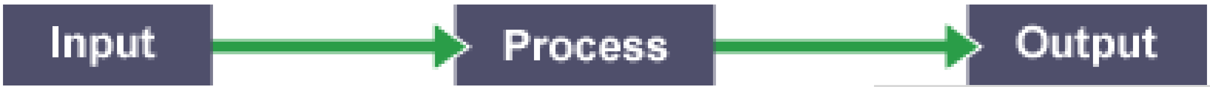
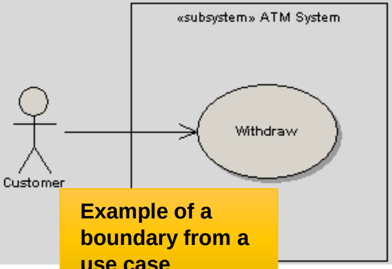
1. Introduction to Computing Systems

Collection of **hardware** and **software** that **work together** to achieve some **data processing** task.



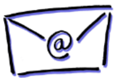
A **system boundary**, are also called **interfaces**.



Example of a boundary from a use case

Importance of computer systems

- **Cheaper** Manufacturing
- **Faster** access to information
- **New** ways of **communication** -**emails** SMS, cell phone



Computer System	Input	Output	Processing
ATM	Card details	Balance, Cash	Check balance, adjust balance
GPS	Signals from satellite, Input from User.	Route, Warnings	Check Position, Locate on Map
Travel Card (Oyster Card)	Money, Input Journey	Display Balance	Calculate Journey Cost

Types of Computer Systems

General-purpose systems

PC's**, desktops, smartphones**

- **Designed** →multiple **tasks,**
- Various apps → **Wide Variety & purpose.**

Dedicated systems

Single function or set of function

- i.e. **Ticket machine**

Control Systems

Control machinery**, rather than produce output.**

- Industrial robots → important app of control systems

Management systems

Management systems look at organisation's data i.e. employees

- School management Information systems
- File finder

Embedded Systems

Part of a lar ger system and are usually **control systems and portable devices.** Design for as fixed purpose.

- **USB**,
- Scientific Calculator
- Controllers of machinery in factory**.**

Expert systems

Designed to behave like a human.

- I.e. credit checks, diagnosing diseases etc.

They have three component parts:

1. A **knowledge base** (a **database** of facts)
2. An **interface engine** (**Software** that makes **deductions** using the knowledge base
3. An **interface** (to allow human user to access the system).

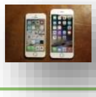
Advantages

- **Faster**
- **Cheaper**
- **Less power use**

Disadvantages

Devices many not use the same systems

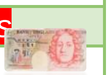
- Does not upgrade with technology
- Hard to backup



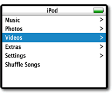
Advantages

- training aid to increase the expertise of staff
- Does not get tired
- No emotion

Disadvantages

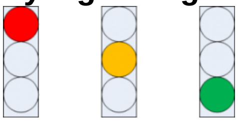


- effort and cost
- Most expert systems are menu driven – hard to pinpoint.
- Does not learn from mistakes



System reliability

Computers are used for everyday life and are important for:

Aircraft navigation	Railway signalling	Medical Systems
		

Mistakes in the design and process can lead to:

Not Found

The requested URL /404 was not found on this server.

Apache/2.2 Server at www.boingboing.net Port 80

- Downtime
- Pricey errors
- Data Loss
- Piracy

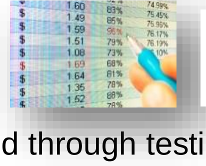
Data Integrity

Data has to be accurate all throughout its life, and stored data **reflects reality**.

Databases usually have **rules** to **prevent incorrect changes** been made to data.

Data integrity **can start** by:	Data integrity **can be stopped** by:
- Human Error - Software bugs - Software/Hardware Malfunctions (i.e. virus)	- Backing up data regularly - Security – Controlling access to data. - Using validation rules -

Data crunching - Computer systems can **collect** and **store** vast amounts of data e.g. Google Analytics



Reliability and testing

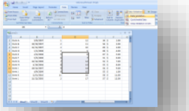
- Reliability is improved through testing.
- Testing is designed to uncover errors.

Testing can never be complete because:

- Complex Software
- Expensive testing
- Time consuming

Testing (Alpha – In-house, Beta – Users feedback)

Unit testing	Validation testing	Volume testing
Components of a program. Tested **part-by-part**.	**Data type** that a system can use.	**large** amounts of **data** is used to see if it can handle it & run for a long time



Computational thinking	Decomposition	Abstraction
Understanding how you **solve a problem** as a computer	**A computer** breaking down a **large problem** into **smaller chunks**	**Reducing** something to a **very simple set of characteristics**, chosen to be most relevant to the problem.

1+1=2

Standards

Standards →Rules and defined by a responsible organisation. Various categories of standards in computing:

Programming languages | Operating systems | Electrical interfaces

Standards are **important** because:

> **different manufactures** – equipment **works together**
> **Learning systems easier** - similar characteristics.
> Bring **costs down – no competition**

De facto Standards	De Jure Standards
De facto standards → **common usage over time.** Files and systems can be used by anyone. *E.g. QWERTY keyboard, Microsoft Word* 	De Jure → **by law**. Universally accepted – only those work *E.g. The 801.11. Wireless standard. PDF* 801.11
Proprietary Standard	**Industry Standards**
Proprietary standards → **owned by organisation.** Widely used, but not approved by certain bodies Reduce competition from rival competitors. *E.g., Apple computers only use Apple products.*	Industry standards → **set by recognised non-commercial organisations.** *British Computer Society (BCS).*
Open Standards	
Public Standards - **Publicly available**. They are not for profit and free of charge Produced collaboratively *E.g. They include HTML* <HTML> <html> <title>HTML</title> <body> This is HTML! </body> </html>	

Ethical (morally breaking) and Legal (law breaking) Issues – Copyright/Terrorism

Data Protection – Government laws ~(deals with Cyberbullying and trolling)

The “UK Data Protection Act” covers any data about living individuals.
The ‘Data Protection Act (DPA)’ – talks about access and selling data.

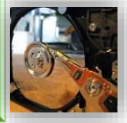
Patents – ownership
Copyright - right

Typical Data Protection Act laws in organisations include:

Allow people to view data held about them.

Data not used for direct marketing

Not use data in a way the can cause damage



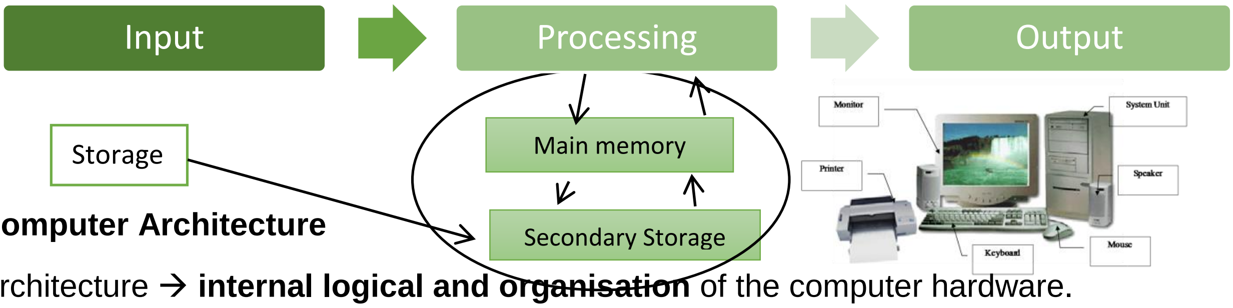
Cause cybercrimes → difficult to police as the internet crosses borders and law.

Environmental Issues

Travel – Reduction of transporting goods (email)	**Waste** –contains toxic and plastic material → long time to decompose.
Less Robots – Robots goods – efficient products 	**Consumes Energy** – Computers use energy Run air-conditioning systems to cool computers. Methods include: ○ Modern screen instead of CRT ○ Automatic standby

2. Computer Hardware

Hardware Components



Computer Architecture

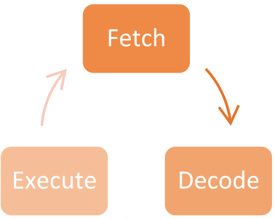
Architecture → **internal logical and organisation** of the computer hardware.
Von Meumann architecture → basis for all modern digital computers.
All **data** and instructions are **stored in RAM**, as **binary numbers**

The Central Processing Unit (CPU)

The CPU carries out all the processing in the computer.

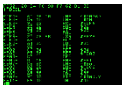
- The **Arithmetic and Logic Unit (ALU)** carries out all A & L **operations**
- The **Control Unit (CU)** uses signals to **control the flow of data** in the CPU.
- **Registers (R) [Internal memory]** - **small** amount of fast **temporary** memory where **ALU/CU** can store/**change instructions** values

The Fetch-Execute Cycle



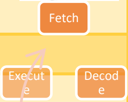
1. **Fetch** the instruction **from memory**
2. **Decode** the instruction to find out **what processing** to do.
3. **Execute** the instruction

The Boot Sequence

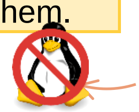


Controls all **instructions** for the **computer to run.** It starts via the boot loader, **Boot** sequence is **completed** → → → **OS takes control** for the CPU to progress.

The speed at which a CPU can process data depends on:

The CPU Clock	**How fast the CPU can run** **The** speed **of the** fetch-execute cycle **is determined by an** electronic clock chip**.** **The** clock speed **is measured in** cycles per second**, known as hertz** (Hz)	
Cache **Memory**	Small amount of storage **- temporarily holds instructions** that the CPU is **likely to reuse** → The CPU **control unit checks cache** for **instructions** before requesting data **Data** that is **in use** is **transferred** to cache memory to make **access to it faster**.	
Type & number **or processor cores**	Multi-core processors use multiple CPU's working together. The CPUs can all fetch, decode and execute instructions at the same time.	
	Advantage	***Disadvantage*** 
	more data is processed simultaneously	More complicated operating systems needed to manage them.

Benchmarking - is a test used to assess the **performance of a computer**.

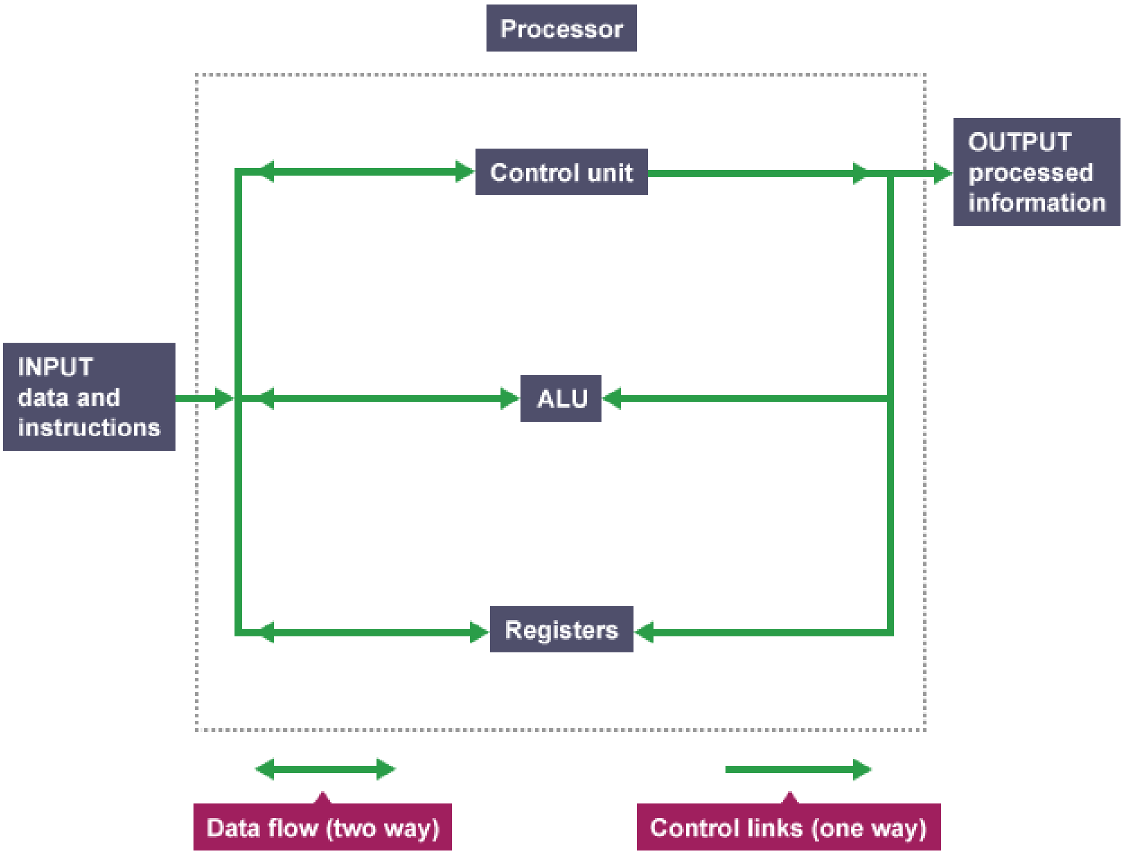


Running a program

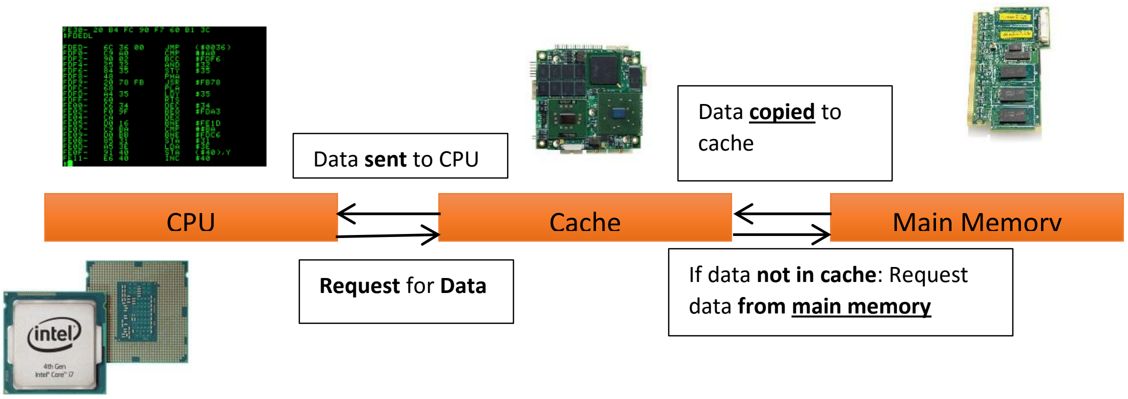
For a CPU to execute instructions → translated into Machine code
Machine code is simple **binary** codes that **activate** parts of the **CPU**.

The CPU only performs a few basic functions:

The main parts of the CPU



CPU and memory working together



Data sent to CPU

Data **copied** to cache

Request for **Data**

If data **not in cache**: Request data **from main memory**

Memory

RAM

Is the **main place** for **storing instructions** and data whilst a **program is being executed.** It is **also** called **main memory. Programs are loaded on RAM *and data is* sent** one by one to **decode and execute.**

ROM

Is **a flash memory chip** that contains a small amount of non-volatile memory

Computer Manufacturer Model Setup

Main Advanced Security Power Boot Exit

Boot-time Diagnostic Screen: [Enabled]
QuickBoot Mode: [Enabled]

Restore On AC/Power Loss: [Stay Off]
On LAN: [Stay Off]

First Boot Device [Removable Devices]
Second Boot Device [Hard Drive]
Third Boot Device [ATAPI CD-ROM]
Fourth Boot Device [Network Boot]

►Hard Drive
►Removable Devices
►Removable Format

Item Specific Help

Displays the diagnostic screen during boot

F1 Help ↑↓ Select Item -/+ Change Values F9 Setup Defaults
Esc Exit ←→ Select Menu Enter Select ▸ Sub-Menu F10 Save and Exit

Random Access Memory (RAM)	Read Only Memory (ROM)
Volatile (data is **lost** when power is turned off and **can change**)	**Non-Volatile** (data is **remembered** when the power is turned off and **can't change**)
Can be accessed and changed by the computer at anytime	Programmed during computer manufacture
Stores programs and data being **used** by the **computer**	**Stores instructions and data** required to **start up** the **computer**
Contains the **operating system**	Contains the **boot program**
Large (4GB or more in a typical computer)	**Small** 1 or 2 MB required for boot.

