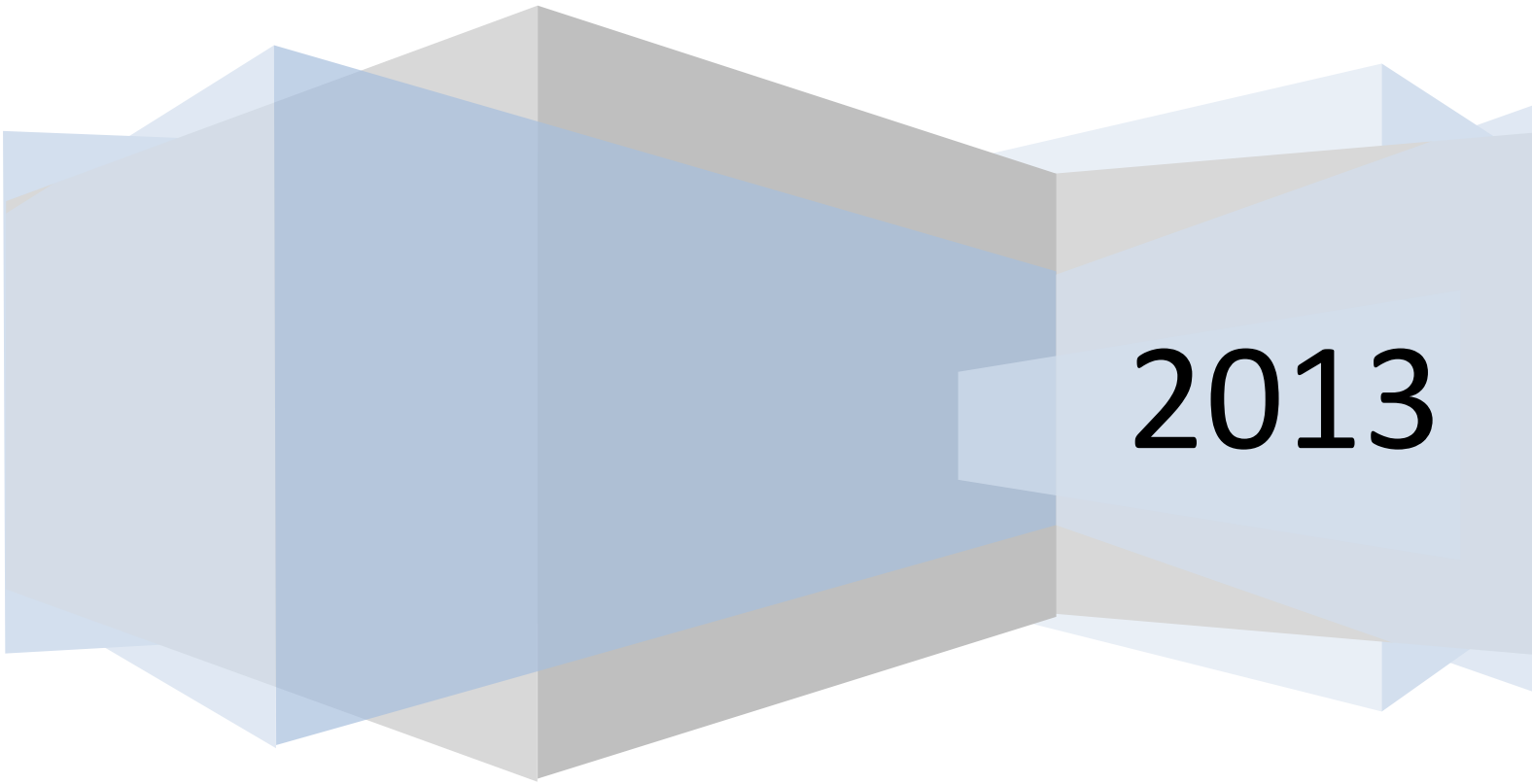
Revised: November 2013

GCSE Computer Science



Geraint D. Jones

Mark D. Thomas

Disclaimer

This resource is provided to support the teaching and learning of GCSE Computer Science. The materials provide an introduction to the main concepts of the theory of the subject and should be used in conjunction with other resources and sound classroom teaching.

It is intended that the resource will be updated periodically due to the evolving nature of the subject. Suggestions as to how the materials may be developed further are welcome through our feedback system or by sending an e-mail to: resources @wjec.co.uk

Contents

Computer systems 4

Data representation 20

Computer system 35

Networks 43

The Internet and communications 53

Algorithms 62

Programming 63

Ethical, social, and legal aspects 70

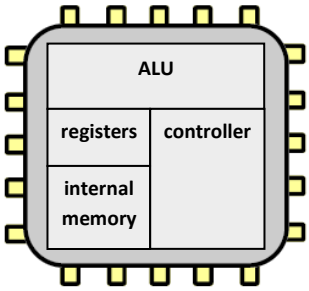
I. Computer systems

CAN YOU IDENTIFY AND DESCRIBE COMPUTER SYSTEMS?

You should be able to:

- identify and describe computer systems:
  - CPU
  - input devices
  - output devices
  - backing storage
  - data bus
  - address bus
  - ports and connectivity

CPU



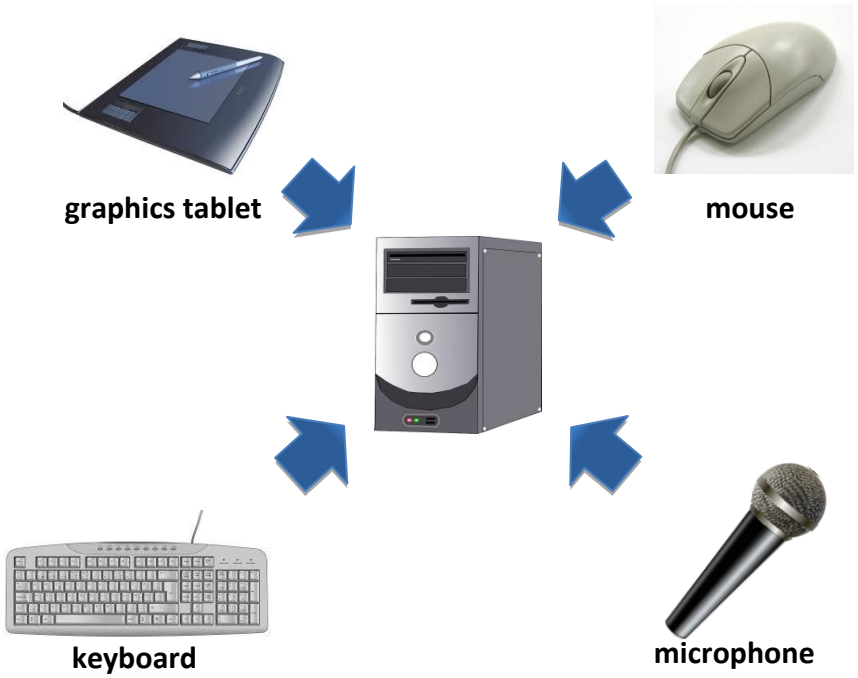
The **CPU** (**Central Processing Unit**) is the main component in a computer for processing data and instructions. It could be considered as the computing equivalent of the human brain. It is a hardware device that is made up of many sub components:

- controller
- ALU: arithmetic/logic unit
- registers
- internal memory
- buses

All of these components have their own specific function. These are discussed in further detail on page 7.

Input devices

An input device allows data, such as text, images, video or sound, to be entered into a computer system. Some common hardware input devices are shown below:



Output devices

There are many outputs created by a computer system. These include printed document, on-screen data, sound. An output device allows data to be communicated by the computer in a human-friendly form, for example, sound being output by a speaker. Some common hardware output devices are shown below:

Backing storage

This is where data is stored when it is not being actively used, usually for retrieval at a later date. These are discussed in further detail on pages 15 and 16.

Address bus and data bus

When data is saved or loaded from memory, the address at which it is to be stored or loaded from must be sent. The storage address of data always travels along an **address bus**.

Data will then need to be moved between several parts of a computer. The path along which data travels is called a **data bus**.

Ports and connectivity

Hardware ports

In computer **hardware**, a port serves as an interface between the computer and other computers or devices. Physically, it is a piece of equipment to which a plug or cable connects. An example of different hardware ports can be seen below.

HDMI **USB** **USB** **audio in** **audio out**

Software ports

Software ports are also used in computer systems. These are non-physical ports and are completely different from hardware ports. They allow multiple software applications to use different software ports on the same physical connection. Software ports are discussed in further detail on page 53.

INTERESTING FACT

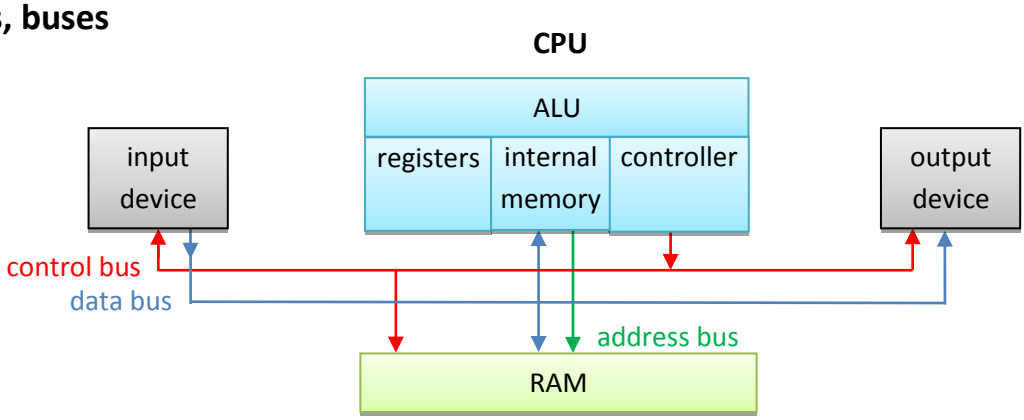
In the TCP/IP protocol, the ports are numbered 1 to 65535. Commonly used ports include port 80, for HTTP web access and port 25 for SMTP sending email.

UNDERSTANDING THE CENTRAL PROCESSING UNIT (CPU)

You should be able to:

- demonstrate an understanding of the Central Processing Unit (CPU), including:
  - controller
  - arithmetic and logic unit (ALU)
  - internal memory
  - register
  - buses

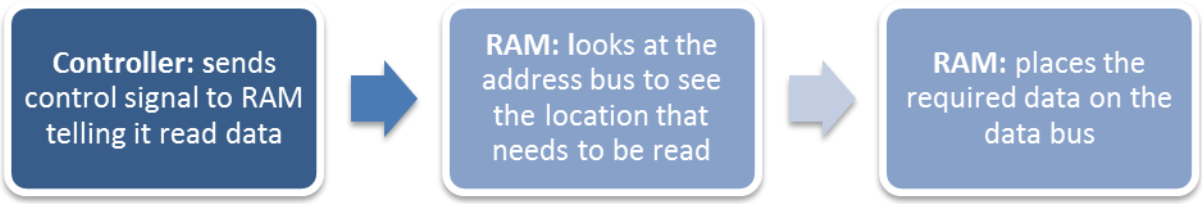
Components of the CPU: controller, arithmetic and logic unit (ALU), internal memory, registers, buses



Controller

The controller sends and receives signals from all parts of the computer. This ensures that all processes take place at the right time and in the correct order. These signals travel along a **control bus**.

Here is an example of a controller in operation:



Arithmetic and logic unit (ALU)

The ALU is the part of the CPU that processes and manipulates data. It performs simple calculations on the data that is temporarily stored in the registers. Examples of calculations that an ALU might perform are addition and subtraction.

The ALU is also able to perform comparisons on data. It is these comparisons that allow programs to make use of *choice* – e.g. in a high-level language an IF statement.

Registers

A register is a storage location found on the CPU where data or control information is temporarily stored. Registers are usually much faster to access than internal memory, since they have to be accessed so often.

KEY DEFINITIONS

Controller: manages the execution of instructions
ALU: processes and manipulates data
Register: fast access temporary data store
Internal memory: fast access memory on the CPU
Bus: connects different parts of the computer

An accumulator is a common example of a register. This is the register used by the ALU to store the results of its calculations.

Internal memory

Internal memory (*sometimes called level 1 cache memory*) is fast access temporary storage on the CPU. Data is moved from the registers to the internal memory when it is not being actively used. Data from internal memory can then either be written to RAM or called back into the registers for further processing. This process of using internal memory speeds up the processing of data.

Buses

Buses allow data to be transferred to different parts of the computer. There are three main buses used by the CPU. Two of these, the address bus and the data bus, are discussed on page 4. The third bus is called the **control bus**.

The control bus is used by the controller to send control signals to different parts of the computer.

