E.g. monitor

*

Define and give an example

Jan 2012, Question 1

- 2 (a) text document: kilobyte, movie clip: megabyte, surname: byte

*

Make sure you use one of the words given in the question

[3]

- (b) 1 terabyte = 1024 gigabytes, 2 terabytes = 1024 x 2 = 2048 gigabytes

[2]

Jan 2013, Question 7

- 3 Advantages of using a computer system:
- A program will be fairer – not subject to human bias
 - Can implement an algorithm consistently
 - Won't forget some patients
 - Can have same software in more than one place
 - Will be able to look back at how the A&E performed over time

* *Use headings to make it obvious that you have addressed everything in the question*

* *Use bullet points to write less but say more - this is not an essay*

Has to be reliable because:

- Lives may be at stake if some patients are not seen quickly enough
- If there is a system failure no one will know who is next – could take a while to sort it out and may lose some data on performance

[6]

Chapter 2: Hardware

Jan 2012, Question 8

- 1 (a) Stores programs that are running
Stores data being used by the programs that are running [2]
- (b) (i) When part of the hard disk is used as an extension of main memory (RAM) to store parts of files that are not currently being executed. Sections are moved into main memory when needed. [2]
- (ii) To allow more programs to be run at the same time when there isn't enough main memory to hold them all. [1]
- (iii) Main memory has a faster access speed than virtual memory.
More main memory means that more programs fit in main memory...
...so virtual memory will be accessed less frequently...
...and programs will run faster. [2]

June 2012, Question 7

- 2 (a) Any two of the following:
- Fetches instructions from the main memory
 - Fetches data from main memory
 - Decodes instructions
 - Executes instructions

* *You must use the correct technical terms, "information" or "programs", is not correct*

(b) Clock speed:

The higher the clock speed the more fetch-execute cycles per second...

...so more instructions are processed per second...

...so the CPU performance is faster

*

A good approach to this question is to state a fact and what the implication is

[2]

Cache size:

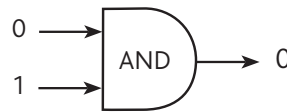
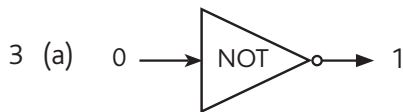
Cache has a faster access speed than main memory.

More cache means that the CPU spends less time accessing main memory,

...so programs will run faster.

[2]

Jan 2013, Question 3



[2]

(b)

p	q	(NOT p) AND q
0	0	0
1	0	0
0	1	1
1	1	0

*

Only use 1 or 0 in a truth table, not ticks and crosses or true and false!

[3]

*

The columns for p and q must show all the possible combinations of values, the order does not matter

Chapter 3: Software

June 2011, Question 5

1 (a) Antivirus: Scans the computer periodically. If a virus does install itself, it detects and removes it. Prevents harmful programs being installed and important files being changed.

[2]

(b) Disk defragmenter: Optimises the use of the hard disk space by collecting together the separate parts of each file in one location on the disk. Also groups together the free space so newly saved files do not have to be split up. Speeds up file access time.

[2]

*

These are pretty much the definitions given in the glossary. Two marks so make sure you say enough

Jan 2012, Question 2

- 2 (a) To allow multiple programs to run on a single CPU. They appear to be running at the same time but they are sharing the processor and main memory. [2]
- (b) Any two of:
- Providing a user interface
 - Providing a platform for applications
 - Memory management
 - File/disk management
 - Peripheral management
- * You would not get a mark for "multi-tasking" because the question asks for two **other** features* [2]

June 2013, Question 9

- 3 (a) Off-the-shelf software is available for anyone to buy and use.
Custom software is written for a specific customer, in this case the school. [2]
- (b) Proprietary software is distributed as an executable file, without source code.
Open source software is distributed with its source code.
or
Proprietary software cannot be altered.
Open source software can be modified (provided it remains open source). [2]

** When discussing a "difference" you need to state it from both sides. Not just state half of the answer and assume the difference is inferred*