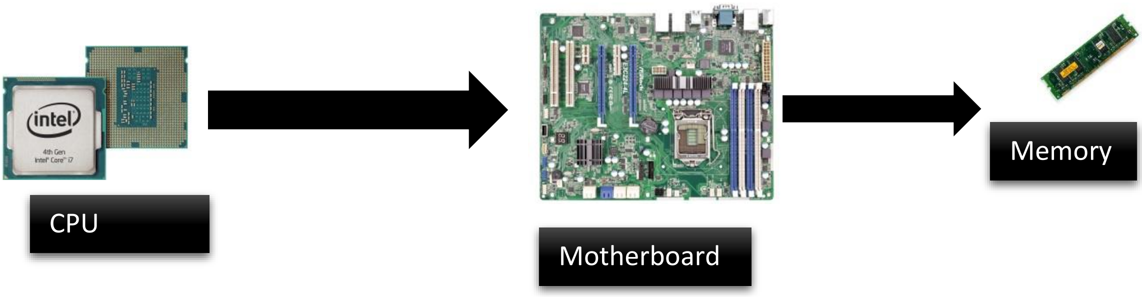
Virtual Memory	Flash Memory
Hard drive part used as an **extension to RAM.** Used to **hold** all **data** and programs required when the **computer doesn't** have enough **RAM** **Data is passed** – RAM ←→ Virtual Memory – Access to **virtual memory** is **slower** than RAM. Adding **more RAM reduces** the use of **virtual memory** and **improves performance**	Type of **ROM - can be rewritten** Used as a **portable** medium for **storing and transferring data**. Advances in processor technology such as **low power/high capacity** has led **to technological convergences** between **mobile telephones and computers.**

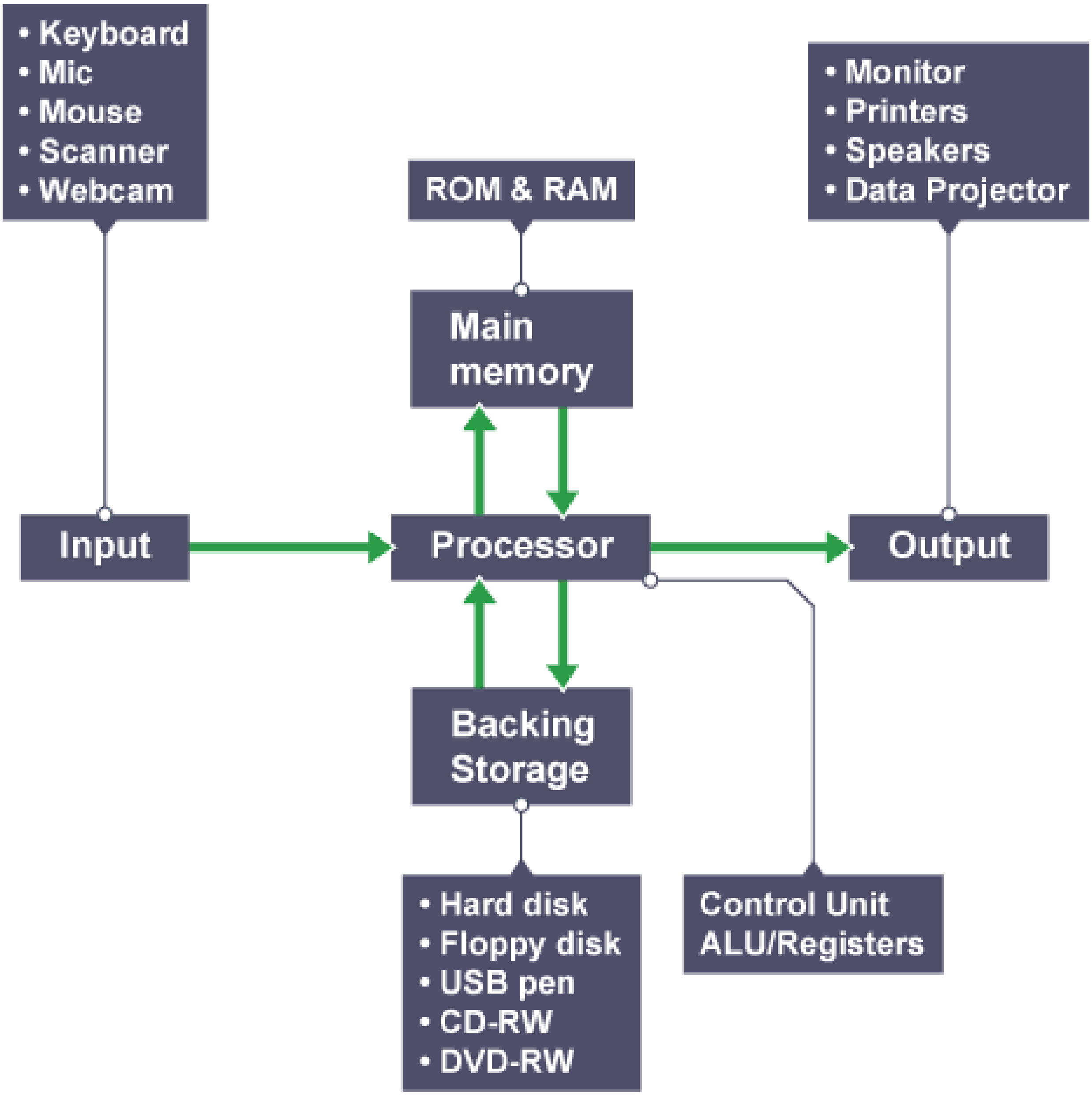
Processor types – CISC – e.g. Intel. **RISC** e.g. smartphones

Motherboard - circuit board that **connects the CPU** to the **memory** and all the other **hardware**. The **CPU sits** on the **motherboard** (also called the **logic board**).

Bridges manage how **data and instructions** are **transferred** between the **CPU**, **memory** and other devices.

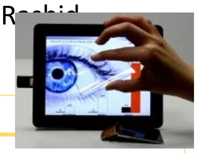
Latency is the time it takes for **components** to **respond** to a request.





Direct and sequential memory

Direct Memory (random access)
Any location jumped in storage to any moment, e.g. DVD chapter
Sequential Memory (ordered access)
Stored one by one in ordered sequence, e.g. film reel



Input Devices	Output devices
- Touch screens, Microphone, Camera, - RFID reader electronic version of bar code - foot mouse - limited hand movement - Puff-suck switch –limited physical mobility - Braille keyboard.	- Monitor - LCD and LED monitors, - touchscreens display for input and output - Plotter – reproduces engineering drawings - Printer – laser (fast/cheap) * Inkjet (quality)

Secondary Storage

Secondary storage → stores data and programs when the power is switched off.

Magnetic Hard Disk

A **magnetic hard** disk stores the **operating system**, installed programs and **user data**. Hard disks are:

- Strong & Reliable
- High Storage Size
- Low cost

Optical Disk

An optical disk is excellent for **transferring files or distributing software**. Optical disks are:

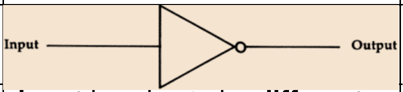
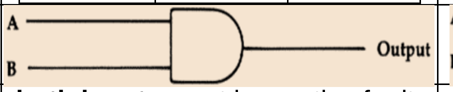
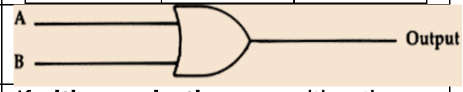
- Lightweight & Portable
- Good Storage size
- Low cost

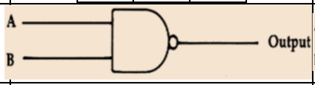
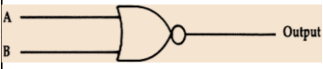
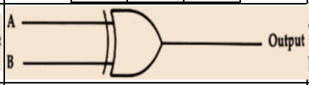
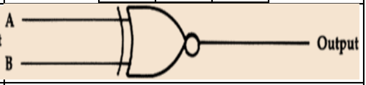
Flash Memory

Flash memory **consumes little power**. Flash memory is:

- Small Size
- Used in hand held devices
- East to transfer files

Binary Logic - All computers work in binary and use logic circuits for calculations.

<table>
<tr><td>Truth Table</td><td>NOT Logic Gates
<table><tr><th>Input</th><th>Output</th></tr><tr><td><b>0</b></td><td><b>1</b></td></tr><tr><td>1</td><td>0</td></tr></table></td><td>AND logic Gates
<table><tr><th>Input A</th><th>Input B</th><th>Output</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td><b>1</b></td><td><b>1</b></td><td><b>1</b></td></tr></table></td><td>OR Logic Gates
<table><tr><th>Input A</th><th>Input B</th><th>Output</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td><b>0</b></td><td><b>1</b></td><td><b>1</b></td></tr><tr><td><b>1</b></td><td><b>0</b></td><td><b>1</b></td></tr><tr><td><b>1</b></td><td><b>1</b></td><td><b>1</b></td></tr></table></td></tr>
<tr><td>Symbol</td><td>
</td><td>
</td><td>
</td></tr>
<tr><td>Define</td><td><b>input</b> is going to be <b>different</b> for the value of the <b>output</b></td><td><b>both inputs</b> must be postive for it to be <b>positive</b></td><td>If <b>either or both</b> are positive the output is positive</td></tr>
</table>

<table>
<tr><td>Truth Table</td><td>NAND logic Gates
<table><tr><th>A</th><th>B</th><th>O</th></tr><tr><td><b>0</b></td><td><b>0</b></td><td><b>1</b></td></tr><tr><td><b>0</b></td><td><b>1</b></td><td><b>1</b></td></tr><tr><td><b>1</b></td><td><b>0</b></td><td><b>1</b></td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>NOR Logic Gates
<table><tr><th>A</th><th>B</th><th>O</th></tr><tr><td><b>0</b></td><td><b>0</b></td><td><b>1</b></td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>Exclusive OR Gate
<table><tr><th>A</th><th>B</th><th>O</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td><b>0</b></td><td><b>1</b></td><td><b>1</b></td></tr><tr><td><b>1</b></td><td><b>0</b></td><td><b>1</b></td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td><td>Exclusive NOR Gate
<table><tr><th>A</th><th>B</th><th>O</th></tr><tr><td><b>0</b></td><td><b>0</b></td><td><b>1</b></td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td><b>1</b></td><td><b>1</b></td><td><b>1</b></td></tr></table></td></tr>
<tr><td>Symbol</td><td>
</td><td>
</td><td>
</td><td>
</td></tr>
<tr><td>Define</td><td><b><u>Opposite to AND</u></b>
A NAND gate is a AND gate followed by a NOT gate</td><td><b><u>Opposite to OR</u></b>
A NOR gate can be regarded as an OR gate followed by a NOT gate.</td><td>An exclusive OR gate Only exclusive 1 will be let through.</td><td>An exclusive NOR gate is the opposite of the Exclusive OR gate</td></tr>
</table>

Computers use binary values because it is **easy** to tell to switch, **on/off → 0/1.**
The **Von Neumann principle** → uses **binary** to **store data and instructions**.
Data and instructions are **stored in RAM**

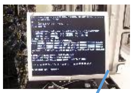
3. Software

Dedicated systems have software installed on a chip of some sort	
Embedded system	**Multi-purpose System**
Runs software for one function i.e. washing machine.	Runs software for different functions i.e. PC, Computer

Various types of software		
System software – **Software that controls hardware** and **data between locations (OS)** - **Hides** complexities - **No program** needs to be written	**Application software –** Handles jobs that users do (**Word on PC**)	**Utility Software –** Has **limited functionality** Used to maintain computer **systems**

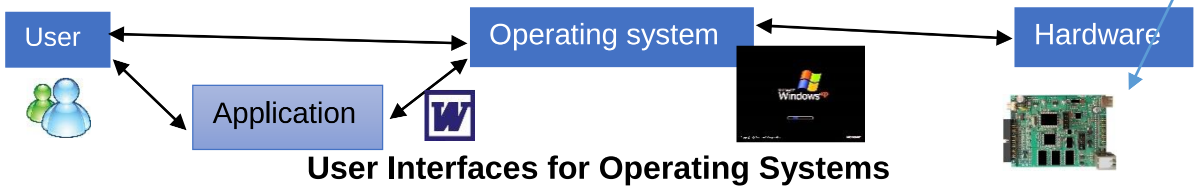
Operating System (OS)

The **operating system** is a **set of programs** that **controls** the **hardware** and lets the **users and applications** work **with** the **computer**.



Kernel → main part of the OS that actually **makes** the **hardware do things**.

Shell → Software users need to **communicate** with the **kernel**.



User Interfaces for Operating Systems

Different interfaces, give commands, asks questions and display responses to users

Command Line Interface (CLI)

CLI have to type in commands to certain languages to do a task. Useful for programmers as you can completely control a computer system. E.g., Linux terminal translates the commands

Advantage

✓ **Quicker** than GUI –*i.e. finding files within many folders.*

Disadvantages

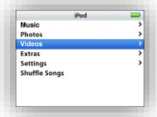
✗ Difficult for beginners to use – i.e. **learn the command** syntax
✗ **Hard to remember** all instructions/commands of tasks.

Menu-Driven Interface (MDI)

The user has to **choose options** from a list and **input the choice** by the code that is given.

Advantage

✓ **Simple user interface -** easy to use.



Disadvantage

✗ **Few items to select** from the menu.

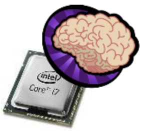
Graphical User Interface (GUI)

This **interface is on top** of the operating system **kernel** and **allows** easy access too many tasks. You communicate graphically with the System.

Advantages

✓ **Easy to Access -** Easy to access hidden files
✓ **Use of icons** – point and clicking.
✓ **Easier to use a mouse** – Just use a mouse.

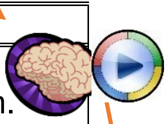
Disadvantages



✗ **More memory needed** – GUI's have large memo
✗ **Faster** and more **powerful processors needed** to run GUIs.

OS Management

Memory Management



Operating systems decide **what goes where** in memory. This is so memory **efficient** and important data is not overwritten during the running of a program.

1. To do this, **memory** is **divided** into **pages**.
2. A **program**, when it is been executed, is **called** a **process**.
3. When a job needs to be done, the **process is loaded** into a **vacant page**. The **operating system** keeps **track** of this and **protects** it from been **overwritten** by other processes.

Virtual Memory

The operating system **swaps jobs in and out** of memory, when there is **more jobs than memory** via virtual memory. It **uses secondary storage** (the disk) to **hold parts of a program**. This can be slow

The operating system keeps track of which pages are vacant and which processes are currently swapped in and out. It is divided into modules.

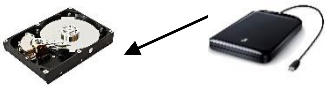
1. The **modules are stored separately** on a **secondary storage**.
2. When a **module is needed**, it is **loaded into memory** (swapped in) and run as a process (swapped out).
3. When a **different module is needed**, it can overwrite an unused module.

Peripheral Management

A **file** is a **named store** of data on a **secondary storage** medium.

Files can be:

- **Data** files – word processing, **database**
- **Program** Files – Operating Systems **(OS)**
- **Configuration** Data – Windows registry



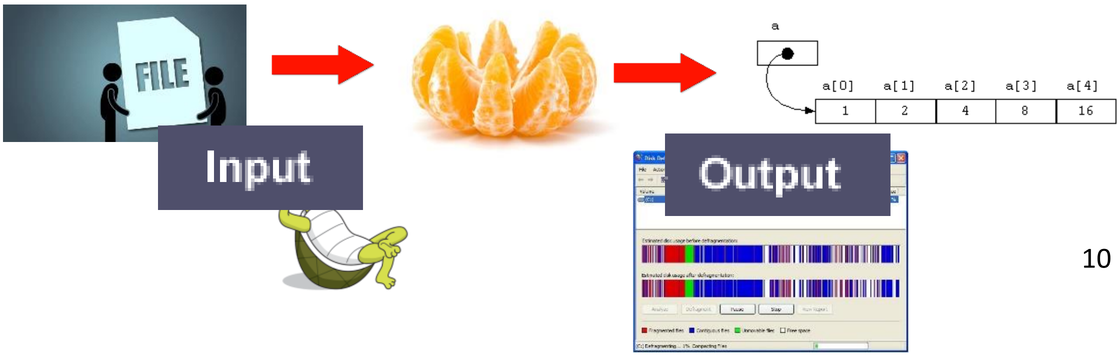
OS needs to tracks files. Stored on secondary storage → copied to main memory when needed.