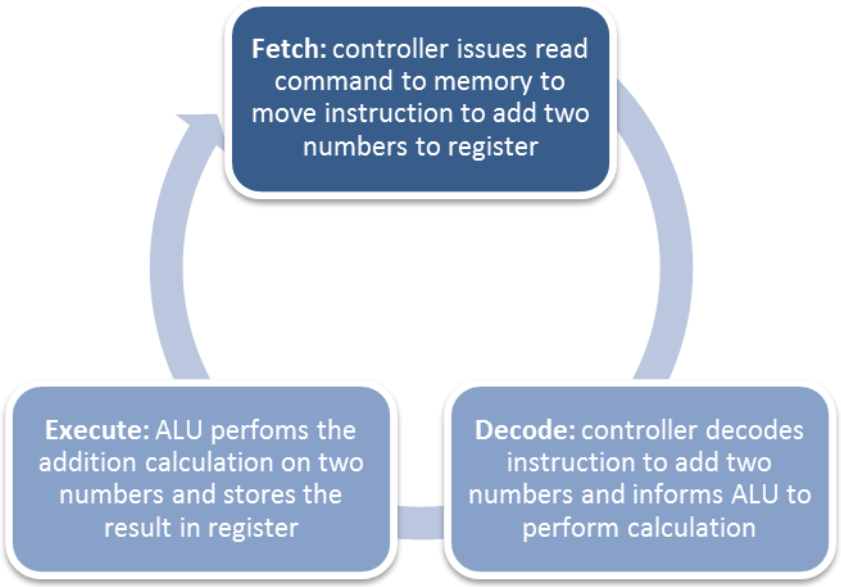
Explaining the role of the CPU in fetching and executing instructions stored in memory

The fetch-decode-execute cycle

There are **three** steps to processing instructions given by a currently running program:

1. The next instruction is **fetched** from memory into the control unit.
2. This instruction is then **decoded** to determine the action that needs to be carried out.
3. The instruction is **executed.**

Here is a simplified example of the fetch-decode-execute cycle, where the instruction is to add two numbers:



HOW IS PERFORMANCE AFFECTED BY FUNCTIONS?

You should be able to:

- explain how performance is affected by functions, including:
  - size of cache
  - speed of clock
  - number of cores
  - types of processors

Size of cache, speed of clock, number of cores, types of processors

Size of cache

Cache memory is a fast access type of memory that is very expensive. Due to its cost, only small amounts of cache memory are present in most computer systems. Cache memory improves the performance of the CPU as it is able to provide instructions and data to the CPU at a much faster rate than other system memory such as RAM. The more cache memory your system has, the better its performance is likely to be.

Speed of clock

The speed at which a processor operates is called the clock speed. The faster the clock speed, the faster the computer is able to run the fetch-decode-execute cycle and therefore process more instructions.

The speed of the processor is measured in Hertz (Hz). One clock tick per second would be measured as 1 Hz. Therefore a processor that operates at 1,000 clock ticks per second would be a 1,000 Hz processor, also known as a 1 kHz processor.

At this stage, it is a good time to introduce *prefix multipliers* for clock speeds:

Prefix	Symbol	Multiplier	Power of 10
Kilo	k	1,000	10^3
Mega	M	1,000,000	10^6
Giga	G	1,000,000,000	10^9

A typical modern day home computer would have a 2.5 GHz processor. This means the clock speed of the processor runs at 2,500,000,000 Hz or clock ticks per second.

INTERESTING FACT

Most CPUs are cooled using a fan to blow air over their surface. Liquid cooling systems have been found to be more effective at cooling CPUs, although the water used in these systems conducts electricity and can therefore be dangerous. A computer designer, Seymour Cray, designed a computer cooling system that used artificial human blood to cool the CPU, as it does not conduct electricity.

The clock speed inside the CPU can sometimes be changed. A processor can be set to run faster than its original design. By doing this however, it uses more energy and produces more heat. If this heat is not removed through cooling, the CPU can overheat, which will damage the CPU and shorten its lifespan. This is called *overclocking*.

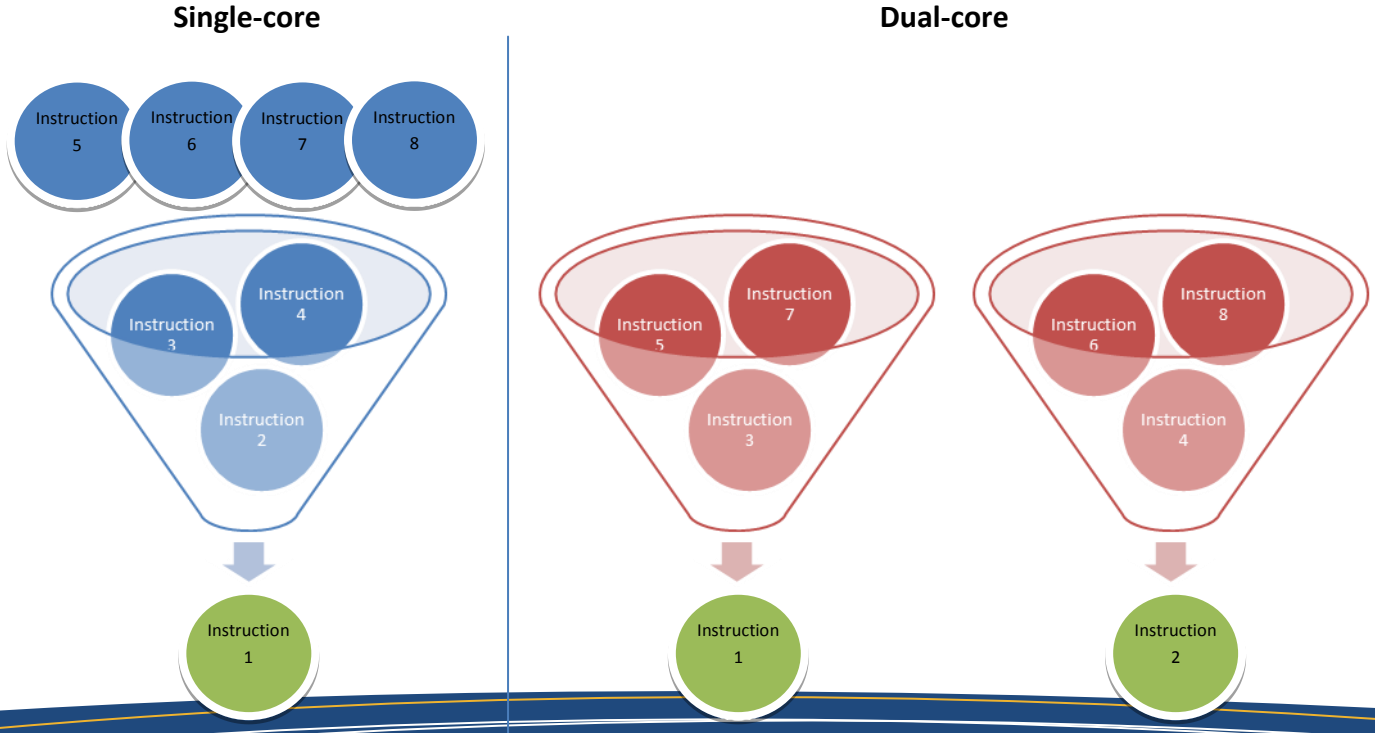
Some computer systems, especially mobile devices, set the clock speed of the CPU lower than its original design. This results in less power consumption, less heat being produced and will therefore increase the battery life of the device. This is called *underclocking*.

Some devices are able to change their own clock speed dynamically. For example, when your computer is idle, the clock speed may be set at a lower rate than if you were running a CPU intensive program, such as a computer game.

Number of cores

We assumed on page 7 that each CPU has only one core. However, this isn't always the case, as some CPU's have multiple cores. You may be familiar with the terms 'dual-core' and 'quad-core', so what exactly do these terms mean?

A core is the term used to describe the processing components within the CPU. Multi-core processors therefore have many processing components within the same CPU. Below is a diagram that illustrates a number of instructions waiting to be processed in single-core and dual-core CPUs.



In a single-core CPU each instruction is processed one after the other, whereas in a dual-core CPU, two instructions may be processed at the same time. In theory,a dual-core CPU should be able to process instructions twice as fast as a single-core CPU. However, this isn't always the case as sometimes *Instruction 2* may need the result of *Instruction 1* before it can be processed.

On the whole though, a computer running many programs at the same time will run faster on a multi-core processor than on a single-core processor.

INTERESTING FACT

Many high-end gaming consoles include CPUs with multiple cores. The Sony *Playstation 3* has an 8 core CPU.

Types of processors

There are two main types of processor, namely **Reduced Instruction Set Computer (RISC)** and **Complex Instruction Set Computer (CISC)**.

RISC processors can process a limited number of relatively simple instructions. To carry out more complex commands the problem is broken down into a longer list of simpler instructions. The advantage of this is that a RISC processor is able to process these simpler instructions quickly. Processing simpler instructions also requires less circuitry to decode and execute these instructions, which in turn means less power consumption and therefore less heat being generated.

CISC processors can process a large number of complex instructions. This allows the processor to understand and carry out complex tasks with only a few instructions. The advantage of this is that a CISC processor is able to process complex instructions, without having to break them down into many simpler instructions. Processing complex instructions however requires more circuitry to decode and execute these instructions, which in turn means more power consumption and therefore more heat being generated.

HOW CAN WE DESCRIBE THE FUNCTIONS OF DIFFERENT TYPES OF MEMORY?

You should be able to:

- describe the functions of different types of memory:
  - Random Access Memory (RAM)
  - Read Only Memory (ROM)
  - flash memory
  - cache memory

Random Access Memory (RAM)

RAM is used for the temporary storage of currently running programs and data. It consists of a large number of **store locations** each of which is identified by a unique **address**. The data in each store location can be changed.

RAM is **volatile** – data is lost when the power is switched off.

Example: When you are working on a word-processed document, the program you are using and the data within the document are both stored in RAM.

Read-only Memory (ROM)

ROM is used for the permanent storage of data. The data in each store location cannot be changed.
ROM is **permanent** –data is not lost when the power is switched off.

Example: ROM can be used for storing the programs such as the **BIOS**. The disadvantage of using ROM to store the BIOS is that it cannot be upgraded.

KEY INFORMATION

Basic Input Output System (BIOS): A low-level program that handles input and output operations relating to the keyboard and screen of the system. It provides an interface between the hardware and the operating system. One of its primary functions is loading and executing the *bootstrap loader* – the program that loads the operating system.

Flash memory

Flash memory is used for the permanent storage of data. However, the data stored in flash memory can be changed.

Flash memory is **permanent** – data is not lost when the power is switched off.

Example: Flash memory can be used for storing the programs such as the **BIOS**, which is advantageous as the BIOS can be upgraded.

Cache memory

Cache memory is used for the temporary storage of frequently accessed data and instructions. It consists of a small number of **store locations** that can be accessed very quickly by the CPU; it is quicker than RAM.

Cache memory is **volatile** – this means that data is lost when the power is switched off.

Example: Cache memory often stores the next set of instructions that need to be processed by the CPU as it is an exceedingly fast memory and is located on the processor.

Summary of different types of memory

	Permanent	Volatile	Data can be changed	Speed
Cache memory		✓	✓	★★★★
ROM	✓			★★★
RAM		✓	✓	★★
Flash memory	✓		✓	★

DO YOU KNOW THE FUNCTIONAL CHARACTERISTICS OF SECONDARY STORAGE TECHNOLOGIES?

You should be able to:

- describe the functional characteristics of secondary storage technologies, including:
  - functional characteristics: suitability, capacity, durability, portability, speed
  - technologies such as: optical, magnetic, solid state, storage in the cloud

Functional characteristics: suitability, capacity, durability, portability, speed

Secondary storage is also known as backing storage.

Data is written from memory to secondary storage when data is no longer being actively used, for retrieval at a later time.

INTERESTING FACT

The first commercial hard disk drives had the capacity to store approximately 5 MB and were the size of a dining room table. They were also called a *Winchester Drive*.

The time a computer takes to access data stored on secondary storage is **longer** than the time taken accessing data from memory.

The most frequently used backing storage media are:

Media	Suitability	Typical capacity	Durability	Portability	Speed
 Flash drive	Moving relatively small files from work to home	2 GB – 64 GB	★★★★	✓	★★★★
 External hard drive	Backing up a home computer system	500 MB – 4 TB	★	✓	★★★
 CD/DVD/Blu-ray disk	Storing multimedia files	650 MB (CD) 9 GB (DVD) 50 GB (Blu-ray)	★★★	✓	★★
 Magnetic tape	Backing large commercial servers on multiple tapes	200 GB – 400 GB	★★	✓	★

Technologies such as: optical, magnetic, solid state, storage in the cloud

Optical

Optical storage media uses technology such as lasers. Laser beams are projected onto a CD/DVD or Blu-ray disk and if light is reflected back, then data is read as a 1 and if light is not reflected back, data is read as a 0. Lasers can be used to read and write information to a disk.

> **KEY INFORMATION**
> **BInary digiT (BIT):** In computer systems, data is represented by either a 1 or a 0.
> See page 21 for more information.