Chapter 4: Data Representation in Computers

June 2013, Question 5

- 1 (a) (i) $64+32+8+1=105$
- (ii) $154=128+16+8+2$ so binary is 10011010 [2]
- (b) (i) 9 7 B [3]
- (ii) Because 4 bits are represented by 1 hex digit it is easy to convert...
...and much easier for engineers to remember and type in correctly.
Less likely to make a mistake. [2]

June 2012, Question 6

- 2 (a) 0011 0111 [2]
 (b) 37 (3 × 16 + 7) [2]

Jan 2012, Question 5

- 3 (a) [5]

	Must be included	Need not be included
The names of the people in the picture		✓
The width of the picture in pixels	✓	
The number of bits used for each pixel	✓	
The number of people in the picture		✓
The colour of each pixel	✓	

- (b) (i) The number of pixels in an image expressed as the-number-of-pixels-across × the-number-of-pixels-down e.g. 400 × 600. Effectively this describes the pixel density. [1]
 (ii) Higher resolution means more pixels are needed, ...which will increase the size of the image file. [2]

Jan 2013, Question 6

- 4 (i) The height of the sound wave is measured at regular intervals. The numbers are converted to binary and stored as a sound file. [2]
 (ii) If the interval is smaller then there will be more samples taken over the same period of time, ... so the file will be bigger ... but the sound will be stored more accurately. The reproduced sound will be better quality. [3]

* Make sure you mention the effects on **file size** and **quality**

Chapter 5: Databases

June 2011, Question 2

- 1 (a) DBMS = Database Management System
 Provides separation between applications and the data
 Allows you to:
- create related tables
 - input data with forms
 - query the data
 - get reports
- Provides security for the data.
- [3]
- * So much you could say here. Be concise but make sure you list more than three things*
- * If the question uses initials, such as DBMS, then always state what they stand for to start with!*
- (b) Forms:
- A screen used for data input and editing.
 User can use controls to enter data as well as type it in e.g. combo boxes and check boxes.
 Data entered is saved to the database.
 The grocery shop may have a form to enter new products.
- Reports:
- A snapshot in time of the data from a database for printing
 ...formatted on a page in a way that makes it information.
- May be grouped or sorted and have totals
 The grocery shop may print a weekly sales report.
- [6]
- * Don't forget to give the example from the shop database*
- (c) (i) Supplier = Killey's: 0003, 0006 [1]
 Price > £1.00 OR Supplier = Hill Farm: 0001, 0002, 0004, 0008 [3]
- * For the second part you get 1 mark if 0001, 0002, 0004 are all in answer, 1 mark if 0002 and 0008 are in answer and 1 mark if 0002 is not repeated and there are no extra answers*
- (ii) Discontinued = False AND QuantityLeft < ReorderLevel [3]

June 2012, Question 9

2 (a) Database Management System [1]

(b) * *BUMP the question to make sure you answer both halves:*
Describe the features of a DBMS that can be used to create customised data handling applications and explain why using a DBMS is desirable.

Features:

- Tools for creating the database structure e.g. creating tables, relationships, queries
- Provides application and data independence
- Can create forms to manage data input and data validation
- Can create queries to select certain data and reports to display it in a meaningful way
- Tools to control access to the data – access rights and security

Why desirable:

- The application provides a tailored view of the data for the user
- User doesn't have to be an expert in using databases
- Separating application and data means the data can be accessed separately by different applications for different people e.g. accounts department and sales staff will use it differently
- Security protects the data – only gives access to people who need it

* *Use headings so it is obvious that you have addressed the whole question*

* *Use bullet points to write less but say more - this is not an essay*

Jan 2013, Question 11

3. (a) A persistent, organised store of data. [2]

(b) Email address must contain an @ symbol
Gender must be "Male" or "Female"
Password must be at least 6 characters long

* *You must describe actual validation, not just name a type of validation such as "length check"* [3]

(c) A user can upload more than one picture.
If they were not in separate tables you would have to store the user details for every picture – data redundancy.