1. **The OS** must know **where these files are located** on the storage medium.
2. When you **save a file**, the **OS looks up** where there is **free space** on the medium and **makes a record** of where it is **located**.
3. Next time the file is used the **OS** looks up **location** & **finds and retrieves** it.

Fragmentation and Defragmentation

Larger files than segment → files are split into blocks across many segments.
If a file is split **across many locations**, it takes **longer** to read and write it.
Each **block** contains **information (pointers)** about the location of the next block, so **the operating system** can **follow** to **pointers** to recover the **whole file**.

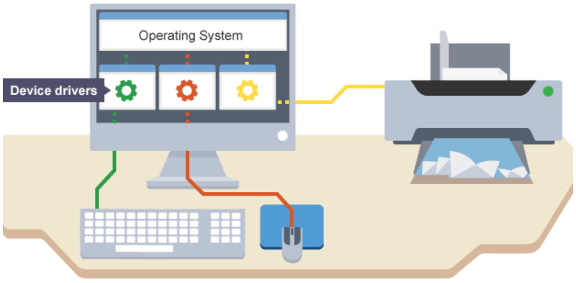
A process called **defragmentation** is used to **access files faster**. **Reorganises files** that have been **split up** on secondary storage, so fragments are close together.



Device Drivers

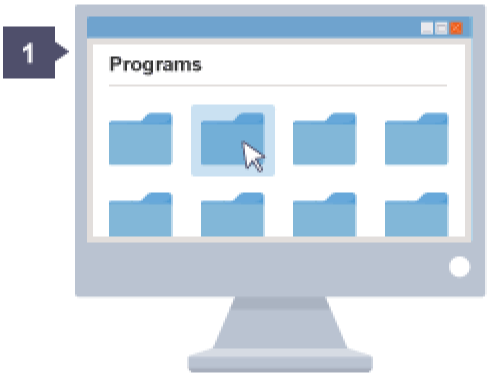
Used with the OS to take control of new input & output devices. Device drivers:

- **Creates an interface for devices** – Translations of device
- Allows an **operating system** to **communicate** with the **device**.
- Allows **devices** to **operate independently** of each other.

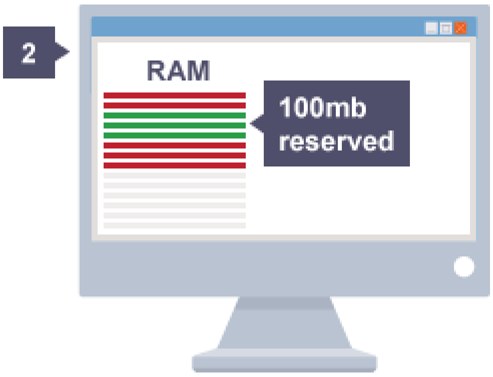


Managing the CPU with the OS – running software

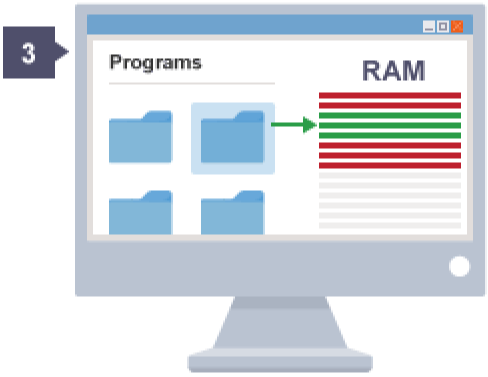
When the OS runs a piece of software it has to **find the program files on the storage drive load them into main memory** and **instruct the CPU** to start



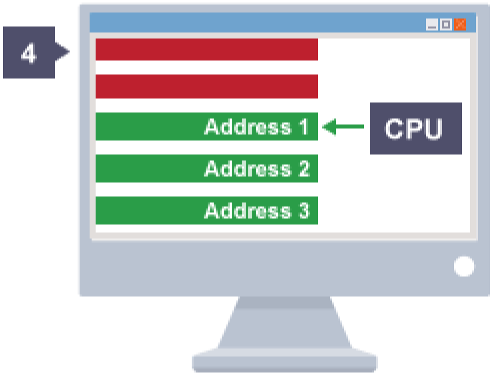
Program selected from hard drive



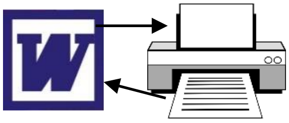
Space in RAM reserved



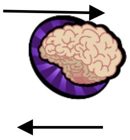
Program copied to RAM



CPU starts program



Multi-tasking



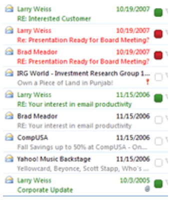
Multiprogramming: Serval programs **loaded** into memory at the same time.
Multitasking: the processes (programs) are **running** at the same time

E.g. **Word and print** so the **OS swaps the processes** from one to another

Scheduler

When there are **several processes sharing the single processor**, the operating system **uses a scheduler** in order to **allocate time**. The allocation is made according to a policy.

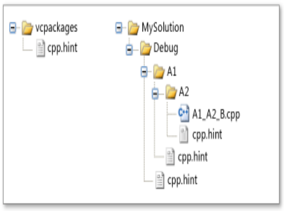
This might be, **shortest** job fist, **round robin**, **priorities**



Larry Weiss 10/19/2007 🟩
RE: Interested Customer
Larry Weiss 10/19/2007 🟥
Re: Presentation Ready for Board Meeting?
Brad Meador 10/19/2007 🟥
RE: Presentation Ready for Board Meeting?
IRG World - Investment Research Group 1... □
Own a Piece of Land in Punjab!
Larry Weiss 11/15/2006 🟩
RE: Your interest in email productivity
Brad Meador 11/15/2006 □
RE: Your interest in email productivity
CompUSA 11/15/2006 □
Fall Savings up to 50% at CompUSA - On...
Yahoo! Music Backstage 11/15/2006 □
Yellowcard, Beyonce, Scott Stapp, Who's ...
Larry Weiss 10/3/2005 🟩
Corporate Update

Files and Directories

OS organise files on secondary storage → Likely **Hierarchical systems:**



vcpackages
cpp.hint
MySolution
Debug
A1
A2
A1_A2_B.cpp
cpp.hint
cpp.hint
cpp.hint

- Files are stored in **Directories** (folders) and include **subdirectories** and file extension *(.mdb Access database)*
- Attributes - extra information on a file (i.e. last modified).
  - Allow **repeated use** of the same name file

Security

Virus

Replicator programs - **attach** themselves to legitimate programs. Do things as:

¶ **Damage** files	¶ **Take control** of a computer	¶ Get **Confidential** data

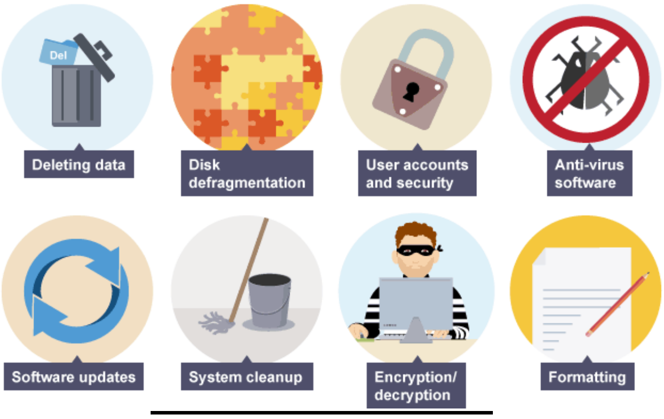
UNIX and Linux – less viruses as systems are more **tightly controlled**

Other forms of security include:

Authentication (User ID & Password)	**Privileges** rights assigned to users and groups – read, write, Execute	**Encryption –** Zipped files

Utilities

Utilities → **software tools** that **help maintain the system** easier.



Software comes from various sources. Different uses for:

- On time
- Reliable cost
- Acceptable quality

Custom written software

specially **commissioned** for a customer

- ✓ Have **exact features** required
- ✓ **No special adaptations** after installation.
- ✓ **Developer** can be **contacted directly**
- ✗ **Not extensively tested**
- ✗ **Expensive -** profit from one customer.
- ✗ It may take **long to develop**.

Off the shelf software

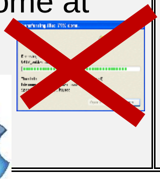
"**Shrink-wrapped**" software. *E.g. Office, Windows*

- ✓ It is **ready immediately & Low Cost**
- ✓ **extensively tested**
- ✓ **Forum** help – widely used
- ✗ **Will not be exactly** what the **customer needs.**
- ✗ It may need **extensive customisation**
- ✗ The customer might have to **search for training providers**

Open Source Software

Public domain. Improve their **skills** or for the **public good**. *E.g. Linux, Open Office, Firefox,*

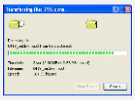
- ✓ Can be **free of charge**
- ✓ **Altered** due to available **source code**
- ✓ **Efficient - Many people** may work on it
- ✗ **No one to contact** - if there are problems directly
- ✗ **Updates may not happen** or come at irregular intervals.
- ✗ **Different platforms**



Proprietary Software

Software for profit. Only the complied code is realised. Users **buy a licence** to use it. *E.g. ASC*

- ✓ **Someone to go to** - if there is a problem.
- ✓ **Extensively tested.**
- ✓ **Updates** scheduled regularly.
- ✗ It can be **expensive**
- ✗ **Deliberate incompatibilities** may be introduced, so user can get locked in
- ✗ It may be **inflexible to a user's needs**.



4. Representing data

Morden computers work in binary because it is easy to represent two states. Made of loads of **transistors** → a **tiny switch** activated electronic signals **Binary** is a **number system** with just **two symbols, 0 and 1**.

- **Each digit** in a binary number is **called a bit (Binary digit) 2^7**
- **Denary** uses base 10 **- tens, units** (10^0, 10^1) and **binary uses** base two (2^0, 2^1)

Converting from binary to denary

The **binary number system** works like the familiar base10 system **using multiples of two instead of ten**, for column values. **In binary, the column values are**:

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

If we want to know what the **binary number 10110** is in **denary (base 10)**. Then we put the number into the table and add together the column values where there's a '1'.

Values	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**	**Total**
Bit				1	0	1	1	0	32
Add				16		4	2		

8 bits in a byte, the column goes up to 255 and 8 columns. **To make 256** you add another **column**, but it will become a more than one byte

Units

A group of 8 binary digits or bits is called a byte. As in base 10, we have numbers for key values based on 2^{10} or 1024 bytes.

Values	Name
8 bits	1 byte
1024 bytes	1 kilobyte
1024 kilobytes	1 megabyte
1024 megabytes	1 gigabyte
1024 gigabytes	1 Terabyte
Half a byte, 4 bits called a **nibble**.	

Converting from denary to binary

One method for converting **denary (base 10) to binary (base 2)** is **repeated division by 2**, recording the remainder each time.

For example, **convert 205 base** into **binary**.

Number	÷ 2	Total	Remainder	Binary Number for 205
205	÷ 2	102	1	10110011 205 in base 10 =110011001 in base 2 (binary)
102	÷ 2	51	0	
51	÷ 2	25	1	
25	÷ 2	12	1	
12	÷ 2	6	0	
6	÷ 2	3	0	
3	÷ 2	1	1	
1	÷ 2	0	1	

Negative Binary Numbers

In an 8-bit pattern, the first bit indicates positive or negative → 0 = Positive = 1
The other seven bits would be used to store the actual size of the number.

For example, **10001001** could represent -9

Negative Binary	1	0	0	0	1	0	0	1

The first bit, 1, indicates a negative number

The other seven bits indicate the number, 0001001 = 9

The **smallest negative binary number** using this method is **-127** (or 11111111)
The largest **possible number** is **+127** (or 01111111).

Negative numbers: Two's complement (TC)

With TC, the bit ≺ far left of the bit **most significant bit** (**MSB)**
MSB is used to **indicate positive or negative** and the **remaining** bits are used to **store the actual size of the number**.

Positive numbers always start with a 0
Four-bit, **positive**, two's complement numbers would be **0000 = 0** up to **0111 = 7**.

Negative numbers always start with a 1.
Smallest negative number is the largest binary value **1111 is -1** down to **1000 = -8**.

<u>Using two's complement for negative numbers</u>

1. Find the **positive binary** value for the **negative number you want** to represent.
2. **Add a 0 to the front** of the number, to indicate that it is positive.
3. **Invert or find the complement** of each bit in the number.
4. **Add 1** to this number.

Examples

Find -1 using two's complement numbers

- 1 = 001
    - Adding 0 to the front becomes **0001**
        - 'Inverted' becomes **1110**
            - Add 1 = **1111** (**-8 + 4 + 2 + 1 = -1**)

Find -4 using two's complement numbers

- 4 = 100
    - Adding 0 to the front becomes **0100**
        - 'Inverted' becomes **1011**
            - Add 1 = **1100** (**-8 + 4 = -4**)

This table shows the two's complement set for 4-bit numbers.

Denary	4-bit binary
-8	1000
-7	1001
-6	1010
-5	1011
-4	1100
-3	1101
-2	1110
-1	1111
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111

Adding Binary Numbers

Adding in **binary** uses the **same approach** as in **base 10**, we add the values and if the values is larger than the column, we “carry” the value into the next column.

To add 1101 to 1011 in binary

		1	1	0	1
+		1	0	1	1
Answer	1	1	0	0	0
Carried Values	1	1	1	1	

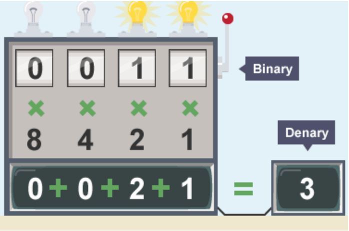
1+1 = 10 (in binary), we write 0 for the answer and carry 1

0+1+1 = 10 so we write 0 and carry 1

1+0+1 = 10 so we write 0 and carry 1

1+1+1=11 so we write 1 and carry 1

0+1=1 so we write 1



If a computer uses storage values that are **8 bits long** and **we add together 11000010 and 10111010**, the following happens.

		1	1	0	0	0	0	1	0
+		1	0	1	1	1	0	1	0
Answer	1	0	1	1	1	1	1	0	0
Carried Values	1						1		

We need a ninth binary digit.

If our computer has only **8 bits to store**, **this last bit will be lost** which is called **overflow**. **The result of the addition is too big to fit in the available space**.

Hexadecimal numbers

Large binary numbers hard to remember – Used for colour references (RGB) Hexadecimal is easy to remember. **An 8-bit byte splits easily into 4-bit nibbles**.

128	64	32	16		8	4	2	1
8	4	2	1		8	4	2	1
These are now 16’s								

In four bits, the largest value we can store is **1111 or 8+4+2+1 = 15**

If we are to represent each nibble using a single digit, we need more symbols.

In **hexadecimal**, we use **the letters A to F** to represent the base 10 numbers **10 to 15.**

Base 10 (Den)	Base 2 (BiT)	Base 16 (Hexl)
0	0	0
1	1	1
2	10	2
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10

Converting between hexadecimal and denary

Converting from Hex (base 16) to Denary (base 10) uses column values.

27 hex in base 10 is
32+7=39 in base 10

16	1
2	7
2 x 16 = 32	7 x 1 = 7

Convert **BD** in hex into base 10
176+13 = 189 in base 10

16	1
B	D
11 x 16 = 176	13 x 1 = 13

To **convert from base 10 to hexadecimal**, we can use the **repeated division** approach: **divide by 16** and record the remainders until the results is 0.