Magnetic

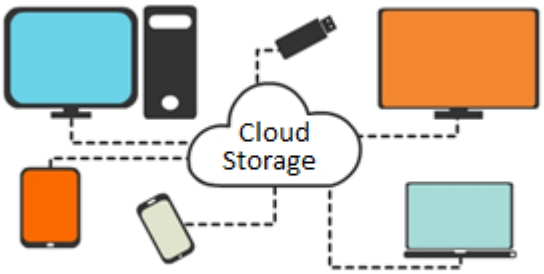
This technology is used in floppy disks, hard disks and tapes. Data is stored on a magnetic medium, which can be a disk or a tape, by writing data using a write-head. Data can then be read by the read-head.

Solid state

Solid state technology is used in storage media such as USB flash memory sticks. The technology is called solid state as it does not have any moving parts, such as a read-head in magnetic storage. Solid state storage technology is increasingly used to replace both magnetic and optical storage, especially in mobile devices, where its low power consumption and high speed access is advantageous.

Storage in the cloud

Storage in the cloud is a contemporary data storage facility that allows users to store their data on third-party servers. They can then access that data from many computing devices.



There are many **advantages** to this, such as maintenance tasks, backup and data replication. Purchasing additional storage devices becomes the responsibility of the cloud storage service provider.

A **disadvantage** of storage in the cloud is that an Internet connection is required. Some other disadvantages include the concern for some organisations that personal data held on a third-party server could in fact be physically stored in a country where adequate data protection legislation does not exist. Another disadvantage is that users are solely reliant on the cloud storage provider when it comes to ensuring that their data is stored safely and can be retrieved at a later date.

DO YOU UNDERSTAND HUMAN-COMPUTER INTERACTION?

You should be able to:

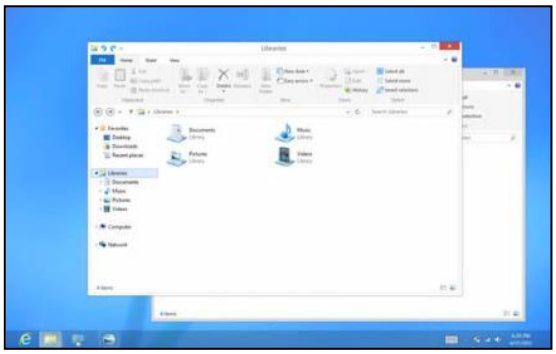
- demonstrate an understanding of human-computer interaction

Human-computer interaction (HCI) is the term used to describe the communication between people and computer systems. To allow a person and a computer system to communicate, an interface is required, often called a **human-computer interface**.

Different interfaces are provided by the operating system and can be identified by the style of communication they use. Some are entirely text-based whereas others use images to represent different commands.

Graphical User Interface (GUI)

A GUI is a type of interface that allows users to interact with a computer system through graphical icons.



GUIs were introduced to help users, especially beginners, use computer systems as Command Line Interfaces (CLIs) were found to be difficult.

There are many different **features** of a graphical user interface. These include:

- windows
- icons
- menus
- pointers
- assistants/help files/tutorials
- favourite settings/change environment/customisation
- shortcuts/hot keys
- task bar/ribbon bar/tabs/customised toolbar

Here are some of the benefits and drawbacks of a graphical user interface:

Benefits	Drawbacks
• Intuitive • Easy to navigate • Uses Windows, Icons, Menus and Pointers • No complicated commands • Data between different software applications is	• Requires a large amount of memory • Is relatively processor intensive • Computing experts may find a GUI slower than a command line interface • GUIs take up a much larger amount of hard disk

easily exchanged	space than other interfaces

Menu driven

This type of interface allows people to interact with a computer system by presenting the user and allowing them to work through a series of menus. The *iPod Classic* is a perfect example of a device that uses a menu driven interface as users are presented with a menu that contains a list of artists. Having chosen an artist, another menu appears with a list of albums belonging to that artist. Following this, another menu is presented with a list of songs belonging to the chosen album.

Here are some of the benefits and drawbacks of a menu driven interface:

Benefits	Drawbacks
• No need to learn a lot of commands • Intuitive/easy to understand • Easy to navigate • Ideal for beginners – everything is in a logical place/order • No need of expert language to learn • Little processing power needed	• Irritating if there are too many menu screens to work through – users get annoyed or bored if it takes too long • Navigating can be a long process

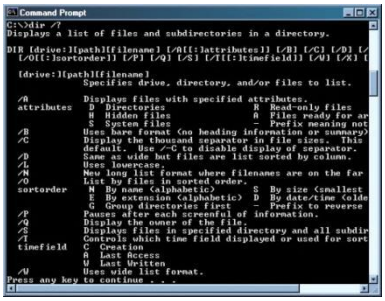
Voice-driven

Voice driven interfaces, also called voice recognition, can be used to issue commands to a computer system and enter data into it. Voice-driven interface is a popular interface as it is natural for people to communicate in this way.

Here are some of the benefits and drawbacks of a voice-driven interface:

Benefits	Drawbacks
• Speech input is much faster than keyboard input • No need to learn to type • Less danger of RSI • Reduces typing mistakes such as spelling/hitting wrong key • Keyboard takes up room on the desk • Users with a disability that prevents typing can use speech input • Hands-free advantages – can multitask • Users find talking more natural than typing.	• Background noise interferes with speech recognition • User when they have a speech impediment, sore throat, cold or a strong accent will not be understood • Users with a disability that prevents speech would need to find a different method for input • Difficult to keep data input private as people can hear what you are saying • Words that sound the same, such as 'too' and 'two' may not be recognised

Command Line Interface (CLI)



A Command Line Interface is an entirely text-based interface that allows a user to communicate with a computer system by typing in commands. However, computer systems will only execute specific commands that are predefined.
Before GUIs were developed, command line interfaces were the most widely used interface.

Here are some of the benefits and drawbacks of a command line interface:

Benefits	Drawbacks
• Quicker to type commands • Quicker to input commands as shortcut keys can be used • Little memory and processing power needed compared with other interfaces • Little storage space is required (no graphical images to store) • Experts who have memorised the commands find it very fast to use	• Very confusing for someone who has never used a command line interface • Commands have to be typed precisely. If there is a spelling error the command will fail • A large number of commands need to be learned • Instructions cannot be guessed • Not suitable for a novice

Touch Sensitive Interface

Touch sensitive interfaces are becoming more popular and are extensively used in mobile computing devices. Commands are issued or data is input by touching the screen with your finger or a stylus pen. As well as tapping the touch sensitive screen, the screen can interpret other actions made by the user, such as pinching and swiping.

Here are some of the benefits and drawbacks of a touch sensitive interface:

Benefits	Drawbacks
• Very intuitive • Easier to use as the user simply touches what is seen on the display • No keyboard or mouse is required • Touching a visual display of choices requires little thinking and is a form of direct manipulation that is easy to learn • Easier hand-eye coordination than mice or keyboards	• Screen can be easily damaged/scratched • Dirty screens are difficult to read • Users must be within arm's reach of the display • It is difficult to select small items • User's hand may obscure the screen • Screens need to be installed at a lower position and tilted to reduce arm fatigue • Some reduction in image brightness may occur

II. Data representation

DATA REPRESENTATION SYSTEMS – CAN YOU RECOGNISE AND EXPLAIN THESE?

You should be able to:

- identify and explain data representation systems:
  - the nature of data
  - why data needs to be converted into binary format
    - binary numbers representing characters
    - hexadecimal numbers representing binary numbers
    - the terms 'character set', Unicode and American Standard Code for Information Interchange (ASCII)
    - truth tables and logical operations: AND, OR

The nature of data

Data is made up of raw facts and figures and can be represented in many different forms including text, numbers, pictures, sounds, video clips. Information can be derived from data when it is processed.

Why data needs to be converted into binary format

You will need to be familiar with three different counting systems. These are denary, binary and hexadecimal.

Denary

The first counting system that you need to be familiar with is the **denary** counting system, also known as the Base **10** or decimal counting system. In the denary counting system, the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 are used to represent numbers. The number *138* for example, actually means 1 'hundred', 3 'tens' and 8 'units'. This gives the total one hundred and thirty-eight:

10^2	10^1	10^0
100	10	1
1	3	8

Binary

The second counting system that you need to be familiar with is the **binary** counting systems, also known as the Base **2** counting system. In order for data to be processed by a computer system, it must be converted into binary format. This is because computer systems can only store and process **B**inary dig**IT**s, also known as BITs. A BIT is either a 1 or 0. You may think of this as a light switch, where the switch is either **ON** or **OFF**:

- If the switch is ON it is stored as the digit 1.
- If the switch is OFF it is stored as the digit 0.

A binary number is a string of BITs, for example 10001010.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
1	0	0	0	1	0	1	0

The binary number 10001010 is therefore a binary representation of the denary number 138 (128 + 8 + 2). For more information see pages 27–8.

Binary numbers representing characters

A **character** can be a letter, digit, space, punctuation mark or various other symbols. When characters are stored on a computer system, they are stored as a binary number.

It is important that computer systems recognise that characters can be represented differently by other computer systems; otherwise data could not be exchanged between computers.

The terms 'character set', Unicode and American Standard Code for Information Interchange (ASCII)

In order to allow for data exchange between computer systems, **character sets** were devised. A character set is a table that maps a character with a unique binary number.

One such character set is the 7-bit American Standard Code for Information Interchange (ASCII). Part of the ASCII character set, that includes printable characters only, can be seen in the table overleaf.

INTERESTING FACT

Before the widespread adoption of graphical user interfaces, programmers used the ASCII character set to design simple interfaces. Try searching for some on the Internet.

Denary	Binary	Hex	Character
32	100000	20	*space*
33	100001	21	!
34	100010	22	"
35	100011	23	#
36	100100	24	$
37	100101	25	%
38	100110	26	&
39	100111	27	'
40	101000	28	(
41	101001	29	)
42	101010	2A	*
43	101011	2B	+
44	101100	2C	,
45	101101	2D	-
46	101110	2E	.
47	101111	2F	/
48	110000	30	0
49	110001	31	1
50	110010	32	2
51	110011	33	3
52	110100	34	4
53	110101	35	5
54	110110	36	6
55	110111	37	7
56	111000	38	8
57	111001	39	9
58	111010	3A	:
59	111011	3B	;
60	111100	3C	<
61	111101	3D	=
62	111110	3E	>
63	111111	3F	?

Denary	Binary	Hex	Character
64	1000000	40	@
65	1000001	41	A
66	1000010	42	B
67	1000011	43	C
68	1000100	44	D
69	1000101	45	E
70	1000110	46	F
71	1000111	47	G
72	1001000	48	H
73	1001001	49	I
74	1001010	4A	J
75	1001011	4B	K
76	1001100	4C	L
77	1001101	4D	M
78	1001110	4E	N
79	1001111	4F	O
80	1010000	50	P
81	1010001	51	Q
82	1010010	52	R
83	1010011	53	S
84	1010100	54	T
85	1010101	55	U
86	1010110	56	V
87	1010111	57	W
88	1011000	58	X
89	1011001	59	Y
90	1011010	5A	Z
91	1011011	5B	[
92	1011100	5C	\
93	1011101	5D	]
94	1011110	5E	^
95	1011111	5F	_

Denary	Binary	Hex	Character
96	1100000	60	`
97	1100001	61	a
98	1100010	62	b
99	1100011	63	c
100	1100100	64	d
101	1100101	65	e
102	1100110	66	f
103	1100111	67	g
104	1101000	68	h
105	1101001	69	i
106	1101010	6A	j
107	1101011	6B	k
108	1101100	6C	l
109	1101101	6D	m
110	1101110	6E	n
111	1101111	6F	o
112	1110000	70	p
113	1110001	71	q
114	1110010	72	r
115	1110011	73	s
116	1110100	74	t
117	1110101	75	u
118	1110110	76	v
119	1110111	77	w
120	1111000	78	x
121	1111001	79	y
122	1111010	7A	z
123	1111011	7B	{
124	1111100	7C	|
125	1111101	7D	}
126	1111110	7E	~

Using the ASCII character set, the character *A* would be stored as the binary number 1000001.

The problem with using this ASCII character set is that it is only able to represent 128 different characters and computer systems need to be able to store more characters than this. For example, you may have noticed that the *£* character is missing from the table above. As a result, other character sets were developed and used to allow computer systems to store more characters.

Unicode is a standard character set that has combined and replaced many others. It was originally an extension to the ASCII character set and it contains many of the characters used around the world.

Hexadecimal numbers representing binary numbers

The third counting system that you need to be familiar with is the **hexadecimal** counting system, also known as the Base **16** counting system. In the hexadecimal counting system, the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 are used to represent 1–9 and then the characters A, B, C, D, E and F are used to represent 10 –15. The hexadecimal number *8A* for example:

16^2	16^1	16^0
256	16	1
0	8	A

The hexadecimal number 8A therefore represents 8 'sixteens' and 10 'units'. This gives the total one hundred and thirty-eight. Remember that A = 10, B = 11, C = 12, D = 13, E = 14, F = 15.