Separate tables means no data redundancy as user details are just stored once per user.
No redundancy minimises data inconsistency.

Tables are linked with a relationship between key fields – primary key in USER is UserID and will also be the foreign key in PICTURE. [4]

* *4 marks so BUMP the question to make sure you answer both halves. It would be easy to forget to say how the tables are linked*

Chapter 6: Communications and Networking

June 2011, Question 3

- 1 (a) (i) Hypertext Markup Language
 A programming language used to define the content and layout for a web page using tags.
 Tags used to show text paragraphs, images to be included and hyperlinks to other pages [2]
- (ii) Important to have a standard language for all web pages so they can be displayed in different browsers correctly. [2]
- (b) [5]

File	File Format
A high resolution image of the band to use as a desktop background.	JPG
Sheet music of their songs ready to be printed in the correct format for guitar players.	PDF
A short video extract from their latest concert tour.	MPEG
A compressed collection of 200 plain text files containing the lyrics of all their songs.	ZIP
An audio recording of a song from their album.	MP3

- (c) (i) Reduces the size of the file so...
 ...it reduces the time taken to download it. [2]
- (ii) Lossless: no data is lost in the compression process so the uncompressed file contains the same data as the original.
 eg: a text file where all data is vital
- Lossy: some non-essential data is removed in the compression process so the uncompressed file will contain less data than the original.
 eg: reducing the resolution of a picture that will be used as a thumbnail [4]

***** You must give examples to get all the marks

Jan 2012, Question 4

- 2 (a) * *Many people think the Internet and the World Wide Web are the same thing. In fact the Internet is the physical network and the Web is one of several services that use this network* [3]

	True	False
The internet is the same as the World Wide Web		✓
The internet is a Local Area Network		✓
The internet is a network between many networks	✓	

- (b) How DNS servers are used:

- You type a URL into the browser
- The local DNS server looks up the URL in its database
- DNS server returns the IP address for that URL or
- If it does not have that URL listed it forwards it to another DNS server to get the address
- In this way DNS servers are constantly being updated by other DNS servers
- The address of any URL can be found on a DNS server somewhere

Advantages:

- People do not need to remember IP addresses
- If the addresses change it is easy to upgrade – domain name stays the same
- By connecting to any DNS server you can have access to any address

[6]

* *Keep your response concise - use headings and bullet points to write less but say more*

June 2012, Question 2

- 3 (a) Bus [1]
- (b) Router [1]
- (c) File server: storing files that other workstations can access
 Email server: storing and managing emails for the users on the network
 Print server: making printers on the network into shared resources [3]
- (d) All devices have equal status on the network rather than specialised roles.
 There are no servers controlling the network.
 Computers can access files on other computers if access rights have been granted. [2]

Chapter 7: Programming

June 2013, Question 7

1 (a) Constant: PlayerKey Variable: Position or KeyPressed [2]

(b) Selection: Where the program will execute certain instructions based on a condition.
e.g. position is only incremented if KeyPressed = PlayerKey is true

Iteration: Where a program executes a group of instructions zero or more times based on a condition or for a fixed number of times.

eg: the program keeps processing the instructions in the Repeat loop until Position = 100 [4]

***** *You must give examples from the program given to get all the marks*

(c) Change "Position = Position + 1" to "Position = Position + random(4)" to add a random number between 1 and 4 to Position.