Convert 197 in base 10 into hexadecimal

197	÷16	=	12	Remainder	5
12	÷16	=	0	Remainder	C
				Answer	C5
197 base 10 is CS in hexadecimal.					

Converting between binary and hexadecimal

To convert from binary to hexadecimal**, split the binary number** into **two nibbles** and **convert each one** to get **the hexadecimal equivalent**.

Convert 10100011 (binary) into hexadecimal

8	4	2	1		8	4	2	1
1	0	1	0		0	0	1	1
10 in base 10 = A in Hex					3 in base 10 = 3 in Hex			
So 10100011 in A3 in hexadecimal								

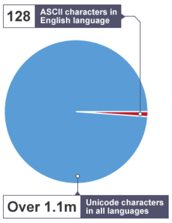
Converting between hexadecimal and binary

To convert from hexadecimal to binary, we **replace each hex digit** with the **equivalent to the binary nibble**.

- **Convert BD** (Hex) to a binary number
    - **B** (Hex) = **11** (base 10) = **1011** (binary)
    - **D** (Hex) = **13** (base 10) = **1101** (binary)
        - **BD** in hex is **10111101** in binary
- Convert **C5** in hex to a binary number
    - **C** (Hex) = **12** (base 10) = **1100** (binary)
    - **5** (Hex) = **5** (base 10) = **0101** (binary)
        - **C5** in hex is **10111101** in binary

Characters

- **Symbols** display by a computer are **represented by a code**.
- The **codes** used are **stored in binary which determines how many symbols**.
- **ASCII uses 7 bits** so can provide 127 characters or symbols plus a null character **(128 in total- 2^7).** → **Extended ASCII** 8 bits - 256 characters
- ASCII is in order - When we sort words 'Zebra' comes before 'apple'
- **Unicode uses 16 bits → 65000 possibilities**,
  - **ASCII** is a subset of **Unicode** – it keeps the same assignment of codes for the original **127 ASCII codes**

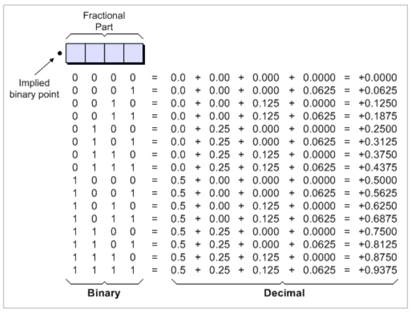


ASCII code examples

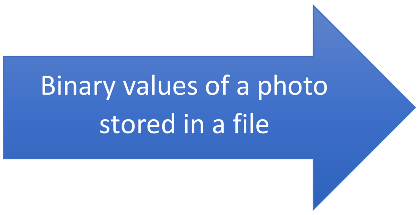
Binary	Hex	Decimal	Character
1111111	7F	127	Delete

Images

Images are **stored in binary** on a computer. This image of flowers is stored as many binary values.

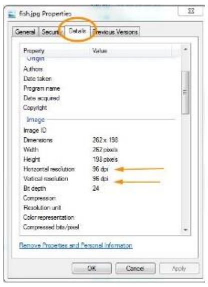


0	0	0	0	=	0.0	+	0.00	+	0.000	+	0.0000	=	+0.0000
0	0	0	1	=	0.0	+	0.00	+	0.000	+	0.0625	=	+0.0625
0	0	1	0	=	0.0	+	0.00	+	0.125	+	0.0000	=	+0.1250
0	0	1	1	=	0.0	+	0.00	+	0.125	+	0.0625	=	+0.1875
0	1	0	0	=	0.0	+	0.25	+	0.000	+	0.0000	=	+0.2500
0	1	0	1	=	0.0	+	0.25	+	0.000	+	0.0625	=	+0.3125
0	1	1	0	=	0.0	+	0.25	+	0.125	+	0.0000	=	+0.3750
0	1	1	1	=	0.0	+	0.25	+	0.125	+	0.0625	=	+0.4375
1	0	0	0	=	0.5	+	0.00	+	0.000	+	0.0000	=	+0.5000
1	0	0	1	=	0.5	+	0.00	+	0.000	+	0.0625	=	+0.5625
1	0	1	0	=	0.5	+	0.00	+	0.125	+	0.0000	=	+0.6250
1	0	1	1	=	0.5	+	0.00	+	0.125	+	0.0625	=	+0.6875
1	1	0	0	=	0.5	+	0.25	+	0.000	+	0.0000	=	+0.7500
1	1	0	1	=	0.5	+	0.25	+	0.000	+	0.0625	=	+0.8125
1	1	1	0	=	0.5	+	0.25	+	0.125	+	0.0000	=	+0.8750
1	1	1	1	=	0.5	+	0.25	+	0.125	+	0.0625	=	+0.9375



Binary values of a photo stored in a file

Computers are able to work out how to **turn these binary values into the image because** the file with the binary data contains **metadata *(data about the data).***



fish.jpg Properties

Property	Value
Origin	
Authors	
Date taken	
Program name	
Date acquired	
Copyright	
Image	
Image ID	
Dimensions	262 x 198
Width	262 pixels
Height	198 pixels
Horizontal resolution	96 dpi
Vertical resolution	96 dpi
Bit depth	24
Compression	
Resolution unit	
Color representation	
Compressed bits/pixel	

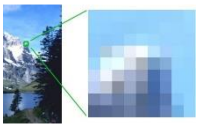
The height and width of the image is measured in pixels:

- A **pixel is one "dot"** in the image.
- The **number of bits** we use for a **pixel** determines **how many colours each dot can represent**:
  - **1 bit** - 2 colours i.e. **black and white**
  - **2 bits** - 2^2 or **4 colours**
  - **8 bits** - 2^8 or **256 colours**

The **more Bits Per Pixels (BPP)** the **greater the colour depth** and more bits stored

16 BPP is called **high colour** --------------------------------- **24 BPP** is called **true colour**

Bitmap images → **digital cameras, online -** file types →**JPEG, GIF and PNG**.



Bitmap images are organised as a **grid of coloured squares** called **pixels.** When **zooming** - the **pixels** are **stretched** → why bitmaps appear as **poor quality**

7x Magnification
Vector
Bitmap

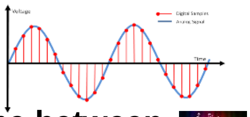
Vector images → **lines and curves**, using editable **coordinates** to **define** the image. Does **not** need to **store every pixel** → scaled without losing resolution - file formats is scalable vector graphics (**SVG**) (open standard)

Sound

Sound files have **metadata** so computers can **interpret the data accurately**.
The data stored includes:

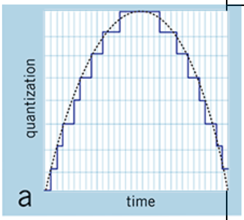
- **Audio codec -** decoding the audio
- **Sample rate -** Makes continuous signal a non-continuous signal

Sound is **analogue** form (**continuously varying**) form.
To **transfer** sound to a **computer**, it needs to be **digitally sampled.**

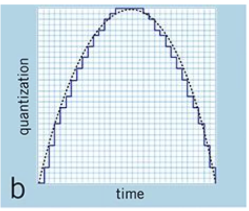


The **sample interval** is used to **describe the sample rate** and is the **time between samples** being taken – **the higher the sample interval, the lower the sample rate.**

Quantization → process of **aligning** musical notes to a **precise setting**.in the file.
Bit Rate → **Amount of space** used for each **sample**

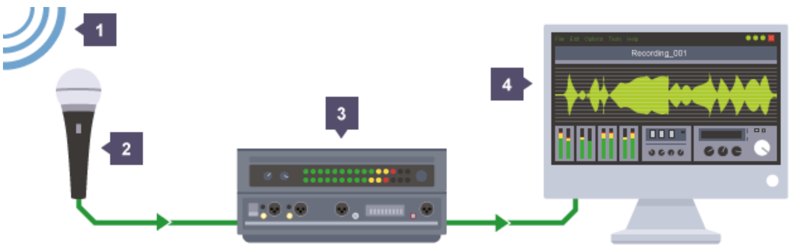


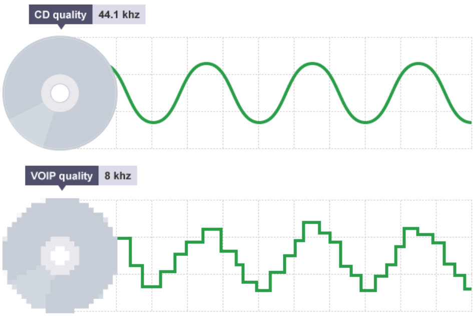
When **sound is sampled** at a **low rate**:	When **sound is sampled** at a **high rate**:
- Very **few samples are taken** - There is a **poor match between** the **original** sound and the **sampled** sound. - A **small size** file is required.	- Many **more samples are taken** - These is a **good match between** the **original** sound and the **sampled** sound - A **large size** file is required.



A **high bit rate** means **accurate sampling, better quality & bigger storage**.
An **MP3 track** is stored at 128 Kbits per second. A **CD** uses 1411.2kbits per second.

Digital audio quality





1. Microphone measures change in air pressure
2. Microphone translates air pressure into electrical voltage
3. Analogue to Digital Converter digitises the electrical voltage to bytes of information
4. Computer displays the digitised sound for manipulation

Sound input through a microphone is **converted** to **digital for storage** and manipulation

Digital sound is **broken down** into thousands of **samples per second**. Each sound sample is **stored as binary data.**

Factors that affect the quality of digital audio include:

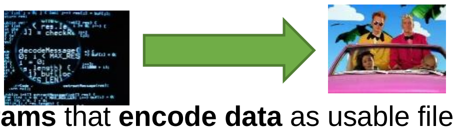
- **sample rate** - the number of **audio samples** captured every second
- **bit depth** - the **number of bits** available for each clip
- **bit rate** - the number of **bits used per second** of audio

Digital video

Digital films are usually around **24 frames**. This can be measured in Frequency (Hz)
Films have a **frame rate per second (fps) 50 or 60 fps → Similar to sample rate**

Video compression

Data **lost** during the **compression** causes **poor picture** quality **(.mp4)** or artefacts.
Artefacts is when **coloured blocks** that **appear and disappear** on the screen.



Codecs

Codecs are **programs** that **encode data** as usable files, **Algorithms** work out what data can be **removed and reduce file size**.

Instructions

When a computer **is instructed** to **run a program**, it is **designed to a specific location** in memory that contains the first instruction in the program.

The **CPU fetches** these **instructions and decodes** it in order to find out what to do next. The instructions are in **two parts**:

Operators – the instructions part e.g. 1001 may mean ADD to the accumulator
Operand – The data part e.g. adds values it finds in memory location 1011

For example, if the first location contains

1	0	0	1	1	0	1	1

This will be split into two parts.

1	0	0	1

Operator

1	0	1	1

Operand

Accumulator → special register in CPU used to **store results** of any **calculation**.
The **CPU cannot tell the difference** between **data and the instructions** and simply deals **what it finds with what it expects to find.**

5. Database

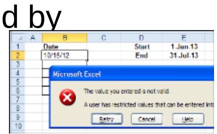
A **database** is a persistent **organised store of data** on a computer system.

- **Persistent** – It is **saved** on **secondary storage** for the future & **easy structure**
- **Accurate, Up-to-Date and Protected Access**

Errors result in → **wrong bills** → **wrongly credited** → **incorrect navigational** data.
Data can be **lost** because of, fire or **flood**, an accident – **deleting** data & **intruders.**
Data is protected against loss → (**cloud**) → **Server** → **Restrictions** → **audits.**

Data integrity

Data integrity means that the **data reflects reality**. Data integrity can be used by



	A	B	C	D	E
1		Date		Start	1-Jan-13
2		10/15/12		End	31-Jul-13

Microsoft Excel

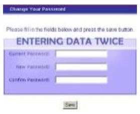
The value you entered is not valid.

A user has restricted values that can be entered into

Validation

Validation is the process of **checking data when it is input**. It checks that the **data conforms to certain rules**. Here are some examples:

Validation Method	Meaning	Examples of Use
Range check	Must fall between **certain limits**	Certain **way** of **D.O.B**
Presence Check*	**Must** be **filled in**	Name when **applying for a job**
Lookup check	Must match what is **held on file**	Check the **password is correct**
Format check	Must be a certain pattern	Car registration is LLNL **(letter & no.)**
Check digit	Exactly **same data** earlier entered	**ISBN number**
Length check	Must be certain no. of characters	long strong **Password**
Type check	Certain type	**Only numerals** allowed



Change Your Password

Please fill in the fields below and press the save button.

ENTERING DATA TWICE

Current Password:

New Password:

Confirm Password:

Verification

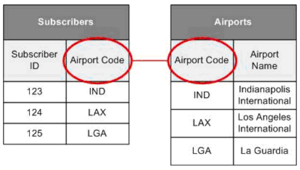
Verification is **checking** that the **data entered is correct**. It is **Visual or Algorithm.**

Database operations

A **subset of the data** in a database is called a **view**. Making **suitable views** for each staff member **increases the efficiency** and **reduces risks** for the **database**.

Standard operations for a database are listed under the term **CRUD.**

- C – Create
- R – Read
- U – Update
- D – Delete



Subscribers

Subscriber ID	Airport Code
123	IND
124	LAX
125	LGA

Airports

Airport Code	Airport Name
IND	Indianapolis International
LAX	Los Angeles International
LGA	La Guardia

Data matching ➲ **compares related databases** to look for a **relationship**.
E.g., compare **housing benefit claims** with **credit agency** data to **uncover fraud**.

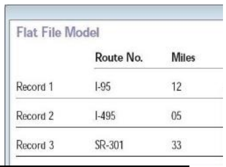
Data mining ➲ **compares unrelated databases**. **Unexpected relationships**.
E.g. connections between **purchases** and **postcodes** to see **what sells where**

Data Redundancy – unnecessary **data** is **repeated**

Data models

Databases **organised** via **models**. Data structure denotes reality, useful for owner.

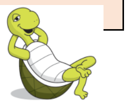
Flat file database



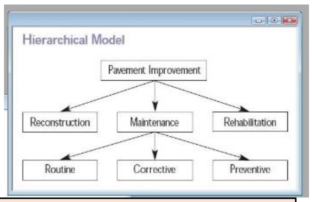
Flat File Model

	Route No.	Miles
Record 1	I-95	12
Record 2	I-495	05
Record 3	SR-301	33

Just rows (records) & columns (fields), suitable for address book using just a spreadsheet

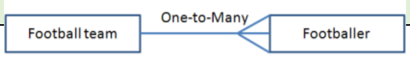
Advantage	Disadvantages
- Easy to **understand**. - Easy to **implement**. - Less hardware and software requirements. - Fewer skills required to handle flat file systems	 - Limited in functionality - Less security easy to extract information. - Data Redundancy (repeated cities in address book) - **Slow** for **huge** database & **Searching** is **time consuming** 

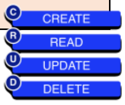
Hierarchical database



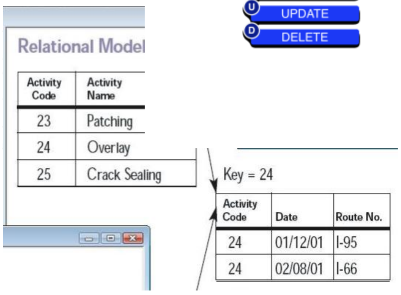
A hierarchical model is sometimes useful when making an **inventory**.
Some parts of **stock items** might always **belong with others**.
This model - **structure of a tree** records → nodes | fields →branches

Advantage	Disadvantages
 ▪ **Data Security**: **More security options** ▪ **Data Integrity**: parent child relationship ▪ **Efficiency**: large number of **1:M** relationship	 ▪ **Complex Implementation**: but easy to design ▪ **Database Management Problem**: changes to **all** application. ▪ **Operational Anomalies:** insert, delete and update anomalies





Relational database



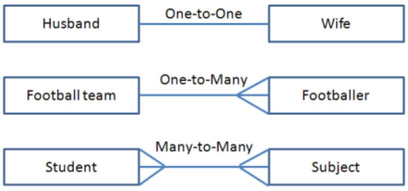
Relational Model

Activity Code	Activity Name
23	Patching
24	Overlay
25	Crack Sealing

Key = 24

Activity Code	Date	Route No.
24	01/12/01	I-95
24	02/08/01	I-66

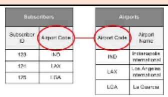
Relational databases most useful, common and **flexible**
Relational databases **store data** in **separate tables**.
The **tables** are **linked together** so related data is easily extracted.