INTERESTING FACT

Up until the late 20th century, traditional Chinese weights and measurements used in the marketplace used the hexadecimal counting system. Other cultures used different base counting systems, e.g. the ancient Babylonians used a Base **60** counting system.

Hexadecimal is widely used as binary numbers can be quickly converted into hexadecimal numbers which are more convenient for humans to use. For example, a telephone conversation where you might read out the binary number 10001010 could cause confusion. It would be easier to simply say 8A and mistakes are less likely to be made.

See pages 27–31 for conversion between binary, denary and hexadecimal numbers.
Hexadecimal notation is used in MAC addresses. For further details see page 57.

Truth tables and logical operations: AND, OR, NOT

Logical operators are used in programming. It is not unusual to find a code such as:

- IF A = 1 **AND** B = 1 THEN
- IF A = 1 **OR** B = 1 THEN

A truth table for a logical operator defines the outputs for different combinations of input.

NOT

The NOT logical operator has only one input and one output. The output is the opposite of the input.

Input (A)	Output (NOT A)
0	**1**
1	**0**

AND

The AND logical operator has two inputs and one output. The output is 1 only if A and B are both 1.

Input (A)	Input (B)	Output (A AND B)
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

OR

The OR logical operator has two inputs and one output. The output is 1 if either A or B is 1.

Input (A)	Input (B)	Output (A OR B)
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

There are other logical operators, such as XOR. This will be covered on page 50 in encryption techniques.

Logical operations

Logical operations can be used in control systems. For example, a control system that is required to close the windows on a commercial greenhouse when at least one of the following conditions is true:

- the wind speed rises above 12 km per hour.
- it is raining.

would use the logical operator **OR**.

A control system that is required to turn on a sprinkler system in a field when both of the following conditions are true:

- the temperature rises above 25° Celsius
- it has not rained in the last five days

would use the logical operator **AND**.

Data types such as: integer, real, Boolean, character, string

Many different data types can be stored on a computer system. The data types which are commonly used are as follows:

Data type	Description	Examples
Integer	Whole numbers, positive or negative	**42**, **-11**, **0**
Real	Numbers, including fractions or decimal points	**12.9**, **-17.50**, **28.0**
Boolean	True or false	**1** or **0**
Character	Letter, digit, space, punctuation mark or various other symbols	'**A**', '**b**', '**7**','**?**'
String	A sequence of characters	**'Computer science'** **'The cat sat on the mat'**

CAN YOU DESCRIBE THE RELATIONSHIP BETWEEN DATA STORAGE UNITS?

You should be able to:

- describe the relationship between data storage units:
  - Bit, nybble, byte, kilobyte, megabyte, gigabyte, terabyte, petabyte, exabyte, zettabyte, yottabyte

Bit, nybble, byte, kilobyte, megabyte, gigabyte, terabyte, petabyte, exabyte, zettabyte, yottabyte

Computer systems can only store and process Binary digITs, also known as BITs. A BIT is either a 1 or 0. When 8-bits are stored as a binary number, they are collectively called a byte.

	Symbol	Value
Byte	B	8 bits
Kilobyte	Kb	1024 bytes
Megabyte	MB	1024 Kb
Gigabyte	GB	1024 MB
Terabyte	TB	1024 GB
Petabyte	PB	1024 TB
Exabyte	EB	1024 PB
Zettabyte	ZB	1024 EB
Yottabyte	YB	1024 ZB

INTERESTING FACT

Half a byte (4 bits) is called a nybble.

CAN YOU CONVERT BETWEEN NUMBER SYSTEMS?

You should be able to:

- use techniques to convert between number systems:
  - Denary to binary and hexadecimal, binary to denary and hexadecimal, hexadecimal to binary and denary

Denary to binary and hexadecimal, binary to denary and hexadecimal, hexadecimal to binary and denary

On pages 20–3, we discussed three different counting systems denary, binary and hexadecimal. In this section, we will discuss how to convert between different number systems.

Denary to binary

One way of converting a denary number to a binary number is by drawing a Base **2** table from right to left.

128	64	32	16	8	4	2	1

In this example, we will convert the denary number 198 into a binary number. Take 198 and see if it is more than the first number on the left. In this case, 128 is the number on the left and so we write a 1 under the heading 128.

128	64	32	16	8	4	2	1
1							

We now deduct 128 from our original denary number, which leaves 70. The next number in our Base **2** table is 64. If the number remaining, 70, is more than the next number on the left, 64, write the number 1 under the heading 64.

128	64	32	16	8	4	2	1
1	1						

We now repeat the process again and deduct 64 from 70, which leaves 6. The next number in our Base **2** table is 32. If the number remaining, 6, is more than the next number on the left, 32, write the number 1 under the heading 32. However, in this case the number remaining is less than the next number on the left, so we write a 0 under the heading 32.

128	64	32	16	8	4	2	1
1	1	0					

This process is repeated until you reach the final heading and the binary number for the denary number 198 is found:

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

The binary number for the denary number 198 is therefore 11000110 (128 + 64 + 4 + 2).

Denary to hexadecimal

You may wish to convert a denary number into a hexadecimal number. To do this, take the number 198 from our previous example and draw a Base **16** table, from right to left, as before.

256	16	1

Take 198 and see if it is more than the first number on the left. In this case, 256 is the number on the left and so we write a 0 under the heading 256.

256	16	1
0		

The next number in our Base **16** table is 16. If the number remaining, 198, is more than the next number on the left, 16, work out how many 16s are needed without going over the number remaining. In this case it is C (C = 12, 12 x 16 = 192).

Remember that A = 10, B = 11, C = 12, D = 13, E = 14, F = 15

256	16	1
0	C	

We now deduct 192 from our remaining denary number, 198, which leaves 6. The next number in our Base **16** table is 1. If the number remaining, 6, is more than the next number on the left, 1, work out how many 1s are needed without going over the number remaining. In this case it is 6.

256	16	1
0	C	6

The hexadecimal number for the denary number 198 is therefore C6.

Hint

You may find it easier to convert a denary number into a binary number first and then into a hexadecimal number. See the example *binary to hexadecimal* below.

Binary to denary

To convert a binary number into a denary number, draw a Base **2** table from right to left and populate the table with the binary number you are converting. In this case we will use 00100011.

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

Simply convert the binary number into a denary number by adding the headings with a 1 under them: 32 + 2 + 1 = 35. The denary number for the binary number 00100011 is therefore 35.

Binary to hexadecimal

To convert a binary number into a hexadecimal number, there is a shortcut that you can use by drawing a Base **2** table from right to left and then populating the table with the binary number you are converting. In this case we will use 00101011.

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

Now split the Base **2** table into two smaller 4-bit Base **2** tables.

128	64	32	16
0	0	1	0

8	4	2	1
1	0	1	1

Now change the headings of the left 4-bit table.

8	4	2	1
0	0	1	0

8	4	2	1
1	0	1	1

Remember that A = 10, B = 11, C = 12, D = 13, E = 14, F = 15.

2 **B**

Now take the hexadecimal number of each 4-bit table and this is the converted hexadecimal number.

256	16	1
0	2	B

The hexadecimal number for the binary number 00101011 is therefore 2B.

Hexadecimal to denary

You may wish to convert a hexadecimal number into a denary number. To do this you may take the number C6 and draw a Base **16** table, from right to left as before.

256	16	1
0	C	6

Now multiply each heading to obtain the denary converted number.

C(12) x 16	=	192	
6 x 1	=	6	+
		198	

The denary number for the hexadecimal number C6 is therefore 198.

Hexadecimal to binary

To convert a hexadecimal number into a binary number, there is a shortcut that you can use similar to the one above by drawing two 4-bit Base **2** tables from right to left.

8	4	2	1

8	4	2	1

In this example, we will convert the hexadecimal number 2B into a binary number. First start by representing the first number, 2, in the left table.

8	4	2	1
0	0	1	0

8	4	2	1

Then complete the second table by representing B in the right table, remembering that B = 11.

8	4	2	1
0	0	1	0

8	4	2	1
1	0	1	1

Now re-label the headings in the left table as shown below and join the two 4-bit tables together to make one 8-bit table.

128	64	32	16
0	0	1	0

8	4	2	1
1	0	1	1

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

And so, the hexadecimal number 2B can be represented as 00101011 in binary number form.

HOW DO COMPUTERS INTERPRET INSTRUCTIONS?

You should be able to:

- demonstrate how computers interpret instructions:
  - how instructions are coded as bit patterns
  - how the computer knows if it is reading instructions or data
  - how sound can be sampled and stored digitally
  - how an image is represented by pixels in binary format
  - why metadata needs to be included in an image file (including height, width, colour depth)

How instructions are coded as bit patterns

Computer programs are compiled into machine code before they are able to run on a computer system. For more information see page 66.

Instructions to a CPU are processed as part of the fetch-decode-execute cycle, discussed on page 9.
A typical instruction to a CPU is to add two numbers together. Another instruction may be to subtract two numbers. All of these instructions are stored as machine code, using a set pattern of bits for each

operation, called the *opcode*. An example may be 1011 which could be the opcode for addition. 1010 may be the opcode for subtraction.

Along with the opcode, an *operand* may be included in the instruction which may include the location in memory containing the data needed for that instruction to be processed, eg the instruction **add 2** will be represented as **10110010**.

How the computer knows if it is reading instructions or data

Both instructions and data are stored as bit patterns in memory. It would be advantageous if the CPU was able to differentiate between the two, since the outcome may not be desirable if, for instance, data was processed as an instruction or *vice-versa*.

In some systems, there is a software flag which allocates certain locations on memory as 'instruction only' locations and other areas on memory marked as 'data only' locations. In other systems a hardware flag is used to differentiate between the two. This is where an additional bit may be allocated to each bit pattern to flag whether it is data or instruction.

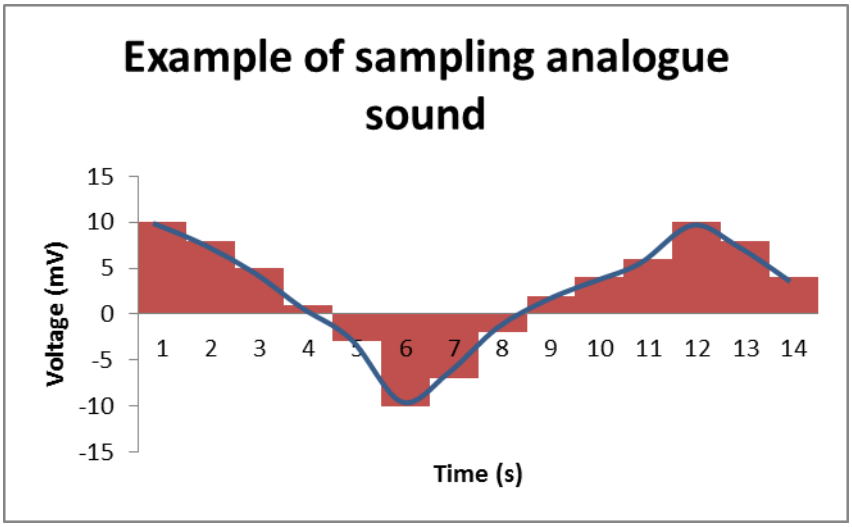
INTERESTING FACT

Hackers often exploit the fact that the CPU can be tricked into executing data as an instruction. This is sometimes called a *buffer overflow attack.*

How sound can be sampled and stored digitally

As we have already established, a computer system is only able to store and process binary digits as it is a digital device. Since this is the case, how can sound be stored as it is an analogue signal not digital? If an analogue signal, such as sound, is sent to a computer system, it has to be converted into a digital signal before it can be processed.

Sound is converted into a digital signal by a process called *sampling*. Sampling is where hardware, such as a microphone, measures the level of sound many times per second and records this as binary digits.

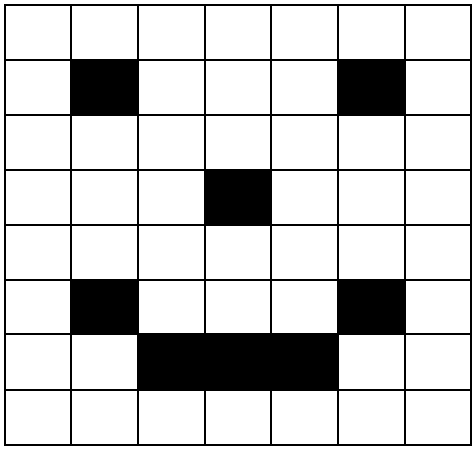


The number of times that the sound level is sampled per second is called the *sampling frequency*. The higher the sampling frequency, the better the quality of the sound recorded. A typical sampling frequency is 44,000 times per second, also known as 44 kHz. This is the sampling frequency used on most audio CDs.

Sound sampled at 44 kHz in stereo will produce a large amount of data and as such, this data may need to be compressed. Data compression techniques are discussed in further detail on pages 54-5.