Change "until Position = 100" to "until Position >= 100" because Position is not going up by just 1 now, it may skip from 99 to 101 and never be exactly 100. [4]

June 2013, Question 10

2 INPUT Length1 [5]
INPUT Length2
INPUT Length3

```
IF Length1 = Length2
THEN
  Output "Isosceles"
ELSE
  IF Length1 = Length3 THEN
    Output "Isosceles"
  ELSE
    IF Length2 = Length 3 THEN
      Output "Isosceles"
    ELSE
      OUTPUT "Not Isosceles"
    END IF
  END IF
END IF
```

***** *Make sure you output the exact words you are told in the question!*

***** *A nested if statement is one approach here. You need to see if side 1 is the same as side 2. If not you check if side 1 is the same as side 3. If neither of these is true, perhaps side 2 equals side 3?*

***** *An alternative is one if statement with a more complex condition:
IF (Length1 = Length2) OR (Length1 = Length3) OR (Length2 = Length3) THEN...*

***** *Indent your constructs in algorithms. It makes it easier to read so you'll be less likely to miss an END IF*

***** *If the Quality of Written Communication is being assessed for an algorithm, meaningful identifiers and indenting will be necessary to get in the top mark band*

Jan 2013, Question 9

3 (a) The program will run but an error in the algorithm means that the program does not output the expected result. E.g. a loop doesn't execute the right number of times because the condition is coded incorrectly.

[2]

(b)

[6]

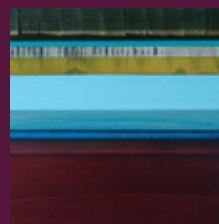
Input Data	Expected outcome	Reason for test
C	DEF	Checks that the next three letters are output correctly
A	BCD	Checks that it goes back to the start of the list
H	Error message	Checks that it outputs an error if the user inputs a letter that isn't in the list – an invalid note

acceptable use policy	102, 108	central processing unit	9, 29	digital	61, 107
access rights	100	centralised management	82	disaster recovery	3, 7, 101
actuator	1, 23	char	137	disk	
ADC. <i>See</i> analogue-to-digital converter		character set	54, 60	defragmentation	40
address	46	clients	86	defragmenter	46
IP	91	client-server	106	organisation utilities	40
MAC	90	network	86	DNS. <i>See</i> domain name system	
algorithm	112, 126, 135	clock speed	10, 29	DNS server	92
amplitude	57	colour depth	56, 61, 103	document file types	104
analogue	61, 107	command-line interface	37, 46	domain name system	92, 108
analogue-to-digital converter	57, 61	communication	82	dot-matrix printer	23
AND	28, 75	comparison operations	137	dual-core	11, 29
antivirus software	38, 46	compiler	62, 120, 121, 136	durability	14
application software	42, 45	compression	109	EBCDIC	54, 60
archiving	101	lossless	103	editing environment	132
arithmetic operations	137	lossy	103	embedded computer	2
array	125, 137	computer		encryption	100, 108
ASCII	54, 60	embedded	2	entity	70, 78
assembler	120, 136	system	1, 7	environmental considerations	6
assembly language	136	system reliability	2	ethical considerations	5
audio		condition	135	existence check	73
file types	104	constant	122, 136	failover	102, 108
auto-documentation	134	Copyright, Designs and Patents Act	4, 5	fault-tolerant	102, 108
automatic update	46	CPU. <i>See</i> central processing unit		features	14
availability	7	criteria	75	Federation Against Software Theft	4
backup	108	CRT. <i>See</i> cathode ray tube		fetch-execute cycle	10, 29
and restore,	101	CSS. <i>See</i> cascading style sheet		field	66, 78
barcode scanner,	20	customising	76	file	
benefits of networking,	82	custom-written software	44, 47	management	47
binary,	49, 60	data		transfer	47
logic	26, 28	duplication	67, 70, 78	transfer and management	40
bit	49, 60	inconsistency	67, 78	firewall	39, 46
bitmap	105	redundancy	70, 78	flash memory	17, 30
image	61, 103	separation	67	flat file	69, 78
boolean		views	68	flow diagram	113, 135
data type	26	Data Protection Act	5	flowchart	113, 135
expression	26, 30, 135, 137	data type	123, 136	foot-operated mouse	21
variables	26	boolean	123, 125	FOR loop	117
boot	13	char	123	foreign key	72, 78
bootstrap loader	13, 30	integer	123	form	72, 79
boundary data	137	real	123	format check	73
braille printer	23	string	123	formatting	47
broadband	94, 107	database		freeware	43, 47
bus	106	definition	65, 78	GHz. <i>See</i> gigahertz	
network	83	database management system	68, 72, 78	GIF. <i>See</i> graphic interchange format	
byte	49, 60	features	68	gigabyte	49, 60
cache	9, 29	DBMS. <i>See</i> Database Management System		gigahertz	10
camelCase	135	de facto standard	3, 4	graphic	
capacity	14	decimal	60	bitmap	61
cascading style sheet	96, 108	decision box	113	vector	56
CASE statement	115	defragmentation	40	graphic interchange format	105
cathode ray tube	22	denary	50, 60	graphical user interface	37, 46