You can show these relationships through entity-relationships diagram ERD

Advantage	Disadvantages
 ▪ Easier to **understand** ▪ **Flexibility**: be **easily manipulated** data. ▪ **Data Independence**: **normalization** structure ▪ **Use Data Manipulation Language**: SQL	 ▪ **Complex Implementation**: but easy to design ▪ **Lack of Structural Independence**: when we **change** the **structure** → compulsory to **change** the **application** too.

Normalisation



Subscribers	
Subscriber ID	Airport Code
123	IND
124	LAX
125	LGA

Airports	
Airport Code	Airport Name
IND	Indianapolis International
LAX	Los Angeles International
LGA	La Guardia

Normalisation - efficient databases → separate tables to reduce redundant data.
In databases data is broken down into smaller tables & relationships are linked.

The Database Management System (DBMS)

A DBMS is software that looks after a database → PC's or large organisations. It:

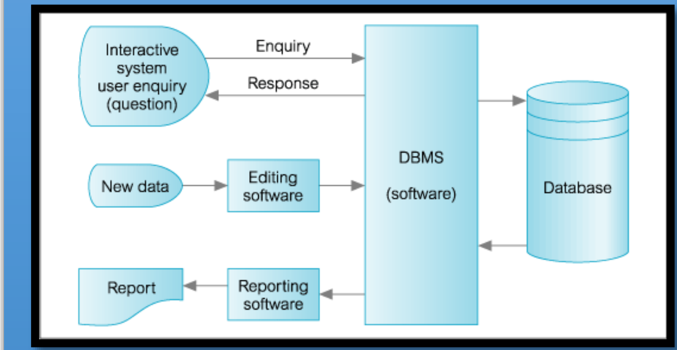
- Create database applications and **Protect data**
- **Run queries** to extract data
- Keep data **accurate**

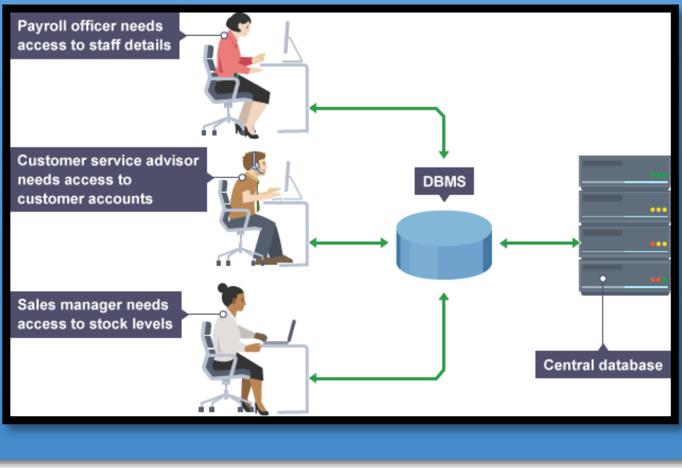
Advantage	Disadvantages
• **Data security: More users -** better enforcement of data privacy and security policies. • **Better data integration:** easier to see how actions in **one segment affect other segments.** • **Improved data access:** answers to queries - SQL	• **Increased costs:** Training & **licensing.** • **Frequent upgrade/replacement cycles** Product Life Cycle Introduction Growth Maturity Decline Revenue / Profit Time

Separation of data and application

The **DBMS** acts as a **go-between**, connecting **application to the data**. *E.g. Access, MySQL.* You must **separate** the **application** from the **data** so that:

- Programmers need not worry about **applications damaging** existing **data structures**.
- Data can be easily **shared** between **applications**.
- Data remains **consistent** because there is just **one copy for all applications**.



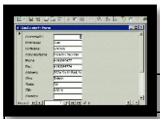


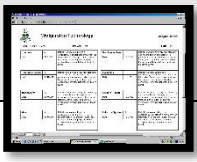
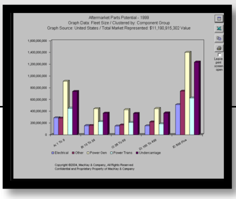
Payroll officer needs access to staff details
Customer service advisor needs access to customer accounts
Sales manager needs access to stock levels
DBMS
Central database

Transactions

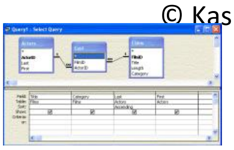
When a **change takes place** in a database, it is **called a transaction.** A **DBMS** has features that **protect data integrity.** DBMS use record locking - one user has opened a record for writing (editing), other users can only view it until the transaction is committed. This is then unlocked.

Common tools provided by a DBMS



Forms	Reports	Graphs
Data can be **input into tables** from forms or **selected data** can be **output to the screen** in a form. Forms can have **buttons** and drop down lists to make them easier to use.	Reports are **output from a database**. They can be set up to **summarise** group and select **data.** 	Some DBMS include **graphing features** so you can call them up to display data. 

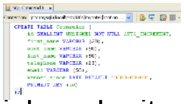
Queries



Field:	Title	Category	Last	First
Table:	Films	Films	Actors	Actors
Sort:			Ascending	
Show:	☑	☑	☑	☑
Criteria:				
or:				

These are **two ways** queries can be **created** **<u>Query By Example (QBE)</u>** uses a **graphical interface** that lets a user assemble the **fields and conditions** for a query

Query Language



```
CREATE TABLE Counselor (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
    first_name VARCHAR (50),
    nick_name VARCHAR (50),
    last_name VARCHAR (50),
    telephone VARCHAR (25),
    email VARCHAR (50),
    member_since DATE DEFAULT '0000-00-00',
    PRIMARY KEY (id)
);
```

<u>SQL (Structured Query language)</u> are **written queries**, which make it possible to write **programs to extract the data that is required**.

Query language provides operators to check conditions before selecting data to display. The most commonly used comparison operators are **AND & OR**:

- **AND operator** – checks that the two conditions are true then selects the data that matches these conditions.
- **OR Operator** – checks that either of two conditions are true then selects the data that matches these conditions.
- **NOT Operator** - excludes results**.**

Selecting a field – SELECT, FROM, WHERE

A typical SQL instruction for extracting a data set is shown below

'SELECT' which means a **set of records** is going to be **extracted** from the database.

'FROM' which defines **which parts of the database** is used, .It is **stated** by **database and field** - 'MyDatabase'.

```
SELECT * FROM MyDatabase.Names WHERE 'First_Name' = 'John'
```

The star ***** means **'every field** in the **record'**.

'WHERE' clause - only a **sub-set of the table**(s) to be **extracted**. In this case the field called **'First_Name'** where all names **equal 'John'**.

Creating a table – CREATE TABLE ((0))

A table can be created in SQL code using the following template:

```
CREATE TABLE tablename
(column1 datatype,
column2 datatype,
column3 datatype);
```

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90
07	Top Gear	Entertainment	60

- **the name of the table - "tablename"**
- the title of each **field** - **"column"**
- the **data type** that is required for each field - "datatype" (e.g. **character strings** - **'varchar'** or numbers - 'int')

```
CREATE TABLE Programmes
(ID int(2),
Title varchar(20),
Genre varchar(20),
Duration int(3));
```

The data type **varchar** indicates that only **character strings are allowed**, and the number in brackets indicates the **maximum number of digits** for each data type.

```
SELECT * FROM Programmes
WHERE Genre='Entertainment’ AND Channel='BBC3';
```

This would return the programme 'The Voice'.

Always remember the ‘’ after equals

Programme less than 20 minutes long or nature programme.

```
SELECT * FROM Programme
WHERE Duration<20 OR Genre='Nature';
```

This would return the programmes "Dick and Dom" and "Wild Brazil".

Programmes by duration.

```
SELECT Programmes.Duration, Programmes.Title
FROM Programmes
ORDER BY Programmes.Duration;
```

- the **SELECT** statement states **which fields** to look at **- Title and Duration**
- the **FROM** statement states **which table to look at - Programmes**
- the **ORDER BY** statement sorts **Duration field in ascending** order by default

Duration	Title
25	Blue Peter
30	EastEnders
50	Newsnight
60	Wild Brazil
75	The Voice
90	Sherlock

WHERE statement also used to isolate one or several records

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90

```
SELECT Programmes.ID, Programmes.Title,
Programmes.Genre, Programmes.Duration
FROM Programmes
WHERE ((Programmes.Title)="The Voice");
```

The **WHERE** statement **always in brackets**

This table shows the results from this query:

ID	Title	Genre	Duration
03	The Voice	Entertainment	75

Wildcards - *, WHERE

The **Wildcard** uses * **symbol**, used in place of any number of **unknown characters**.
E.g., the **following code** searches for **all programmes** with the **letter** I in the title:

```
WHERE ((Programmes.Title) LIKE "*i*");
```

This table shows the results from this query:

ID	Title	Genre	Duration
02	Newsnight	Current Affairs	50
03	The Voice	Entertainment	75
05	Wild Brazil	Nature	60

Adding records – INSERT INTO, VALUES

To **add new data** you use the function **INSERT INTO**.

```
INSERT INTO Programmes
VALUES (07, "Top Gas", "Entertainment", 60);
```

Note that **quotes** surround **string entrie**s. **Numbers do not** have **quotes**.

After inputting this code the table would look like this:

ID	Title	Genre	Duration
07	Top Gas	Entertainment	60

Editing records – UPDATE, SET, WHERE

SQL also has the **UPDATE** function for editing data.

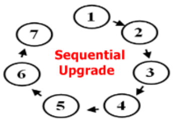
1. **Identify the table** to be updated using **UPDATE -** UPDATE Programmes
2. **Identify what the field** needs to be changed to using **SET -** SET Programmes.Title = "Top Gear"
3. **Identify which record** needs to be updated using **WHERE -** WHERE Programmes.ID = 07

```
UPDATE Programmes
SET Programmes.Title = "Top Gear"
WHERE Programmes.ID = 07;
```

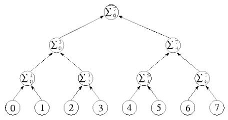
The amended table will look like this:

ID	Title	Genre	Duration
01	EastEnders	Drama	30
02	Newsnight	Current affairs	50
03	The Voice	Entertainment	75
04	Blue Peter	Children's	25
05	Wild Brazil	Nature	60
06	Sherlock	Drama	90
07	Top Gear	Entertainment	60

Different ways to soft out data



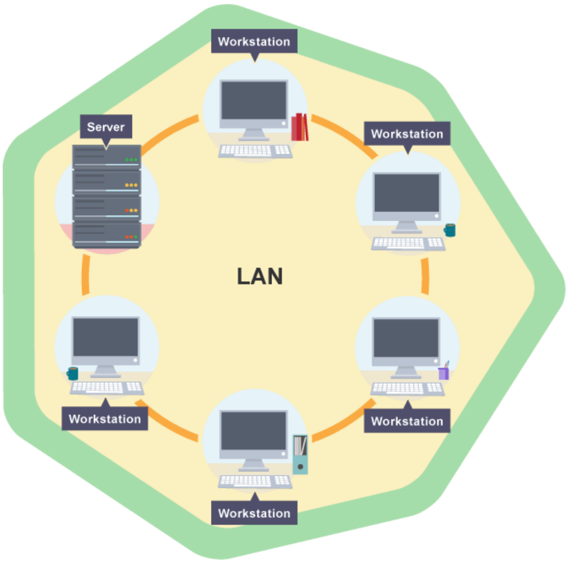
- **Sequential** data runs in a **sequence** slow one record at a time (**dictionary**)
    - **Index - Data structure - groups records** A-E, F-J, K-O, M-Q, R-V, W-Z.
- **Binary search** (Divide and conquer) – Find the middle

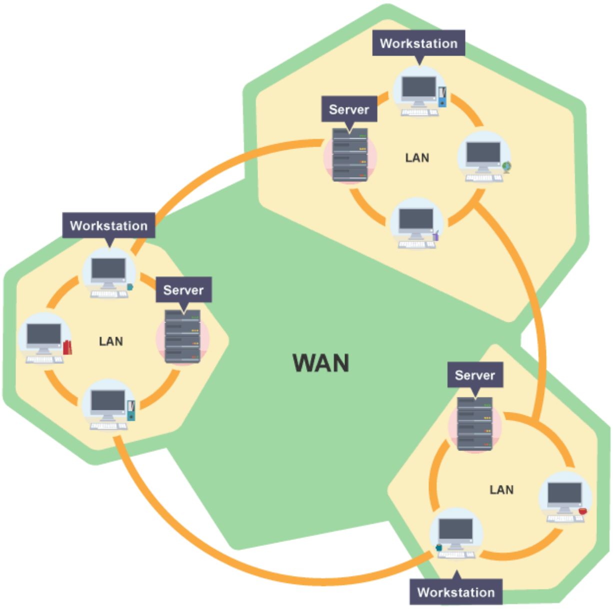


6. Computer communication and networking

A **network** is a **collection of computers**. Each **device** on a network is a **node**.

- Computers **process and store data**
- The data is **represented** as **binary** as it is very **simple** way to **store data**.
- **Binary** data is **easy to transmit** without errors, because distinguish of **0 and 1.**





LAN	WAN
▪ LANs **cover one site** (office, university campus) ▪ **Connected and maintained** (outsource) using company owned equipment	▪ WANs covers a **large area** (city, country) ▪ This could be a **network of school computers** ▪ Not all WANs might meet **local needs**
Advantage	Advantage
✓ **share** devices - laser **printer fileserver** can be used to **share documents** and files centrally ✓ **Email** - sent **between computers** ✓ **Centrally managed** e.g. one copy of word for all workstations	✓ **Wider geographical coverage** - for accessing files and sending emails. ✓ **Messages** can be **sent very quickly** ✓ **Everyone can use the same data**- older information cannot be rewritten
Disadvantage	Disadvantage
✗ **Security** due to **authorised access** ✗ **Networks** are **difficult to set up** and need to **be maintained** by technicians. ✗ **One down all done** file server is down, all users are affected.	✗ **Security** - against hackers and viruses. ✗ **Lot of maintaining** - network supervisors and technicians to be employed. ✗ **Expensive and complicated**

A VPN (virtual private network) - hosted **securely on another network**.
VPNs are often used when working on secure information **(company or school).**

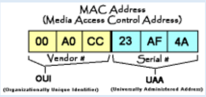
A WPAN (wireless personal area network) - connect devices (Bluetooth)

Network hardware

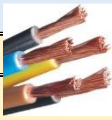
This is to do with **creating receiving and routing** electrical **signals**.

Network Interface Controllers/cards (NICS) [LAN Adaptor]

Each device on a network needs an **NIC** (LAN adaptor) → Most LANs make **use** of a **network standard** called **Ethernet**
Every **NIC** has a **unique number** stored in its **ROM** → MAC address → allows **node** on the network to be **identified**.
The **NIC generates and receives** suitable electrical **signals**.
It can also carry out **simple addressing** by making use of **MAC addresses**


MAC Address
(Media Access Control Address)
00
A0
CC
23
AF
4A
Vendor #
Serial #
OUI
UAA
(Organizationally Unique Identifier)
(Universally Administered Address)

Cables



LAN made with **copper cable**
→ **light and flexible** which makes it **easy to install.**
WAN made with **fibre-optic cable**
→ **transmits** signals by using **light waves**

Fibre-optic cables can carry **more signals** for their size than **copper cables** and are **cheaper too**.

Hubs

Hardware device that connect **many network devices together (BT Hub)** making them in a **single network segment**.