How an image is represented by pixels in binary format

Images on a computer system are made up of thousands of small coloured dots, known as pixels (short for picture elements). Bitmap images are stored as an array of pixels. A black and white bitmap image will store a 1 for a black pixel and 0 for a white pixel.



This bitmap image can be represented using a 56 bits (or 7 bytes).

A colour bitmap image is stored by replacing the 1s and 0s above with a longer number that represents how much red, green and blue (RGB) is required in the colour of each pixel; this is known as *colour depth*. In a 256 colour palette, the image would require 1 byte of storage per pixel – so we would need 448 bits (or 56 bytes) to store the image above in colour. There are other colour depths available, which can store more information about the colours in each pixel of an image. The more information stored about the colour of each pixel, the larger the file size becomes.

You may also have heard of vector images. These images do not store the data by pixels, but are a set of instructions for drawing a geometric shape. The advantages of a vector image are that they can be scaled without loss of quality (pixellation etc.) and use less storage space.

Images require a large amount of storage space and as such, may need to be compressed.

Why metadata needs to be included in an image file (including height, width, colour depth)

The term metadata refers to 'data about data'. Key properties which are needed to display an image correctly are stored as metadata. Data such as an image's height, width and colour depth are typical examples of data stored in the metadata about an image. Without metadata, a computer system may render an image incorrectly on screen, such as displaying all pixels in one row.

Other data may also be stored in the metadata of an image file, such as the date the image was made, the geographical location of a photograph.

III. Computer system

CAN YOU IDENTIFY AND EXPLAIN THE FUNCTIONS OF AN OPERATING SYSTEM (OS)?

You should be able to:

- identify and explain the functions of an Operating System (OS):
  - managing resources and providing user interface

Managing resources and providing user interface

An operating system is software that manages a computer system. The operating system is loaded by the bootstrap loader. For more information see page 13. One of its primary functions is to manage resources. Here are some examples of how the operating system manages the computer systems resources:

Manages peripherals such as input and output devices

- Communicates with and sends data output to a printer/monitor/other valid output device
- Communicates with and receives data input to a keyboard/mouse/other valid input device

Manages printing using spooling

- Data is stored on hard disc/in memory/stored in a queue
- Document is printed when printer is free/in correct order
- Benefit of spooling – user can carry on working/log off when waiting for job to print

Manages backing store

- Ensures that data is stored and can be retrieved correctly from any disk drive
- Creates and maintains Filing system such as FAT or NTFS (accepted but not expected)
- Organise files in a hierarchical directory structure.

File compression

- The amount of data is reduced and the file is made smaller
- Compression is used to save disk space

Disk de-fragmentation

- Fragmented files are split up and stored on different parts of the disk
- Disk fragmentation will slow down disk access speed
- Disk de-fragmentation is when file parts are physically re-arranged (re-organised, moved, re-ordered on disk into the order required for access)

Manages memory (RAM)

- Ensures that programs/data do not corrupt each other
- Ensures that all programs and data including itself is stored in correct memory locations

Manages processes

- Ensures that different processes can utilise the CPU and do not interfere with each other or crash
- On a multi-tasking O/S ensure that all tasks appear to run simultaneously

Manages security

- Allows creation and deletion of user accounts
- Allows users to logon and change passwords

Another function of the operating system is to provide a user interface. Here are some examples of how the operating system provides a user interface:

- allows copying/deleting/moving/sorting/searching of file or folders
- allows access to system settings such as hardware
- provides a command line interface
- allows users to have more than one window open
- provides a graphical user interface (Windows, Icons, Menus, Pointers)
- provides user with errors/help messages
- allows customisation of interface, e.g. change desktop background/layout
- allows user to switch between tasks (programs/windows)

There are many different types of interface. These are discussed in depth on pages 17–19.

HOW CAN WE DESCRIBE THE FUNCTIONS OF COMMONLY USED PROGRAMS?

You should be able to:

- describe the functions of commonly used programs:
  - libraries, software development environment
  - disk organisation such as: file transfer, formatting, compression
  - system restore (roll back), disk defragmentation, control panel, system maintenance tools, virus protection, firewall

Libraries, software development environment

Libraries

A library is a collection of commonly used private functions and subprograms. Examples of private functions include standard mathematical operations such as square root, random number generators. Examples of subprograms include standard input/output routines, such as saving data to disk. These functions and subprograms can be called from within your program anytime, but only when the appropriate library has been linked. For urthe discussion see page 68.

The advantages of using standard libraries are that:

- related private functions and subprograms are stored in the same location
- time is saved as the programmer can simply use the private functions and programs stored in a library
- subroutines contained in a library have already been tested, so they should work reliably and not need further testing
- programs will contain less code and will therefore be easier to maintain

Most computer languages use standard libraries, although it is also possible to create your own custom libraries.

Software development environment

Software development environments, also known as Integrated Development Environments (IDE), provide programmers with various tools that are needed to create computer programs. Here are some of the tools and facilities offered by software development environments:

Editor	Allows a programmer to enter, format and edit source code
Compiler	Converts source code into executable machine code. Once compiled, a program can be run at any time
Interpreter	Converts each line of source code into machine code, and executes it as each line of code is run. The conversion process is performed each time the program needs to be run
Linker	A program which allows previously compiled code, from software libraries, to be linked together
Loader	A program which loads previously compiled code into memory
Debugger	A program which helps locate, identify and rectify errors in a program
Trace	A facility which displays the order in which the lines of a program are executed, and possibly the values of variables as the program is being run
Break point	A facility which interrupts a program on a specific line of code, allowing the programmer to compare the values of variables against expected values. The program code can then usually be executed one line at a time. This is called *single-stepping*
Variable watch	A facility which displays the current value of any variable. The value can be 'watched' as the program code is single-stepped to see the effects of the code on the variable. Alternatively a variable watch may be set, which will interrupt the program flow if the watched variable reaches a specified value
Memory inspector	A facility which will display the contents of a section of memory
Error diagnostics	Used when a program fails to compile or to run. Error messages are displayed to help the programmer diagnose what has gone wrong

Disk organisation such as: file transfer, formatting, compression

File transfer

File transfer is the ability to transfer data from one location to another. This can be done by simply copying a file from one folder (directory) to another, or from one storage medium to another. You may wish to carry out either of these tasks in order to organise your files better, using subfolders or to back-up your work onto a secondary storage device, such as a flash memory stick.

Formatting

Formatting is the process of preparing a disk for use. During this process, a new file system is set out on disk and all data may be erased in readiness for new data to be stored.

INTERESTING FACT
Certain specialist software can be used to "unformat" a formatted disk and recover all the data originally stored on it.

Compressing

Compression is the process of making a file size smaller. This may be advantageous as it allows more data to be stored on the disk and files may also be transferred more quickly. There are two methods of achieving disk compression; one is software based and the other hardware based.

Software based disk compression is often included as a facility of an operating system and so it is readily available on most computer systems. The disadvantage of this is that it slows down the process of reading and writing to disk.

Hardware disk compression requires specialist hardware, which can be expensive. However, it does not affect the speed of access as much as software based disk compression.

Disk based compression is always lossless. For further discussion see page 54.

System restore (roll back), disk defragmentation, control panel, system maintenance tools, virus protection, firewall

Many different system maintenance tools are included with operating systems that allow users to maintain the upkeep of their computer systems. Here are some of the tools below.

System restore (roll back)

System restore is the process of replacing lost or corrupt data by replacing it with an earlier backup.

INTERESTING FACT
Some modern viruses exploit the system restore facility by deliberately seeking out back-ups and placing copies of themselves there.

Disk defragmentation

Files are stored on computer systems that can, over time, become fragmented. This means that they are split and stored on different parts of the disk. If a file is fragmented, it takes longer for the disk heads to move between parts of the file, which slows the process of loading it.

Defragmentation is the process where files are physically re-arranged on disk so that they are no longer fragmented and the parts of each file are stored together. This improves the speed of accessing data from disk.

Control panel

Many operating systems use a control panel to give the user control of software and hardware features. It enables the user to change settings, such as sound, device and display settings all from one convenient location.

Virus protection

A virus is a computer program that is able to copy itself onto other programs often with the intention of maliciously damaging data. The consequences of a virus spreading can have catastrophic effects on computer systems, and therefore need to be prevented.

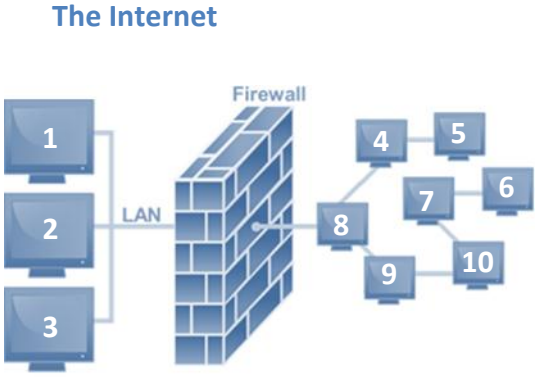
Virus protection software, also called anti-virus software, is a program that can be loaded into memory when the computer is running. It monitors activity on a computer system for the signs of virus infection. Each virus has its own unique 'signature' which is known to virus protection software and stored in a database. Data stored on a computer system is scanned to see if any of the virus signatures within the database exist on the system.

INTERESTING FACT
Some advanced viruses attempt to evade the virus protection software by changing their own code so that they no longer match the "signature" in the virus signature database.
These are known as *polymorphic viruses*.

There are many thousands of known viruses, and new viruses are created daily. Virus protection software therefore needs to be updated regularly to combat these.

Firewall

A firewall can be a software or hardware security system that controls the incoming and outgoing network traffic. Packets of data are analysed to determine whether they should be allowed through or not. More information on data packets can be found on page 43.



The basic function of a firewall is to monitor where data has come from and where it is going to, and to determine if this communication is allowed. It does this by checking a list of pre-defined rules. For example, if computer system number 1 above was attempting to connect to software port 80 on computer system 10, this would be allowed, as this would be listed in the pre-defined rules as a common port for browsing web pages. However, if computer system 6 attempted to access computer system number 2 on software port 139, this would be blocked as this would not be listed in the pre-defined rules and so this could be a sign of attempted hacking. A more advanced firewall would analyse the contents of each packet of data to ensure that it matches what is expected to be sent to that port.

HOW CAN WE DESCRIBE COMMON APPLICATION SOFTWARE?

You should be able to:

- describe common application software:
  - applications such as: word processing, spread sheets, presentation, database, drawing

Applications such as: word processing, spread sheets, presentation, database, drawing

A wide variety of software applications are used in the day-to-day use of computer systems. You will be familiar with many of these having studied ICT at Key Stage 3.

Here are some of the most common software applications and their typical uses:

Word processing	A computer program used for storing, manipulating, and formatting text.	Typing a letter.
Spread sheets	A computer program used for organising and analysing numerical data in tabular form.	Modelling household budget.
Presentation	A computer program used to display information in the form of a slide show.	Presenting new ideas to management.
Database	A computer program that stores a comprehensive collection of related data.	Gas company storing customers records.
Drawing	A computer program used for producing and editing digital images	Creating a company logo

IV. Networks

WHAT ARE PACKETS AND PROTOCOLS?

You should be able to:

- define packets and protocols:
  - IP, TCP, HTTP, FTP, protocol stacks

IP, TCP, HTTP, FTP, protocol stacks

Packet

A **packet** is a collection of data that is transmitted over a packet-switched network. For more information on packet-switching see page 55. Here is a simplified diagram showing what a packet will typically contain:

<table>
<tr><td>The source address</td><td>The destination address</td></tr>
<tr><td colspan="2">Information which enables the data to be reassembled into its original form</td></tr>
<tr><td colspan="2">Other tracking information</td></tr>
<tr><td>The data itself</td><td>A checksum that checks that the data has not been corrupted</td></tr>
</table>

Data may be split up into a number of packets. These packets are transmitted over a network and may take different routes to their destination. When all the packets have arrived, the data is reassembled.

INTERESTING FACT

When packets are transmitted over a Wi-Fi network, they can be intercepted by any device, this is called *packet sniffing*. If you perform thorough analysis on a large number of packets, you can often break the encryption. This is why security services, such as National Security Agency, do not allow any Wi-Fi devices on their network.

Protocol

A **protocol** is an agreed-upon format which allows two devices to communicate. The protocol, put simply, is a set of rules. These rules can include the following:

- handshaking, where two devices establish their readiness to communicate
- how the sending device will indicate that it has finished sending a message
- how the receiving device will indicate that it has received a message
- the type of error checking to be used
- agreement on the data compression method to be used

There are many standard protocols used with computer systems. Here is a table that illustrates the protocols with which you need to be familiar:

TCP/IP (Transmission Control Protocol/Internet Protocol)	Two protocols that combine to allow communication between computer systems on a network. IP is a protocol that sets out the format of packets and an addressing system. TCP is a protocol that allows packets to be sent and received between computer systems
HTTP (Hypertext Transfer Protocol)	HTTP is a protocol than can be used to transfer multimedia web pages over the Internet.
FTP (File Transfer Protocol)	FTP is a protocol that can be used when copying a file from one location to another via a network or the Internet. It is typically used for the transfer of large files, as it allows broken communications to resume transferring a file rather than having to restart.