Index

GUI. See graphical user interface		library		peer-to-peer	86
hardware	1, 7	programs	33	policies	101
HCI. See human-computer interface		software	45	ring	84
Health and Safety at Work Act	5	licence	47	security	99
Hertz	10	light-emitting diodes	23	star	84
hexadecimal	53, 60	local area network	82	topologies	83
high level language	62, 112, 119, 136	logic		wide area	88
HTML. See hypertext markup language		diagram	30	network interface card	85, 106
HTTP. See hypertext transfer protocol		error	127, 128, 137	nibble	49, 60
HTTPS	100, 107	gates	28	NIC. See network interface card	
hub	106	logical operators	30, 75, 79, 137	non-volatile memory	30
human-computer interface	45	lossless compression	103, 109	NOT	28, 75
Hungarian notation	135	lossy compression	103, 109	object code	120, 136
hypertext markup language	96, 108	MAC address	90, 107	OCR. See optical character recognition	
hypertext transfer protocol	92, 107	machine code	62, 119, 136	off-the-shelf software	43, 47
IDE. See Integrated Development Environment		magnetic media	15, 30	OMR. See optical mark recognition	
identifier	71, 136	main memory	29	opcode	58, 59, 62
IF...THEN...ELSE	115	maintenance utilities	41	open source	4, 47
image		man-machine interface	45	software	43
file types	103	mean time between failure	2, 7	operand	59, 62
immediate access store	13	megabyte	49, 60	operating system	45
impact printer	23	megahertz	10	operation	137
imperative language	115, 136	memory	12	arithmetic	124
inkjet printer	22	flash	17, 30	comparison	124
input device	1, 7	management	34, 45	optical	
instruction set	58, 62	non-volatile	30	character recognition	20
integer	136	primary	13	mark recognition	20
Integrated Development Environment	131	secondary	13, 30	storage media	16, 30
interface		solid-state	17, 30	OR	28, 75
command-line	37	technologies	18	output device	1, 7
graphical user	37	virtual	13, 30	overflow	60
user	37	volatile	30	packets	90, 93, 107
Internet	91, 107	message	107	PascalCase	118, 135
Protocol	91	metadata	56, 61	password protection	100
interpreter	120, 121, 136	MHz. See megahertz		PDF. See Portable Document Format	
invalid data	137	microphone	21	peer-to-peer	106
IP. See Internet Protocol		MMI	46	network	86, 87
address	91, 107	mnemonics	119, 136	pen drive	30
iteration	117, 135	modem	94, 107	peripheral	35, 45
joystick	20	module	76	management	35, 45
JPEG	103, 105	monitor	22	persistent storage	65, 78
keyboard	19	mouse	20	pixel	56, 61
specialised	21	MP3	104, 105	PNG. See portable network graphic	
kilobyte	49, 60	MPEG	105	point of sale (POS)	24
LAN. See local area network	106	MTBF. See mean time between failure		portability	14
laser printer	22	multi-core processor	11	Portable Document Format	104, 105
LEDs. See light-emitting diodes		multi-tasking	35, 45	portable network graphic	105
legal considerations	5	network		presence check	73
length check	73	bus	83	primary key	71, 78
		client-server	86	primary memory	13
		local area	82		