Switches

Switches connect network **segments** or bridges with other networks.
A **switch differs** from a **hub** by **transmitting a message only** to the **device** intended instead of **all connections**

Wireless Access Points – Standard WIFI

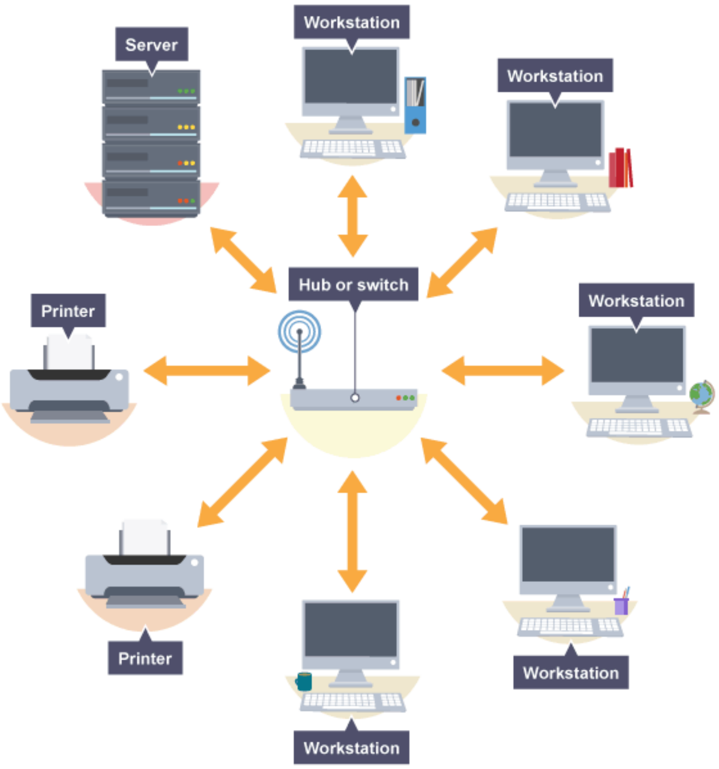
Access points - connected to router → **save money**/effort nodes are **wireless**

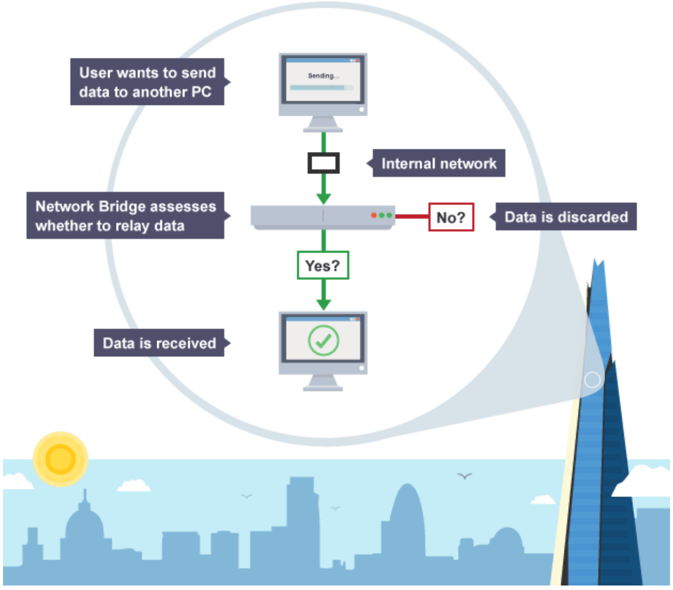
Introduce **security risks** → **encryption**, hiding broadcast and MAC Addresses.

Routers

Router receives data → form of **packets** and **forwards** them to their stop → **another router**.

Routers **direct traffic** through large networks, notably the internet or Small routers connect individual computers to the (ISP).




User wants to send data to another PC
Sending...
Internal network
Network Bridge assesses whether to relay data
No?
Data is discarded
Yes?
Data is received

Types of network

Networks are organised in two principal ways.

Client-server network

Most common as one or more servers provide service to many client machines

Servers are computers - set up to **handle network functions**. Servers include:

- Database servers which store the corporate database/
- Game servicers – which provide online gaming access
- Web servers – which holds a website

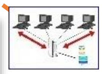
Servers do not have to be separate – They can be a **virtual server**, where one **physical machine** can take on **multiple function**. Backup → main server

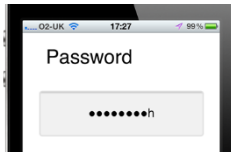
Advantage

Disadvantages

- ✓ **Easy to modify** - many services that can also be used by multiple users.
- ✓ **Security** is more **advanced** – multiple password profiles
- ✓ **Access** and resources **better** - appropriate permissions may enter

- ✗ **Expensive start-up** - you have to pay for the start-up cost.
- ✗ **One down all done** - server crashes all the computers become unavailable.
- ✗ **Same thing – longer** - When everyone does the same thing, it takes

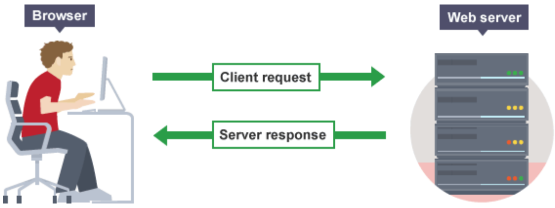




O2-UK 17:27 99%

Password

••••••••h



Peer to peer network

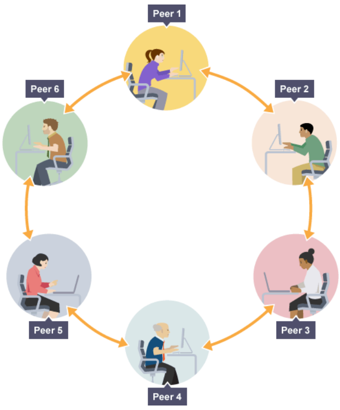
All **computers** are **equal**. **No single provider** is responsible for being the server. **Each** computer stores files and **acts as a server. (Bit torrent).** Delete easily

Advantage

Disadvantages

- ✓ **Cheaper** – as it's all users that are needed
- ✓ **No Web Server -** files **can be shared directly** with **users without web servers.**

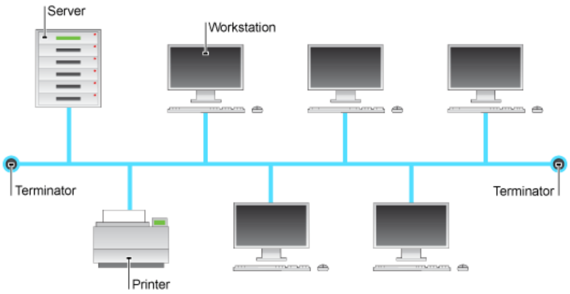
- ✗ **Maintenance** - is more difficult
- ✗ **Poor Security.**
- ✗ **Slow** – amount of multitasking



Network Topologies

Topology →layout of the **network** components, the cabling and the node positions. These are **three** principles **layouts** that you need to know about:

Bus Network



Attached to a single backbone. A **terminator** is attached at each **end** to **prevent reflection** of signals. Signals **travel** in **either directions.**

In a bus network **all** the workstations, servers and printers **are joined to one cable** (the bus).

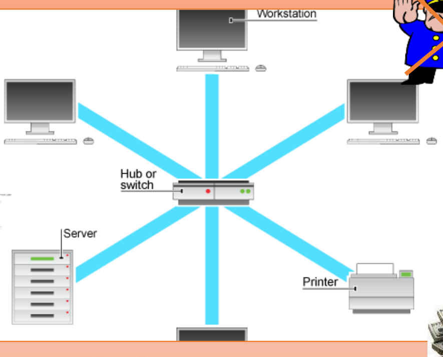
Advantage

✓ **Cheaper** – as it's all users that are needed
✓ **Cheap to install – Not much cables**

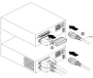
Disadvantages

✗ **Whole network down** - main cable fails
✗ **Slow performance** - data collisions every workstation on the network
✗ **Data risk** – "sees" all data on the network

Star Network



Connected to a **central switch or hub** than servers. **Signals** travel in **either directions.**



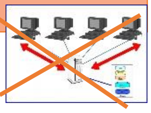
Each device has its **own cable** that **connects** to a switch or **hub**. A **hub** sends every packet of data to **every device**, **switch** only **sends** a packet of data to the destination device.

Advantage

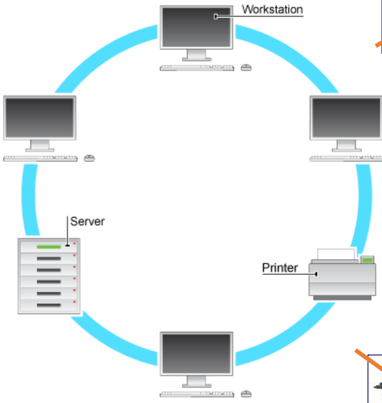
✓ **Reliable –** if one **cable** or device **fails** then **all others** will **work**
✓ **High performing - no data collisions**

Disadvantages

✗ **Expensive install** – lots of cables
✗ **Extra hardware** – hubs & switches
✗ **Hubs/switches must work** – network fails



Ring Network



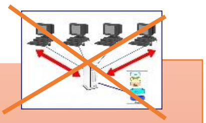
Data passes through **each node**, carried in data units in **one way**, which **prevents collisions**.

Each device (workstation, server, and printer) **is connected to two other devices**, like a ring. Each **packet** of data travels in **one direction** and each device receives each packet, **until** the **destination** device **receives it**.

Advantage

✓ **Quick transfer –** even if large no. of devices – one direction

Disadvantages



✗ **If one fails than all fail – one breaks down, all break down**

Network Technicalities

Protocols (TCP/IP)

Widely adopted Protocols **TCP/IP (Transmission Control Protocol/Internet Protocol). TCP/IP** de facto standard for **data transmission** over the **internet**.

Hosts

Hosts are **computer systems** that accessed **remotely** and **hold data** or other **facilities** such as **web servers**. **TCP** is **concerned** with the **host connections**. It is **not concerned** with the **nature** of data **being sent**.

TCP/IP (transmission control protocol/internet protocol)

TCP/IP (also known as the internet protocol suite) is the set of protocols used over the internet. It **organises** how **data packets are communicated** and makes sure **packets have the following information**:

- **source** - which computer the message **came from**
- **destination** - where the message should **go**
- **packet sequence** - the **order the message** data should be re-assembled
- **data** - the **data** of the message
- **error check** - the check to see that the message has been **sent correctly**

Packets

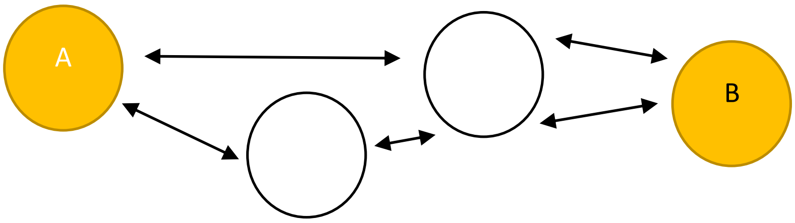
Packets are **collections of data** forming **part of a message**. A packet is **constructed** according to rules **laid down** by the **appropriate protocol**. A packet contains a standard field such as:

- **Protocol**
- **Checksum:** number is checked to ensure packet has not been corrupted
- **Total length**
- **Senders & Address**
- **Time to live** – if not delivered it needs to be destroyed
- **Packet number** – This is used to reconstruct the message in the original order
- **Data** - the part of the message that is being constructed.

IP is **concerned** with the **construction of the packets**

Packet switching

Packets sent **individually** across a network. Packets may **take different routes** according to **availability** and **traffic** conditions. They **assembled** from the **complete message** at the **receiving end**. This process is called **packet switching**. It improves the **reliability** if some route is down or congested, another can be found.



Protocol	Meaning	Application
DNS	Domain Name System	Translates domain names, such as **ocr.org.uk** in to **IP addresses**
TSL/SSL	Transport Layer security/Secure socket layer	Designed for **secure communication**
FTP	File transfer protocol	From **copying files** from **hosts**
HTTP	Hypertext transfer protocol	For distributing **hypermedia file** as – almost web pages
IMAP	Internet message access protocol	One **method** for accessing **emails**
POP3	Post office protocol (VER 3)	Another method for **accessing emails**, used by most **webmail servers**

IP Addressing

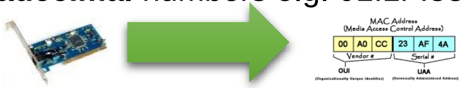
A **letter** sent through the **post needs an address** in order to be **delivered correctly**. Similarly, **a data packet** in a network needs an **address for delivery**. **Each computer** on a network has an **IP address → 32** binary numbers.

10000011 01101011 00010000 11001000

Each group of 8 bits is called an octet can store numbers ranging from **0 to 255.** The **address** is **quoted** in **four groups**. For example, 131.107.32.200. **IP addresses** can be **permanently allocated** to a **device** *(static addressing),* but they can also be **temporally allocated** *(dynamic addressing),* so computers will not always have the same IP address.

MAC Addressing

MAC means Media Access Control. A MAC address is a **unique number stored in each NIC →** used **to identify a device** on a **network**.
A **MAC address** given **six pairs** of **hexadecimal** numbers e.g. 01:1F:33:69:BC:14.


MAC Address
(Media Access Control Address)
00
A0
CC
23
AF
4A
Vendor #
Serial #
OUI
(Organizationally Unique Identifier)
UAA
(Universally Administered Address)

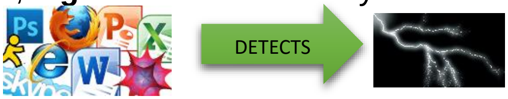
Network Security

Data can be lost by **unauthorised access -** Data loss, Theft of data & malware
Data can be lost by **accident** – fires, malfunctions, wars, terrorism hazards.
Data should be copied to a **secure facility off site**.

Archives

Archived data is **old data** that is **no longer** in regular **use**, it **used** for In case of **further enquires**, **legal reasons** – when you leave school.

Failover



Software detects a likely **disaster** or abnormality and immediately **transfers operations** to a **duplicate system**.

The internet

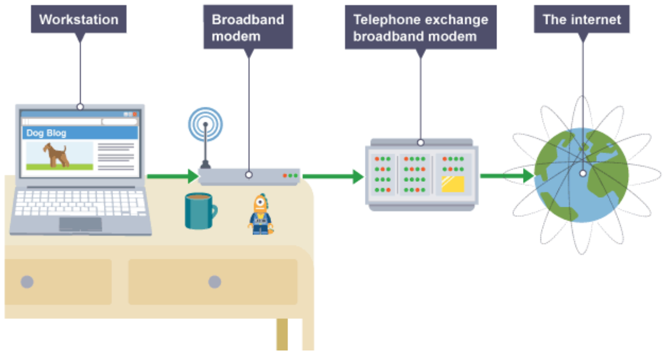
Broadband internet is transmitted on physical wires that run underground/ocean (BT)
Most people use the internet to view WWW web pages, emails & communication.
Download speeds tend to be **faster** than upload speeds → More demand
Network speeds - megabytes per second (MBps)

Hardware

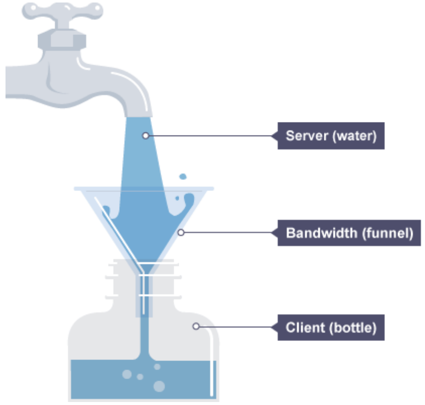
To **connect to the internet**, some **specialised hardware is needed**.

- **Modem** – phones are analogue. Computers are digital signals. A modern **converts** between **these signals** and allows **connections**.