A **protocol stack** is the term used when a programmer takes the rules of a protocol and implements them when creating a program. For instance, although computer A and B can both communicate using HTTP, the programmer of the protocol stack on computer A may have implemented the rules of the protocol slightly differently from the implementation of the protocol stack on computer B, which could lead to minor communication problems.

INTERESTING FACT

Although the Bluetooth protocol has been agreed, the protocol stack varies considerably from device to device. Try sending a photograph via Bluetooth from one smartphone to another.

CAN YOU DESCRIBE NETWORK TOPOLOGIES AND CONNECTIONS AND THEIR ADVANTAGES AND DISADVANTAGES?

You should be able to:

- describe network topologies and connections and their advantages and disadvantages:
  - Local Area Network (LAN), Wide Area Network (WAN), common network topologies
  - the connections necessary to link a stand-alone computer to a network

Local Area Network (LAN), Wide Area Network (WAN), common network topologies

Networks

A network consists of a number of computer systems connected together. There are many advantages and disadvantages of using a computer network over a stand-alone computer.

• Share hardware • Share software • Share data/files • Easier for internal communication/can send email • Central backup • Easier to monitor network activity • Centrally controlled security • Can access data from any computer	• A network manager may need to be employed – expensive • Security problems – files sent between computers could spread a virus • Hackers can gain access to data more easily • If the server is down, all workstations on the network are affected • Initial cost of servers, communication devices, etc. can be expensive

There are two different types of network, namely a **Local Area Network (LAN)** and a **Wide Area Network (WAN)**.

A LAN is a network in which the computer systems are all located relatively close to each other, for example, in the same building or on the same site, such as a school.

A WAN is a network in which the computers systems are all located relatively distant from each other, for example, in different buildings all over the country or in different countries. The Internet is an example of a WAN. You will note that many LANs could be linked using a WAN.

Computer networks use agreed upon protocols to communicate, i.e. common methods of sending data and consistent data formats. If they did not agree on the protocols to be used, the individual computer systems would not be able to communicate with each other. For further discussion see page 44.

Network topologies

A network topology is the theoretical layout of computer systems on a network. There are a number of different network topologies. Some common network topologies include:

- bus network
- ring network
- star network

Bus network

The computer systems, also called *nodes* of the network, are each connected to a single cable on which data can be sent, called the bus. A bus network has terminators on each end, which is needed to ensure that the network functions correctly.

The bus carries packets along the cable. As the packets arrive at each computer system, it checks the destination address contained in the packet to see if it matches its own. If the address does not match, the computer system ignores the packet. If the address of the computer system matches that contained in the packet, it processes the data.

• Easy to implement and add more computer systems to the network • Quick to set up – well suited for temporary networks • Cost-effective – less cabling	• It is difficult to troubleshoot the bus • Limited cable length and number of stations – performance degrades as additional computers are added • If there is a problem with the main cable or connection, the entire network goes down • Low security – all computers on the bus can see all data transmissions • Proper termination is required • Data collisions are more likely, which causes the network to slow down. A collision is when two computers try to send a packet at the same time

Ring network

In a ring network, computer systems are connected in a ring or a loop. Around the ring,packets are sent, being passed from one computer system to the next until they arrive at their destination.

• Data is quickly transferred without a bottleneck – consistent data transfer speeds • The transmission of data is relatively simple as packets travel in one direction only • Adding additional nodes has very little impact on bandwidth • It prevents network collisions.	• If any of the computer systems fail, the ring is broken and data cannot be transmitted efficiently • If there is a problem with the main cable or connection, the entire network goes down • It is difficult to troubleshoot the ring • Because all nodes are wired together, to add a another you must temporarily shut down the network

Star network

In a star network, each computer system is connected to a central node, also known as a hub.

• Good performance/fast network speed • Easy to set up • Possible to add more computer systems without taking the network down • Any non-centralised failure will have very little effect on the network • Minimal network collisions • Better security	• Expensive to install – more cabling required • Extra hardware required, such as a hub

The connections necessary to link a stand-alone computer to a network

To connect a computer system to a network, a Network Interface Card (NIC) is required. One method of connection is provided by a physical hardware port allowing a cable to connect your computer system to the network. The second method is to connect a computer system using a wireless connection, called a Wi-Fi.

Typical network speeds

A physical connection may be made using:

- a copper cable, with typical data transfer speeds of between 100 Megabits per second (Mbps) and 1 Gigabit per second (Gbps)
- a fibre-optic connection which has typical data transfer speed of between 1-10 Gbps

Wi-Fi connections have typical data transfer rates of 54 – 108 Mbps. However this can be severely affected by the distance between the device providing the Wi-Fi connection and computer systems. The data transfer rates can also be severely affected by atmospheric conditions, in particular heavy rain.

You should be able to:

- discuss network security techniques and network policies:
  - security: user access levels, suitable passwords, encryption techniques
  - network policies for: acceptable use, disaster recovery, backup, archiving

Security: user access levels, suitable passwords, encryption techniques

Network security is of paramount importance to any network as the loss of data, in particular, personal or confidential data can have many serious consequences.

User access levels

It is not desirable that every user should be able to access all the data on a computer system. User access levels are one method used to allow certain users read and/or write access to data on a computer system. For example, in a program used within a company, an administrator, possibly the owner, will have read and write access to all data on the system. A secretary, however, will not have access to confidential data such as employees' salaries. User access levels will define which users have different types of access to data.

Suitable passwords

Passwords are commonly used to prove a person's identity to a computer system, thus allowing them access to relevant data.

Different programs may require a user to use different complexities of password, as well as different character lengths. An example of a simple password may be the user's town of birth, or the word 'password'. A more complex password may require the user to use a combination of upper and lower case alphanumeric characters, for example'Pa55word1234'. Increasingly, computer programs require users to use a combination of upper and lower case alphanumeric characters as well as other non-alphanumeric character such as @ ! ~ - / \ %, for example 'P@55word/1234!'.

Obviously, short simple passwords can be either guessed by another user or a hacker may have access to programs that have the ability to try multiple guesses in quick succession. This is known as a *brute force*

attack. Passwords that use a combination of upper and lower case alphanumeric characters as well as other non-alphanumeric character, will be much harder to guess and will take longer to 'brute force'.
As a rule of thumb, the following formula can be used to determine the number of attempts it would take to brute force a password.

$$Attempts = Number\ of\ characters^{Password\ length}$$

So a password, such as 'computer' (8 characters), which only contains lower case characters from the 26 letter English alphabet will take:

$$Attempts = 26^8 = 208{,}827{,}064{,}576$$

(on a typical 3.5GHz computer, this would take less than 6 seconds to brute force)*

Whereas a password that contains upper and lower case alphaneumeric characters, such as 'Computer1' (9 characters), has 26 + 26 + 10 = 62 possible characters. This will take:

$$Attempts = 62^9 = 13{,}537{,}086{,}546{,}263{,}552$$

(on a typical 3.5GHz computer, this would take just over 1 hour to brute force)*

*assuming one attempt per clock-tick

Encryption techniques

Encryption is the conversion of data, using an algorithm, into a form called cyphertext that cannot be easily understood by people without the decryption key.

On pages 23–4, we discussed AND, OR and NOT logical operations on a computer system. When data is encrypted, a different logical operator is sometimes used, called the XOR logical operator.

XOR

The XOR logical operator has two inputs and one output. The output is 1 only if A and B are different.

Input (A)	Input (B)	Output (A XOR B)
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

When encrypting data, the XOR logical operator is performed on the original data and a *key*. The key is a secure binary number, known only to the sender and recipient.

In this example, we will encrypt the data 10101010, using the key 11110000.

Original Data	**10101010**	
Key	**11110000**	XOR
Cyphertext	**01011010**	

The original data, 10101010, is now encrypted and can be transmitted as 01011010.

To recover the original data, the cyphertext is *XOR'ed* with the key.

Cyphertext	**01011010**	
Key	**11110000**	XOR
Original Data	**10101010**	

Other, more complex techniques are also used to encrypt data. These include SHA256 and Blowfish.

Network policies for: acceptable use, disaster recovery, backup, archiving

Acceptable use

Network policies are documents written to outline the rules that users are required to follow while using a computer network. Each document is often several pages long, written and agreed by a committee. Following its publication, network users will be expected to adhere to the rules.

Typical rules set out in these policies include a list of unacceptable types of website that should not be visited, activities that are not allowed on the network, such as gambling and installation of unauthorised software.

Disaster recovery

Given the amount of important data often stored on a computer network, it is essential that an effective disaster recovery policy be in place, in the event of data being lost due to a disaster. Disasters include:

- fire, flood, lightning, terrorist attacks etc.
- hardware failure, e.g. power supply unit failing
- software failure, e.g. virus damage
- accidental and malicious damage, e.g. hacking

There are usually three parts to a disaster recovery policy:

- **before the disaster:** risk analysis, preventative measures and staff training
- **during the disaster:** staff response – implement contingency plans
- **after the disaster:** recovery measures, purchasing replacement hardware, reinstalling software, restoring data from backups

Backup

A **backup** is the process of copying data so that it can be preserved and restored if the original data is lost.

Backups of all data should be made regularly as the older the backed up data becomes, the less likely it is to match any current data stored on a computer system.

A backup policy sets out how often and to what medium backups are made. The backup medium is generally different to the active storage medium. Historically, the medium used was magnetic tape backup For further information see page 16.

A typical backup policy would require that three different backups be kept at any given time, with one of these being stored off-site. The oldest backup copy would be named the *grandfather*, the second oldest backup being named the *father* and the most recent backup being called the *son*. When a new backup is made, the oldest backup, the *grandfather* is overwritten and becomes the *son* backup, with the original son becoming the *father* and the father becoming the *grandfather*. This backup policy is called the *grandfather-father-son* method.

Archiving

Data held on computer systems is often **archived**. Archiving is the process of storing data which is no longer in current or frequent use. It is held for security, legal or historical reasons. The process of archiving data frees up resources on the main computer system and allows faster access to data that is in use.

V. The Internet and communications

CAN YOU DESCRIBE HOW INTERNET TRANSMISSIONS WORK?

You should be able to:

- describe how Internet transmissions work
  - the Internet and the world wide web
    - necessary hardware to connect to the Internet
    - common file standards associated with the Internet, e.g. HTML, JPEG
    - compression and compression types (including lossy and lossless) for files to be transmitted via the Internet
    - encryption, data compression, data redundancy (for error detection and correction) and security
    - packet switching
    - routing
    - MAC addresses
    - IP addresses

The Internet and the World Wide Web

The Internet is a wide area network (WAN). The World Wide Web, abbreviated www, is a service available over the Internet.

Necessary hardware to connect to the Internet

Most computer systems use a *modem* to connect to the Internet. A modem, the abbreviation of modulator/demodulator, is a device which allows digital computer systems to communicate over analogue systems, most commonly the telephone network. Traditionally, whilst a modem was in use, the telephone line would be unavailable to make telephone calls.

Broadband modems were later released which allowed for faster data transfer and would also allow users to make telephone calls over the same line at the same time.

A third common type of hardware used to connect to the Internet is a *media converter*, which allows computer systems to connect to the Internet using fibre optic cable.

Typical data transfer speeds

- Modem: 56.6 Kbps
- Broadband modem: 8 – 24 Mbps
- Fibre optic: 100 Mbps

Common file standards associated with the Internet, e.g. HTML, JPEG

A number of common file standards are associated with the Internet. Some of these include:

.HTML	HTML, which is an abbreviation of HyperText Markup Language, is one of the main programming languages used when creating web pages
.JPEG	A format used for storing compressed images. The file type uses *lossy* compression and is favoured for its small file sizes that allow for quick download speeds while maintaining reasonably good quality
.PNG	Another format used for storing compressed images. The file type uses *lossless* compression and is favoured for excellent quality and that they are generated using a non-copyrighted algorithm
.FLA	The format used to store flash multimedia. The files can contain interactive games, videos and music
.MP3	The format used to store compressed audio. It uses *lossy* compression and is favoured for its small file sizes that allow for quick download speeds while maintaining reasonably good quality

Compression and compression types (including lossy and lossless) for files to be transmitted via the Internet

Compression is the process of making a file size smaller. This may be advantageous as it allows more data to be stored on the disk and files may also be transferred more quickly. There are two primary methods that are used to compress files stored on a computer system; these are *lossy* and *lossless*.

Lossless compression

Lossless compression uses an algorithm that compresses data into a form that may be decompressed at a later time without any loss of data, returning the file into its exact original form. It is preferred to lossy compression when the loss of any detail, for example in a computer program or a word-processed document, could have a detrimental effect.

A simplified version of lossless compression on a word-processed document may to be to replace a common string, such as 'the', with a token such as the symbol @. You will have learnt on page 21, that one character takes 1 byte of memory; therefore, the string 'the' would take 3 bytes.