printer		sensors	21	translator	33, 45, 120, 133, 136
braille	23	temperature	21	truth table	27, 30
dot-matrix	23	sequence	115, 135	type check	73
impact	23	server	85	Unicode	54, 60
laser	22	sharing resources	82	user	
process	46, 113	sip-and-puff switches	21	access levels	108
processor	29	software	7, 33, 45	interface	37, 45
professional standards		antivirus	38	utilities	33
in documentation	4	application	42	utility programs	38, 42, 45
in system development	3	applications	33	valid data	137
program-data independence	68, 78	custom-written	44	validation	76, 79
programming languages		freeware	43	checks	72, 73
first generation	119	off-the-shelf	43	variable	122, 136
second generation	119	open source	43	VBA	76
third generation	119	proprietary	43	vector graphic	56, 61
proprietary software	43, 47	system	33	verification	73, 79
protocol	89, 107	solid-state memory	17, 30	video	
pseudocode	114, 118, 135	sound		file types	104
QBE. <i>See</i> Query By Example		quality	57	views	68, 78
quad-core	11, 29	synthesis	57	virtual memory	13, 30
query	74, 79	waves	57	virus	38, 46
Query By Example	74	source code	120, 136	Visual Basic for Applications	76
quicktime	105	speakers	22	volatile memory	13, 30
RAM. <i>See</i> random access memory		spyware	46	WAN. <i>See</i> wide area network	
random access memory	12, 30	protection	39	waterfall model	4
range check	73	star network	84, 106	web servers	91
Rapid Application Development	4	storage	1	WHILE loop	117
read only memory	13, 30	device	7	wide area network	88, 106
real	136	media capacity	18	WIMP	37, 46
record	66, 78	string	137	Windows Media Player	105
redundancy	3, 7, 70	structured English	135	wireless access point	106
redundant	102, 108	switch	85, 106	WWW	107
related tables	71	syntax	137		
relational database	70, 78	syntax error	127, 128, 137		
relationship	70, 71, 79	system			
reliability	7, 14	availability	2		
measuring	2	buses	29		
REPEAT loop	117	cleanup	46		
report	75	cleanup tools	42		
resolution	56, 61	diagnosis	46		
ring network	84, 106	software	33, 45		
ROM. <i>See</i> read only memory		table	70, 78		
router	91, 94, 107	tags	108		
run-time environment	132	terabyte	49, 60		
sample		terminator	113		
interval	57, 61	test plan	129		
rate	61	testing	129		
resolution	57, 61	TFT	22		
secondary memory	13, 30	thermistor	1, 21		
security	36, 45, 99	topology	106		
selection	115, 135	touch screen	20		



Cover picture © 'Starting point' 2007
Oil on aluminium 51 x 51 cm
By Katherine Palmers-Needham

GCSE

Computing (OCR)

Computer Systems and Programming

3rd edition

This popular textbook, now in its third edition, has been redesigned and brought up to date with new content and an index for easy reference. Aimed at GCSE students, it provides comprehensive yet concise coverage of all the topics covered in *Unit A451: Computer Systems and Programming* of the OCR GCSE Computing Specification J275, written and presented in a way that is accessible to teenagers.

It will be invaluable both as a course text and as a revision guide for students nearing the end of their course. It is divided into seven chapters corresponding to the seven sections of the specification, each ending with a "Glossary of terms" and exam questions from past OCR GCSE papers. Answers to all questions, with hints and tips on how to tackle questions, are given at the end of the book.

The book is also available as a downloadable PDF, sold as a lifetime site licence to schools. This PDF may be loaded onto the school's private network or VLE.

To accompany this textbook, PG Online also publishes a series of seven downloadable teaching units. Each lesson in a unit consists of a PowerPoint presentation, teacher's notes, worksheets to be completed during the lesson and homework sheets with exam-style questions. These units are sold as a lifetime site licence and may be loaded onto the school's private network or VLE.

For more information on the PDF version of the book and the teaching units, please visit www.pgonline.co.uk



PG ONLINE

www.pgonline.co.uk