- **Router** – connect **networks together**. **Digital Subscriber Line (DSL) routers** are **combined** with a **modem** in order to **make use of unused bandwidth** in telephone line, ADSL – Subscriber cable

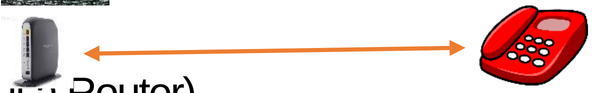


Broadband connections



The **speed** that **data** can be **transferred** is dependent on a number of factors:

- Signal quality can vary between phone lines.
- The distance between the modem and the telephone



3G and 4G – (Wi-Fi without a Router)

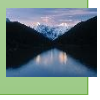
Transmitted through **network** rather than the **physical cabled network** of broadband. Anyone can connect to the internet as long as there is a 3G or 4G

Advantages

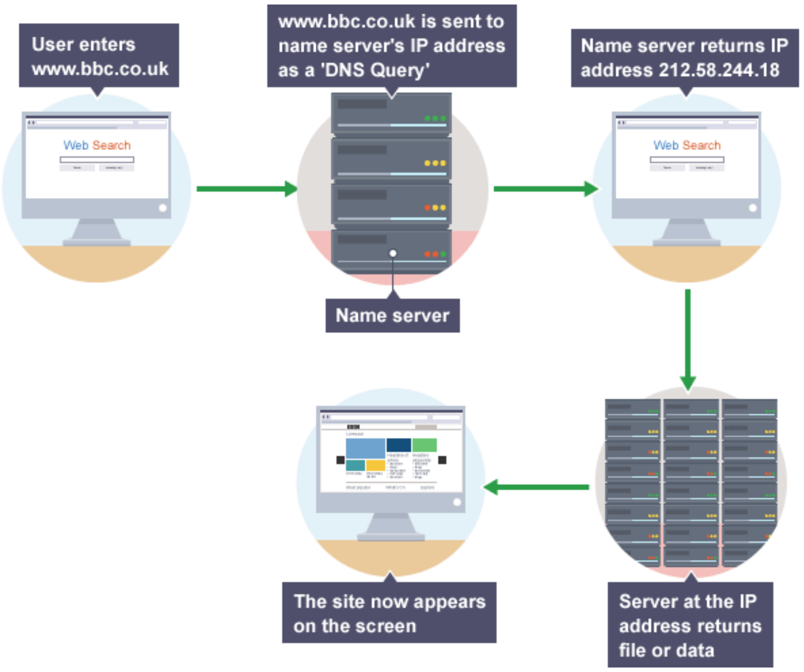
- it provides an internet connection on the move
- there is the ability to transfer data fairly quickly

Disadvantages

- it can be expensive to download data
- some areas don't have 3G or 4G connections



Search engines


User enters www.bbc.co.uk
Web Search
www.bbc.co.uk is sent to name server's IP address as a 'DNS Query'
Name server
Name server returns IP address 212.58.244.18
Web Search
Server at the IP address returns file or data
The site now appears on the screen

A search engine - **huge web database** of web addresses.
Web pages are stored as an **index on a server.**
The **index** has already been **put together** by an automated program called a **crawler that is run by the search engine program**. **The crawler** frequen **web sites** and **takes a record of the address** and **keywords** and **adds** this information to a **database with an index** → connected through hyperlinks.

Addressing

IP addressing are used to identify connected resources. **(DNS)** is a **protocol** that connects IP Addresses (212.58.253.67) to user-friendly names (bbc.co.uk).

DNS servers maintain **databases** that **match** the **names** against the **numbers**.
A **URL** is a **Uniform Resource Locator** - a resource on the internet.

Internet domain names are split up like this: www.bbc.co.uk/computing/ipaddresses

- The **domain** name	www.bbc.co.uk
- The **path**	/computing
- The **web page** required -	/ipaddresses

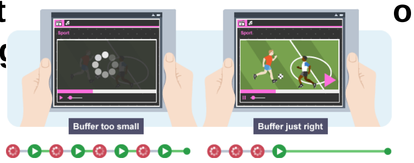
Streaming

Streaming allows **data** to be **used immediately** but the **whole file is not downloaded**.

- Data is **streamed** by the **service to the client** - Web browser, app etc

Buffering

- A buffer is a **temporary storage** space where **data** can be **held and processed**. The **buffer holds** the **data** that is required. As data for a file is **downloaded**, it is **held** in the **buffer t** **ough data is in the buffer**, the **file** will start **playing**



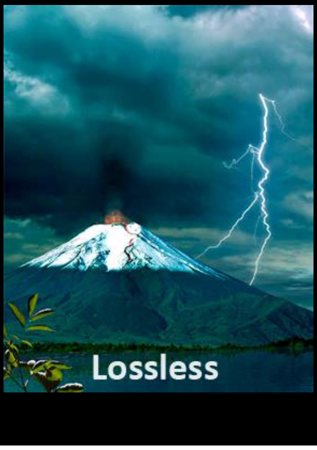
Compression

Files compression means **reducing the size of a file**. This is done in order to:

- **Save storage space** on device media
- **Reduce transmission times** on a **network**, especially the internet

One **disadvantage** of using **compression** → slower operations.

There are two main types of file compression

Lossy Compression	Lossless Compression
Lossy Compression works by **removing some of the data** from the file. **The data removed cannot** be **recovered**. **Lossy compression** is used in **file formats** such as **MP3, JPEG** – good for websites. The **more an image is compressed**, the **less details** will be visible. **Lossy compression** algorithms **often attempt to remove** the **least important details** such as the **highest frequency sounds in a music** file that many people cannot hear. **Lossy compressions** can produce **dramatic savings** in file **size**. 	**Lossless Compression** does not **store repeated details** such as **lots of blue pixels** in a photo that includes the sky. REPEAT NO REPEAT It allows the **original file to be reconstructed** exactly. A **computer program** will **not work if** much of it **has been removed** to save space. 

Here is one way lossless compression can be used.

A table of details such as words or pixels is set up in order to store the data, the table is stored plus an index for each word whine it occurs.

1	2	3	4	5	6	7	8	9
If	you	are	not	fired	with	Enthusiasm	will	be

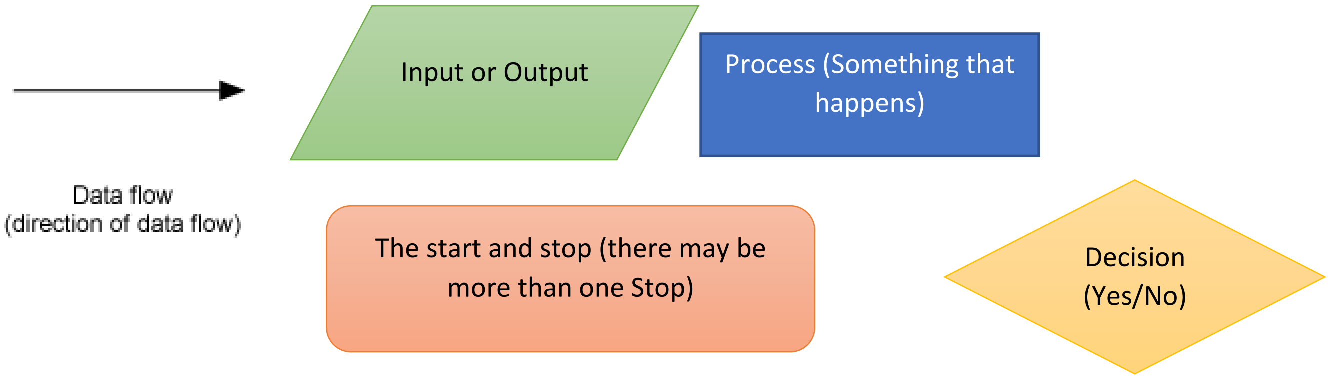
Therefore, a sentence could be written from the table such as ***1234567289567*** aka if ***you are not fired with enthusiasm you will be fired with enthusiasm***

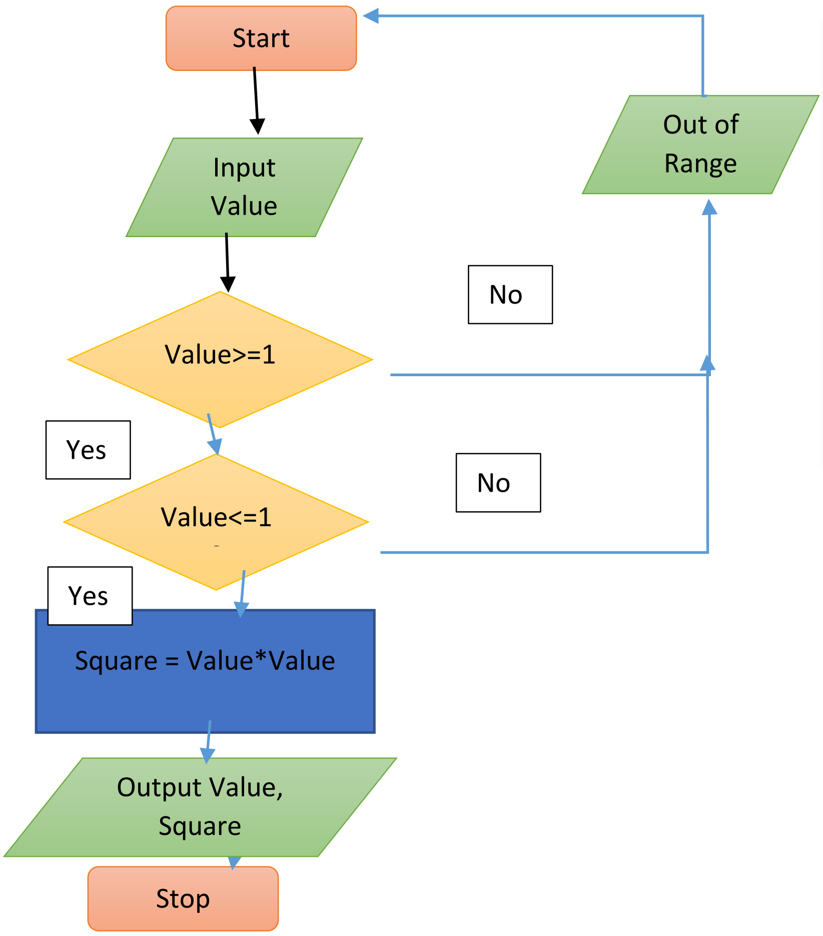
7. Programming

Algorithms are **sets of rules** that **define a solution to a problem**. **Algorithms can be expressed** in **many ways** but **typically** as a **flowchart** or in **pseudocode**:

Flowcharts

Display how data flows and decisions are made. The basic ones include:


Input or Output
Process (Something that happens)
Data flow
(direction of data flow)
The start and stop (there may be more than one Stop)
Decision
(Yes/No)



A simple flowchart a number between 1 to 10 and its square

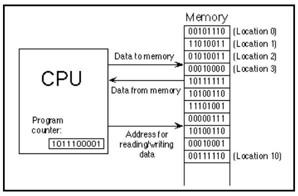
1. *Enter the number between 1 and 10*
2. *Reject values not in range and request the input again*
3. *If valid input, calculate the square of the value*
4. *Output the value and its square.*

Pseudocode

Pseudocode is data flow in structured English, For the flowchart it might be:

```
Repeat
        Input a Value
Until the value is between 1 and 10
Square = value * value
Output value and square
```

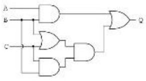
Indents are used to show that the instruction between **REPEAT and UNTIL** to show it is **repeated until the condition is met**.



Machine languages

Software has to be provided to the **processor** in the form of **machine code**:

- **Stream of binary** bit patterns that represent the **instructions** carried out.
- Binary Stream is **decoded** by the processor's **logic circuits**.
- They are then acted **upon or executed**, one after another



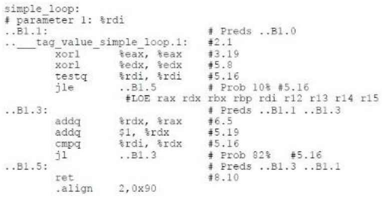
Machine code is a type of **low-level code**, which means that it **works at the level** of the **computer hardware**.

Writing programs in **machine code** is **difficult and time-consuming work**:

- Each operation of the **processor has to be defined**
- **Machine instruction causes the processor** to **carry out just one operation**.

Nearly all machine code instruction consist of two parts:

- An **opcode** – which tells the processor **what to do**
- An **operand** –which tells the processor **what to do it to**



```
simple_loop:
# parameter 1: %rdi
..B1.1:                                # Preds ..B1.0
..___tag_value_simple_loop.1:          #2.1
        xorl      %eax, %eax           #3.19
        xorl      %edx, %edx           #5.8
        testq     %rdi, %rdi           #5.16
        jle       ..B1.5               # Prob 10% #5.16
                   #LOE rax rdx rbx rbp rdi r12 r13 r14 r15
..B1.3:                                # Preds ..B1.1 ..B1.3
        addq      %rdx, %rax           #6.5
        addq      $1, %rdx             #5.19
        cmpq      %rdi, %rdx           #5.16
        jl        ..B1.3               # Prob 82%   #5.16
..B1.5:                                # Preds ..B1.3 ..B1.1
        ret                            #8.10
        .align    2,0x90
```

Assembly Language

This is also a **low-level language.**

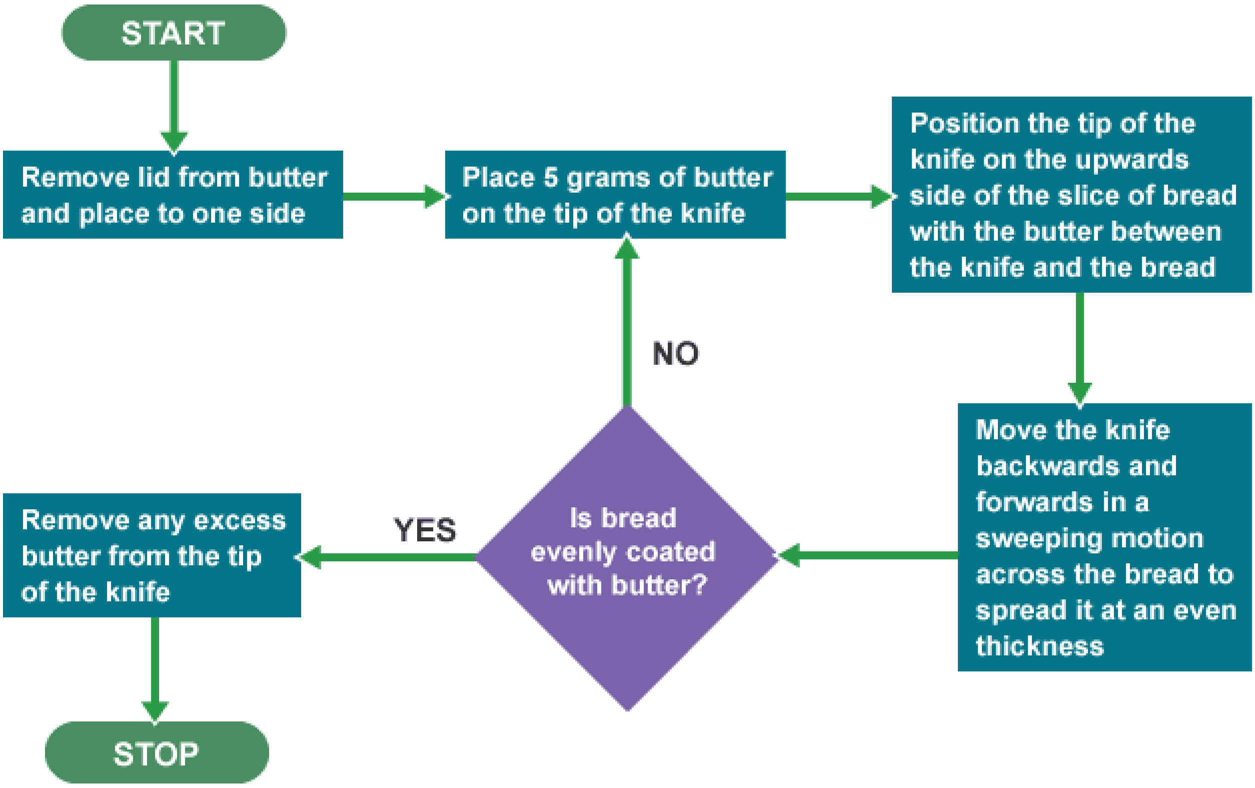
- As with machine language, each **instruction causes one** processor **operation**.
- **Assembly language** uses **mnemonics** to **represent instructions**.

In machine code, a program might write 00010 00000 00000 00000 1000 000000

Here the first group of bits is the **opcode** 000010 or 2 in denary. **This opcode could mean JUMP**, which means the program must continue at the specified address. **The operand is the remaining digits**.