	The word the, is the most frequently used word in the English language.	71 characters (bytes)
	@ word @, is @ most frequently used word in @ English language.	63 characters (bytes)

This is a 11% reduction in the file size!

Lossy compression

Lossy compression is a data compression technique that compresses the file size by discarding some of the data. The technique aims to reduce the amount of data that needs to be stored.

The different versions of the WJEC logo below show how much of the data can be discarded, and how the quality of the images deteriorate as the data that made up the original is discarded. Typically, a substantial amount of data can be discarded before the result is noticed by the user. The compression ratio is calculated using the simple formula:

$$Compression\ ratio = \frac{Original\ file\ size}{Compressed\ file\ size}$$

		
Original image 100 kB	Compressed image 10 kB (compression ratio = 100/10 = 10 or 10:1)	Compressed image 5 kB (compression ratio = 100/5 = 20 or 20:1)

Lossy compression is also used to compress multimedia data, such as sound and video, especially in applications that stream media over the Internet.

Encryption, data compression, data redundancy (for error detection and correction) and security

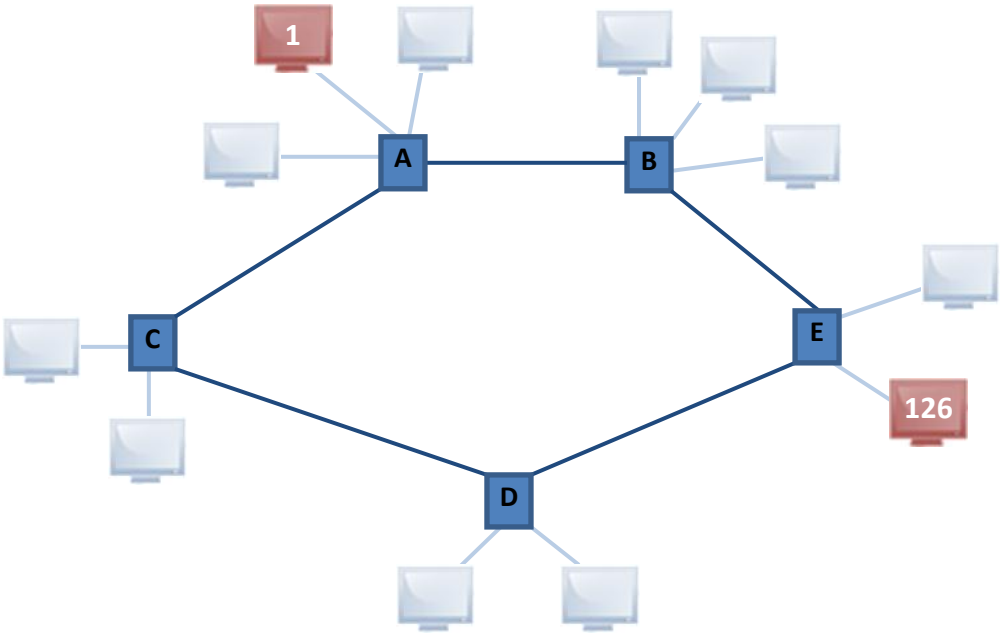
Encryption and data compression have been covered extensively on page 49 and pages 54–5 respectively.

Packet switching (including data redundancy)

We have discussed packets in detail on page 43. Packet switching is the process of delivering packets from one computer system to another using a designated device, such as a *switch* or a *router*. Packets are provided to a network for delivery to a specified destination. Each packet of data is redirected by a computer system along the network, until it arrives at its destination. Data may be split up into a number of packets. These packets are transmitted over a network and may take different routes to its destination. When all the packets have arrived the data is reassembled. The Internet is an example of a packet-switching network.

Routing

Routing is the name given to the method of selecting paths along which packets are sent on a computer network. Specialist computer systems such as routers, switches, bridges, firewalls and gateways construct in their memory a *routing table*, which stores a number of paths along which it is best to send packets to reach a specific destination. Maintaining accurate routing tables is essential for ensuring that packets are delivered as quickly as possible.



In the example above, computer system 1 is sending a packet to computer system 126. Clearly, the quickest route for the packet to arrive at its destination is to be sent from router A, on to router B followed by router E for delivery to computer system 126. This path would be determined by routing, using a routing table. A poorly constructed routing table may choose to send the packet from router A, on to router C followed by router D and then router E, for delivery to computer system 126. This would take longer and is not a good use of network resources.

Most routers use only one network path at a time, such as the preferred route above (Computer system 1 > Router A > Router B > Router E > Computer system 126). Some multipath routing techniques enable the same packets to be sent using multiple alternative paths at the same time. This means that in the event of Router B failing in the transmission above, the same packet would also have been sent via the alternative longer route set out above (Computer system 1 > Router A > Router C > Router D > Router E > Computer system 126), to ensure that the packet arrives at its destination.

MAC addresses

A MAC address (media access control address), also known as a physical address or a hardware address, is a unique hexadecimal number given to any communication device, such as a network interface card. An example of a MAC address is 74:E1:B6:8E:18:77. The address is usually stored in a communication devices' ROM. Hexadecimal notation is used as it allows for over 281 trillion different combinations of MAC address.

INTERESTING FACT

Although MAC addresses are designed to be unique and unchangeable, some devices or specialised software allow you to change your own MAC address. This is called *MAC address spoofing* and can be used by hackers to trick computer systems into providing data.

Routing tables store the MAC address of communication devices in computer systems on its network, as the address is permanent and does not change like an IP address.

A computer system can have multiple network interface cards, each with its own unique MAC address.

IP addresses

An IP address is an address which is allocated to a computer system on a network, usually by a DHCP server. For further details see page 58. Alternatively, you may assign your own IP address if you do not wish to rely on the services of a DHCP server. An example of an IP address is 195.10.213.120.

It is used by the TCP/IP protocol (see page 44) to uniquely identify computer systems on a network, thus allowing communication between them. In routing tables the corresponding IP address of a unique MAC address is stored and updated as necessary.

You should be able to:

- describe and explain Internet communication protocols
  - how Domain Name System (DNS) servers and Internet Protocol (IP) addresses work
  - advantages for the Internet Service Provider (ISP) and the user
  - why HTML is important as a standard for web page creation
  - the role of cookies
  - how search engines work

How Domain Name System (DNS) servers and Internet Protocol (IP) addresses work

A Domain Name System (DNS) is a distributed database that matches IP addresses to computer system resources.

One example of this is to match an IP address to a human friendly domain name. For example, if you wanted to visit the Google search engine, the computer system on which the website is stored has an IP address assigned to it; 173.194.34.191. Try typing this into the address bar of your web browser; you should be able to view the website that you would be more familiar with when accessing the domain name www.google.co.uk. Here your computer system sent a request to its DNS server for the IP address that is mapped to the domain name www.google.co.uk. The DNS server returned the IP address 173.194.34.191, which allowed your computer system to communicate with the computer system where the Google search engine is stored.

Of course, in reality, there are many different DNS servers located across the world. If your local DNS server does not store the address of the resource you are requesting, it will pass the request along to another higher level DNS server, such as your Internet Server Provider's (ISP) DNS server. If again the address is not found, you ISPs DNS server will pass the request on to a higher level DNS server which may be the DNS server responsible for an entire zone, such as the *.co.uk zone*. This continues until the address is found or the DNS query fails.

Another example where a DNS server is used is where a computer system, on joining a network, would query the DNS server for the IP address of other useful computer systems, such as the logon server, which stores the details of all usernames and passwords.

Advantages for the Internet Service Provider (ISP) and the user

An Internet Service Provider (ISP) is an organisation which provides end users with access to the Internet, usually via a connection such as dial-up, broadband or a fibre optic link.

An advantage of an ISP is that it provides and uses an agreed set of protocols and services, such as web browsing, email, FTP etc. For further discussion see page 44 on Protocols.

Why HTML is important as a standard for web page creation

HTML, which is an abbreviation of HyperText Markup Language, is a standard used when creating web pages. This is discussed in further detail on page 60.

Web standards, such as HTML, are important as the development of web pages is simplified as web programmers will be able to understand another developer's code. It is also important for the end users of web pages as following standards ensures that different web browsers are able to display web pages in the way in which they were intended.

The role of cookies

Cookies are data stored on a computer system. They allow websites to store a small amount of uniquely identifying data on your computer system while you are visiting their website. It may be useful as the website can then identify you without requesting that you identify yourself each time, i.e. by entering a username and password. Another use of a cookie would be when you wish to add items to a shopping basket over a period of time. The cookie allows you to store this information between separate browsing sessions.

How search engines work

A search engine is an application that can be accessed over the Internet which can be used to search for websites on a particular topic.

An index of topics will be built up by programs known as *bots* or *crawlers*. These programs visit websites and record information about their content. Users wanting information about a topic will access a search engine and search for their chosen topic by entering keywords that may be associated with the topic.

A search engine will then look through its index and return a list of websites that are associated with the keywords.

DO YOU UNDERSTAND HTML AND KNOW HOW TO USE IT?

You should be able to:

- use and demonstrate an understanding of HTML
  - use HTML to create a web page
  - use HTML tags: <html>, < body>, <h1>, <p>, , , <u>, <big>, <small>, , <img>, <a href="url">, and their closures

Use HTML to create a web page

HTML, which is an abbreviation of HyperText Markup Language, is one of the main programming languages used when creating web pages. HTML code consists of tags enclosed in angle brackets, < and >.

Use HTML tags and their closures

HTML tags commonly come in pairs, such as <html> and </html> or and . The first tag in a pair is called the opening tag and the second tag is called the closing tag. Between these tags, programmers can add text, more tags, comments and other types of text-based content.

The purpose of a web browser, such as Internet Explorer, Google Chrome and Safari is to read HTML code and render it on screen. The browser does not display the HTML tags, but instead uses the tags to interpret the content of the page.

The text between the <html> and </html> tags describes the web page. The text between <body> and </body> tags includes the contents of the web page.

The table below shows how unformatted text will look when placed within the commonly used formatting tags.

<h1>	</h1>	Computer science	**Computer science**
		Computer science	**Computer science**
		Computer science	*Computer science*
<u>	</u>	Computer science	Computer science
<big>	</big>	Computer science	Computer science
<small>	</small>	Computer science	Computer science

		Computer science	Computer science

Other tags include the tag, which can have a closing tag of . This is the paragraph tag which starts a new paragraph. When an element within a web page is hyperlinked, it is placed within the and tags. For example, WJEC will be displayed as WJEC.

The <img> tag is slightly different, as it does not contain a closing tag. For example, <img src="logo.gif"> will display the image file logo.gif.

Here is an example of how original text is formatted using HTML tags.

For Sale
Bluetooth Hands Free Car Kit
Make calls without wearing a headset with this Bluetooth v1.2 EDF Multipoint Hands-free Speakerphone! Visit www.edfweb.com to see. Simply pair this device to any Bluetooth enabled phone and talk hands-free today!

HTML

```
<html>
<body>

<h1><center>For Sale</center></h1>

<p> <b>Bluetooth Hands Free Car Kit</b></p>

<p>Make calls without wearing a headset with this Bluetooth
v1.2 EDF Multipoint Hands-free Speakerphone!</p>

<p>Visit <a href="http://www.edfweb.com/">www.edfweb.com</a> to
see.</p>

<p><i>Simply pair this device to any Bluetooth enabled phone
and talk hands-free today! </i></p>

</body>
</html>
```

For Sale

Bluetooth Hands Free Car Kit

Make calls without wearing a headset with this Bluetooth v1.2 EDF Multipoint Hands-free Speakerphone!

Visit www.edfweb.com to see.

Simply pair this device to any Bluetooth enabled phone and talk hands-free today!

VI. Algorithms

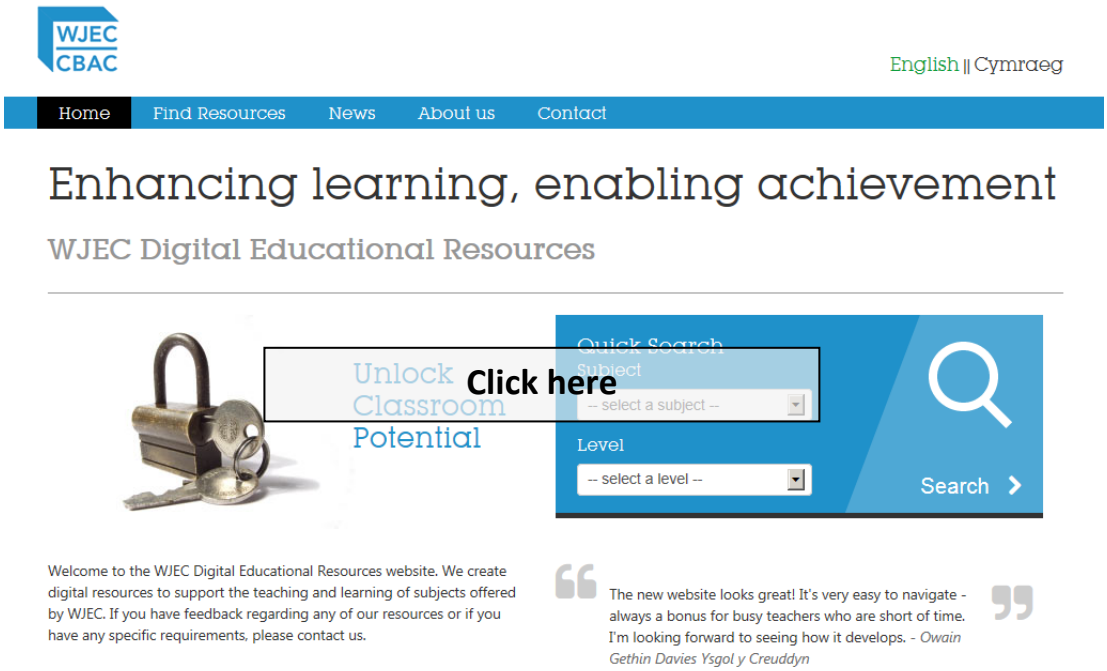
CAN YOU DEMONSTRATE AN UNDERSTANDING OF ALGORITHMS AND USE ALGORITHMS?