With a food recipe, a simple command like 'spread butter on bread' could be made much more detailed. For example:

1. Remove lid from butter tub and place to one side.
2. Place 5 grams of butter on the tip of the knife.
3. Position the tip of the knife on the upwards side of the slice of bread with the butter between the knife and the bread.
4. Move the knife backwards and forwards in a sweeping motion across the bread to spread it at an even thickness.
5. Repeat steps 2 to 4 until one side of the slice of bread is evenly coated with butter.
6. Remove any excess butter from the tip of the knife.


START
Remove lid from butter and place to one side
Place 5 grams of butter on the tip of the knife
Position the tip of the knife on the upwards side of the slice of bread with the butter between the knife and the bread
Move the knife backwards and forwards in a sweeping motion across the bread to spread it at an even thickness
NO
Is bread evenly coated with butter?
YES
Remove any excess butter from the tip of the knife
STOP

Bubble sort algorithm example

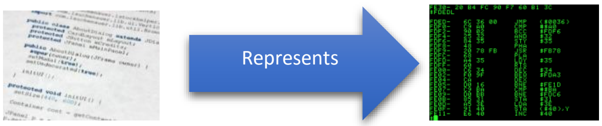
This algorithm could be used to sort the following list:

3, 2, 4, 1, 5

The first loop of the algorithm would produce:

- **3**, **2**, 4, 1, 5 (2<3 so the two values are swapped)
- 2, **3**, **4**, 1, 5 (3<4 so the two values are **not** swapped)
- 2, 3, **4**, **1**, 5 (1<4 so the two values are swapped)
- 2, 3, 1, **4**, **5** (4<5 so the two values are **not** swapped)
- 2, 3, 1, 4, 5 (**First pass completed**)

It will do it until the numbers are in order and there are no more passes.



High-level language.

A **high-level language** command **may represent serval machine code instructions** as assembler:

- **In a high-level language**, we can usually **multiply two numbers together** in one command.
- **At machine level**, that is **not possible** and it has to be **repeated addition**.

High-level commands have to be **turned into binary** instructions the machine can understand, this process is called translation.

There are **two ways** of **translating high-level code** to **machine code**:

- **Complier**: **converts** the **whole code** into machine code **before** running it.
- **Interpreter**: **converts** the **code** one instructions **at a time**, running each instruction before translating the next

Source code is the code written by the programmer.
A complier translates this source code into an object code in machine language.
Object code runs independently of the source code and translator

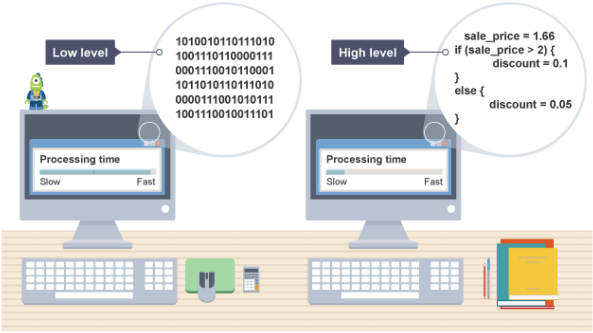
Translator	Advantage	Disadvantage
Assembler	**Precise** and **direct** instructions to the computer hardware	**Difficult** to code **Limited range** of commands available
Complier	Code **runs quickly** once complied Difficult for **others to modify no access** to the **source code**	Process can be **slow** Errors **generated at one** - **hard debug**.
Interpreter	easy to **debug** - executes 1 at a time Tested in **stages** - Code **More portable** - any machine	Interpreter **needed** on **target machine** interpreter **takes up space** in memory Code **executes more slowly**

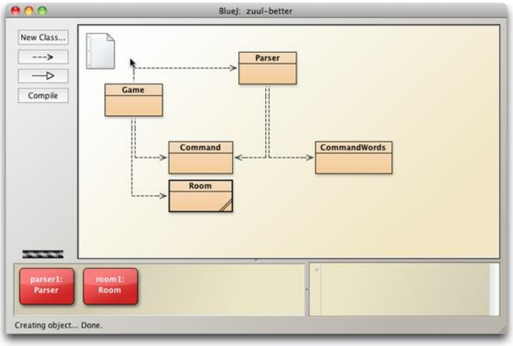
Integrated development Environment (IDE)

Translators usually **include** an **IDE** to help programmers.

IDE Features include:

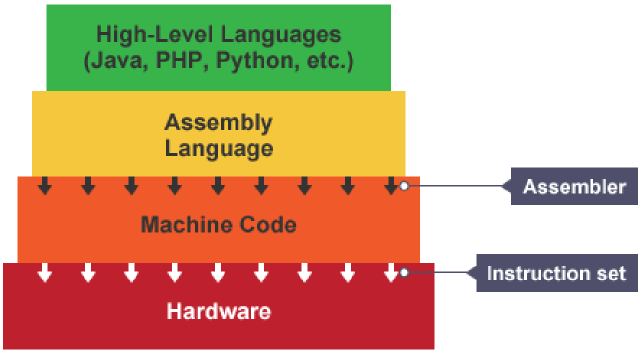
- **Source code editor** – indents & **colour coding** words, variables and comments
- Error **diagnostics and debugger** – warning identify **potential problems** with code and listing errors with the code
- **Run time environment** – allows the developer to run code during development to check for **logical errors.**
- **Translator** (complier or interpreter) – **complies** or interprets the source code in to **runnable machine code**





Assembler

An assembler translates assembly language (low-level) into machine code (lower level. Assembly language is a low-level language - reflects operations of CPU.



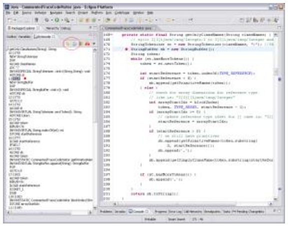
```
import com.lauchenauer.iatockhelper.
import com.lauchenauer.lib.ui.Vertic
public class AboutDialog extends JDia
  protected CardLayout mLayout;
  protected JButton mCredits;
  protected JPanel mMainPanel;
 public AboutDialog(JFrame owner) {
  super(owner);
  setModal(true);
  setUndecorated(true);
  initUI();
 }
protected void initUI() {
  setSize(440, 600);
  Container cont = getContent
 JPanel p =
```

Interpreter (before)

Interpreted languages are also called **scripting language** –web applications
An interpreter **translates code into machine code, instruction by instruction** - **Interpreted code** is slower to execute than compiled code.
Interpreted **languages** include **JavaScript, PHP, Python and Ruby**.

Mnemonics

Assembly language instructions use abbreviations called **mnemonics**. An example of a mnemonic assembly language instruction is **LDA 50**, which stores the value **50** into a register of the CPU. **Mnemonics** are **easier for humans to remember** and understand than Binary machine code instructions.



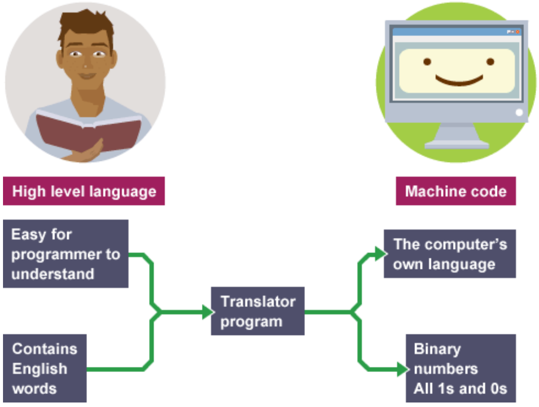
Compiler (after)

A compiler **translates the program into machine code before the programs run**.
It can be **difficult to test** individual lines compared to interpreted languages as all **bugs** are reported **after the program** has been compiled.
The **machine code** is **saved** and stored **separately** to the **high-level code**.

Java is a high-level programming language, which is **compiled to produce bytecoden, which** is then interpreted by a **virtual machine** (VM).
Bytecode is code, which is compiled and can then be interpreted.

Translators

Converts high-level languages into machine code. These translators are known as **compilers** and **interpreters**. Programs - translated using compiler or interpreter.


High level language
Machine code
Easy for programmer to understand
Contains English words
Translator program
The computer's own language
Binary numbers All 1s and 0s

Control flow in imperative language

Sequence

For example, **a program** to **input two numbers** then **output their total**:

```
Input num1

Input num2

Total = num + num 2

Output total
```

Selection

The IF-THEN-ELSE construct allows the program to take one of several paths
To determine if a person can go onto a ride in a theme park, we might use:

```
Get height

IF height >= 1.5M THEN

        Alow onto ride

ELSE

        Do not allow onto ride
```

A quiz may use CASE statement or the ELSEIF statement if there is more then two answers for example. I.e. use = or use <> or output if answer id different).

Iteration

The program completes a set of instructions several times until a condition is met. To make a program execute a set of commands several times, we can use a **count-controlled loop.**

In a **count-controlled loop**, we use an **index value** to **tell the program** how many **times to complete the loop**. For example, to print a times table

```
FOR k= 1 TO 12

        OUTPUT k "times 7 is k*7

        Next K
```

In a condition-controlled loop, we tell the **program to either stop** executing the commands when a **condition is reached** or tell it to execute the commands while a condition is true.

- **Repeat-Until** executes the **code** that follow **until a condition is true**.
- **While-End** executers the code that follows **while a condition is true**.

For example, **a program to collect numbers** from a user until they decide they have entered all the data, and output, the total values and the average of the values

```
Count = 0
Total = 0
REPEAT
        INPUT “enter number” value
        Count = count +1
        Total = total + value
        INPUT”More numbers?” more
UNTIL more <> “yes”
```

```
Count = 0
Total = 0
More = “yes”
WHILE more = “yes”
        INPUT “enter number” value
        Count = count +1
        Total = total + value
        INPUT”More numbers?” more
ENDWHILE
```

The condition in a **repeat loop is checked** at the end of each pass. **In a while loop** it is **checked before the first pass**.
A repeat loops always executes at least once.

Variables and Datatype

Data types such as **REAL** are used to represent a **decimal number** in a program for instance. This can be used to describe the variable that goes in front it in a program. They are **declared at the start** to **avoid the danger** of any **data already sorted** in that location by a previous program. Being used and giving false results

Operations & comparison operators

These are mathematical operations that program can perform to variables

Operator	Name	Example	Comment
MOD	Modulus	Value= num1 MOD num 2	Return the **remainder part** (modules) IF num1 = 23 and num2 = 5 value = 3
DIV	Quotient	Value= num1 DIV num 2	Returns the **number part** (Quotient) IF num1 = 23 AND num2= 5 Value = 4
<>	Not equal	UNTIL question <> yes	This means is not equal to

Operator Priority

The order in which operators are applied can be important. The priority for the operations is

1. Operations inside brackets are dealt with first
2. Unary operators such as minus sign and NOT
3. Multiplication and division
4. Addition and subtraction
5. Boolean operators such as AND & OR
    a. 5*(7-3) means 5 x 4 = 20
    b. 5*7-3 means 35-3 = 32-

Arrays

If we need a number of variables all with the same name then we use an array. An array is a set of variables with the same name (or identifier) and an index number to identify the different variables.

For example, a set of names for 20 people could be stored in the array names using name (1), name (2), name (3) etc. up to 20 names. Sometimes arrays start with 0.

0	1	2	3	4	5	6	7	8
Ash	Gary	Brock	Misty	Surge	Sabrina	Erika	Blane	Giovanni

Errors

- Syntax errors – problems with the code that is written
- Logic errors – conditions that cannot be met, infinite loops, incorrect expressions. This can lead to run time errors.

Arrays in Python

A Python array → uses 'list' instead of array - more flexibility different types & lengths
It is often convenient to use no need to know datatype.

Simple programmable computers.

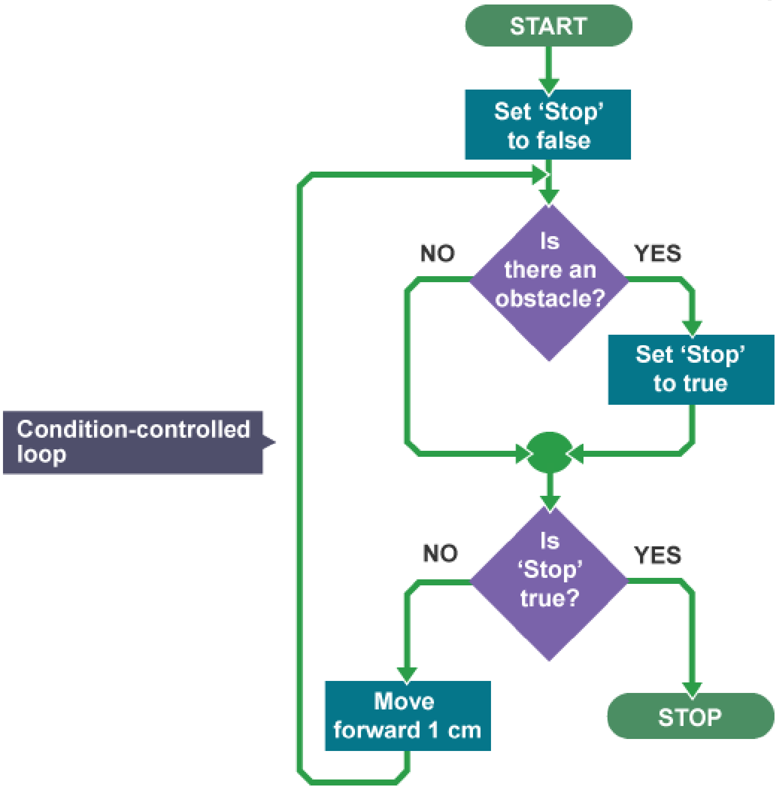
The Raspberry Pi

The Raspberry Pi → **card-sized**, the Pi is basic machine has to be **programmed**
It runs **Linux**. It can use **keyboard & mouse**. Programmed using **Python**.
Light and portable, it can be used for computing projects outside.

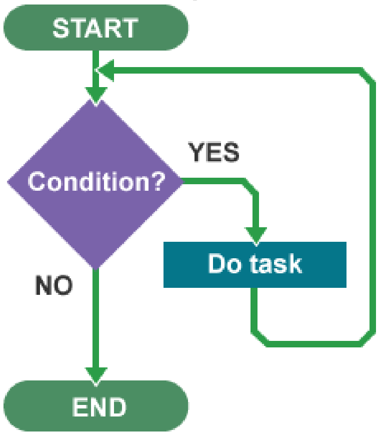
Arduino (Open Source)

The Arduino → **simpler device than Pi** - makes interactive devices **sense light** or respond to switches. The Arduino contains a **CPU & memory** within one chip.
It is programmed by uploading a program through a **separate computer via USB**.

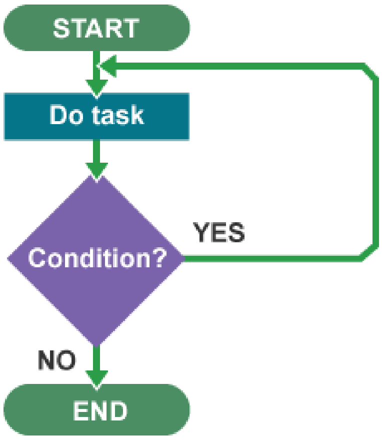
Count-controlled loop



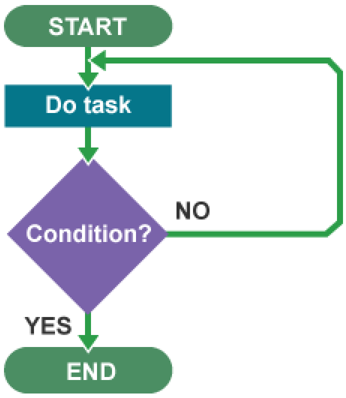
WHILE loops



DO WHILE loops



REPEAT UNTIL loops



Data Structures

- A **static** structure is **fixed in size** but a **dynamic** structure can grow or shrink.
  An array - static structure - size is fixed. A list - dynamic structure – size can change
- A **mutable** structure – data - edited, deleted or moved,
  immutable structure – data - cannot be changed → more data added.