You should be able to:

- demonstrate an understanding of algorithms
  - interpret and dry run simple algorithms
  - choice may be influenced by data structure and data values
  - the need for accuracy in both algorithm and data
- use algorithms
  - write algorithms in pseudocode or flow diagrams to solve problems
  - complete algorithms
  - test and correct algorithms

Getting started with programming

Please use the range of resources on the WJEC resources website - http://resources.wjec.co.uk/ to learn the basics of a programming language. Follow the link below.



VII. Programming

HOW CAN WE DEFINE AND DESCRIBE PROGRAMMING TOOLS AND TERMS?

You should be able to:

- define and describe programming tools and terms
  - tools: high level languages, machine code, variables, constants
  - variables such as: local, global, static, dynamic

Tools: high level languages, machine code, variables, constants, data types

High level languages

A high level language is a programming language that allows code to be written. It is similar to a natural human language, such as Welsh or English. Some programmers prefer to use high level programming languages as they are easier to understand, learn and program. Their commands are similar to natural languages like English or Welsh and identifiers can be long and meaningful. High level programming languages also allow the use of powerful commands that perform quite complex tasks such as MsgBox in Visual Basic or the SORT clause in COBOL. High level languages are used when the execution speed is not critical, e.g. in common productivity applications, such as a word processor. Examples of common high level programming languages include:

- Basic
- Java
- Pascal
- COBOL
- C#
- C++

Machine code

Machine code is the opposite of a high level language in that it does not resemble any natural language and is made up entirely of bit patterns (instructions or data) that can be executed directly by the CPU. Examples of machine code instructions are opcodes and operands, discussed on pages 31–2. High level languages have to be converted into machine code before they can be executed by the CPU. For further discussion see pages 66–7. Although highly uncommon, some programmers may wish to program directly in machine code. This is primarily done when programming device drivers or where a fast execution speed are critical.

Variables

In computer programming, a variable may be required to store data that can change. Variables are given identifiers (names) that should reflect the data being stored in them. An example of a variable is txt_FirstName. This variable has a self-documenting identifier, which implies the type of data being stored in it is text containing a person's first name.

Constants

Constants are used in computer programming to store data that does not change. Constants are also given self-documenting identifiers that should reflect the data being stored in them. An example of a constant would be Pi = 3.14, as this is a self-documenting identifier that does not change.

Variables such as: local, global, static, dynamic,

Local and global variables

A **local** variable is a variable that is defined within a sub procedure and as such is only accessible from within that same sub procedure. This is known as its *scope*. A **global** variable is a variable that has a larger scope as it is defined globally and is therefore accessible from anywhere within a program. The advantage of defining a local variable over a global variable is that it is easier to track the changes to a variable and the reason for the changes when it is only used within a sub procedure. An advantage of defining a global variable over a local variable is that sometimes it can be the most efficient way of ensuring that an important piece of data is accessible to all sub procedures, e.g. the details of the user currently logged into a program.

Below is an example of an algorithm that calculates the area of a circle:

```
1  Pi = 3·142
2  Radius is real
3
4  declare subprocedure FindArea {procedure to calculate the area of a circle}
5  Area is real
6
7  start
8    Area = Pi * Radius * Radius
9  End
10
11  startmainprog
12     output "Type in the radius"
13     input Radius
14
15     call FindArea
16
17     output "The area is ", Area
18  endmainprog
```

An example of a global variable here is *Radius* as it can be accessed throughout the entire program. An example of a local variable is *Area*. It is a local variable as it has been defined within the sub procedure called *FindArea*.

Can you spot the error with the following line from the algorithm above?

```
17      output "The area is ", Area
```

How would you amend the algorithm to correct this problem?

Static and dynamic variables

Each time a program is run, **static** variables are stored in a location in memory and have a lifespan that lasts the entire time that program is running. For example, if you assign the number 2.14 to a static variable (and do not change it), the static variable will still contain 2.14 the next time you run the sub procedure and until the whole program stops running.

Each time a **dynamic** variable is defined, it is assigned a new location in memory and has a lifespan that ends when the sub procedure in which it was defined ends. For example, if within a sub procedure you assign the number 2.14 to a dynamic variable (and do not change it) and the sub procedure ends, the next time you run that sub procedure the dynamic variable will no longer contain any value.

CAN YOU DESCRIBE AND EXPLAIN TRANSLATORS?

You should be able to:

- describe and explain translators
  - why translators, compilers, interpreters, and assemblers are needed
  - the types of error that may occur in programming code such as: syntax, run time/execution, logical, linking, rounding, truncation

Why translators, compilers, interpreters, and assemblers are needed

Assemblers

An assembler is a program which coverts the low level assembly programming language into machine code. The assembler does this by converting the one word assembly instructions into an opcode, e.g. converting AND to 0010. It also allocates memory to variables, often resulting in an operand.

AND A	→	0010	0001
LOD B	→	0110	0010

Interpreters

Before high level programming languages can be run, code is converted by an interpreter, one line at a time, into machine code, which is then executed by the CPU.

Compilers

A compiler is used when high level programming languages are converted into machine code, ready to be executed by the CPU. There are four main stages of compilation:

Lexical analysis

- Comments and unneeded spaces are removed.
- Keywords, constants and identifiers are replaced by 'tokens'.
- A symbol table is created which holds the addresses of variables, labels and subroutines.

Syntax analysis

- Tokens are checked to see if they match the spelling and grammar expected, using standard language definitions. This is done by parsing each token to determine if it uses the correct syntax for the programming language.
- If syntax errors are found, error messages are produced.

Semantic analysis

- Variables are checked to ensure that they have been properly declared and used.
- Variables are checked to ensure that they are of the correct data type, e.g. real values are not being assigned to integers.
- Operations are checked to ensure that they are legal for the type of variable being used, e.g. you would not try to store the result of a division operation as an integer.

Code generation

- Machine code is generated.
- Code optimisation may be employed to make it more efficient/faster/less resource intense.

Translators

A translator changes (translates) a program written in one language into an equivalent program written in a different language. For example, a program written using the PASCAL programming language may be translated into a program written in the C programming languages by a translator.

The types of error that may occur in programming code such as: syntax, run time/ execution, logical, linking, rounding, truncation

Syntax	An error that occurs when a command does not follow the expected syntax of the language, e.g. when a keyword is incorrectly spelt	• **Incorrect:** IF A ADN B Then • **Correct:** IF A AND B Then
Runtime/ execution	An error that only occurs when the program is running and is difficult to foresee before a program is compiled and run	• Program requests more memory when none is available, so the program *crashes*
Logical	An error that causes a program to output an incorrect answer (that does not necessarily crash the program)	• An algorithm that calculates a person's age from their date of birth, but ends up giving negative numbers
Linking	Aan error that occurs when a programmer calls a function within a program and the correct library has not been linked to that program	• When the square root function is used and the library that calculates the square root has not been linked to the program
Rounding	Rounding is when a number is approximated to nearest whole number/tenth/hundredth, etc.	• 34.5 rounded to nearest whole number is 35, an error of +0.5
Truncation	Truncating is when a number is approximated to a whole number/tenth/hundredth, etc. nearer zero	• 34.9 truncated to whole number is 34, an error of -0.9

HOW CAN WE DESCRIBE AND EXPLAIN PROGRAMMING ERRORS?

You should be able to:

- describe and explain programming errors
  - strategies to avoid errors which may occur in programming code

CAN YOU DESCRIBE AND USE PROGRAMMING CONSTRUCTS?

You should be able to:

- describe and use programming constructs
  - one-dimensional arrays
  - pseudocode
  - algorithms
  - logical expressions
  - appropriate data types
  - variables and constants

HOW CAN WE DEMONSTRATE AN UNDERSTANDING OF AND USE PROGRAMMING?

You should be able to:

- demonstrate an understanding of and use programming
  - programming languages such as:
    - small domain-specific languages
    - visual languages
    - text-based languages
    - HTML to edit/create web pages
  - design and write solutions to given problems such as: write an app, a game or a tool to perform a task
  - debug programs
  - explain how a program achieves its intended result

VIII. Ethical, social, and legal aspects

WHAT IS THE IMPORTANCE OF CONFORMING TO PROFESSIONAL STANDARDS?

You should be able to:

- explain the importance of conforming to professional standards
  - the individual's own personal code, any informal code of ethical behaviour that exists in the work place, exposure to formal codes of ethics
  - legislation relevant to computing

The individual's own personal code, any informal code of ethical behaviour that exists in the work place, exposure to formal codes of ethics

A code of ethics, also commonly called a code of conduct, defines acceptable behaviour within an organisation. Higher standards are generally promoted when a code of ethics is accepted and followed by members of an organisation. It is useful as individuals working for the organisation have a benchmark upon which they can judge their own behaviour and that of others.

Informal and formal codes

Most small organisations do not have a formal written code of ethics and instead rely on senior members of staff to lead by example, showing what acceptable behaviour is. Members understand the informal code by observing how senior members conduct themselves, e.g. the type of language used in emails and behaviour towards clients.

Formal codes are written documents that outline expected behaviours within an organisation. Formal codes of ethics are usually enforced by the threat of disciplinary action should the code not be adhered to. Each code of ethics is different and usually reflects an organisation's ethos, values and business style. Some codes are short and set out general guidelines, whereas other codes are large documents that include a variety of aspects relating to an organisation's values, ethics, objectives and responsibilities.

An individual's own personal code

An individual's own personal code often supersedes the bare minimum requirements of an organisations ethics code. An individual's own personal code will vary from person to person as they choose to act upon their own ethical standards in their everyday actions.

Legislation relevant to computing

Many pieces of legislation govern the use of computer systems. Two such pieces of legislation are the:

- Data Protection Act 1998
- Computer Misuse Act 1990

Data Protection Act 1998

The Data Protection Act 1998 (DPA) was put in place by the Government in response to growing concerns about the amount of personal data being stored on and processed by computer systems. Organisations that store and process personal data are required to register with the Information Commissioner, who is the person responsible for the DPA. Organisations must register information on the type of data they wish to store and why it is being collected.

Organisations are required to adhere to the eight principles of the DPA. These specify that personal data must be:

- processed against loss, theft or corruption
- accurate and where relevant kept up to date
- adequate, relevant, not excessive
- prevented from being transferred outside EU to countries without adequate provision
- fairly and lawfully processed
- processed within the rights of subjects
- deleted when no longer needed
- used only for the purpose collected

There are a number of exemptions from the DPA. These include:

- the prevention or detection of crime
- the capture or prosecution of offenders
- the assessment or collection of tax or duty
- personal data by an individual for the purposes of their personal, family or household affairs
- national security and the armed forces
- personal data that is processed only for journalistic, literary or artistic purposes
- personal data that is processed only for research, statistical or historical purposes
- personal data relating to an individual's physical or mental health
- personal data that consists of educational records or relates to social work
- personal data relating to human fertilisation and embryology
- adoption records
- statements of a child's special educational needs

- personal data processed for, or in connection with, a corporate finance service
- examination marks and personal data contained in examination scripts

Computer Misuse Act 1990

When the use of computer systems became widespread, the Computer Misuse Act 1990 (CMA) was put in place to help combat issues arising from their misuse.

The CMA makes it an offence to:

- access data without permission, e.g. looking at someone else's files
- access computer systems without permission, e.g. hacking
- alter data stored on a computer system without permission, e.g. writing a virus that deliberately deletes data

HOW CAN WE ENSURE THAT WE USE COMPUTER SYSTEMS RESPONSIBLY AND EFFECTIVELY?

You should be able to:

- use computer systems responsibly and effectively
  - respect the integrity of the systems used, including not divulging passwords or private keys to anyone else
  - recognise that certain data is confidential and that the intended use of all data must be respected
  - users should become familiar with and abide by the guidelines for appropriate usage of the systems and networks that they access