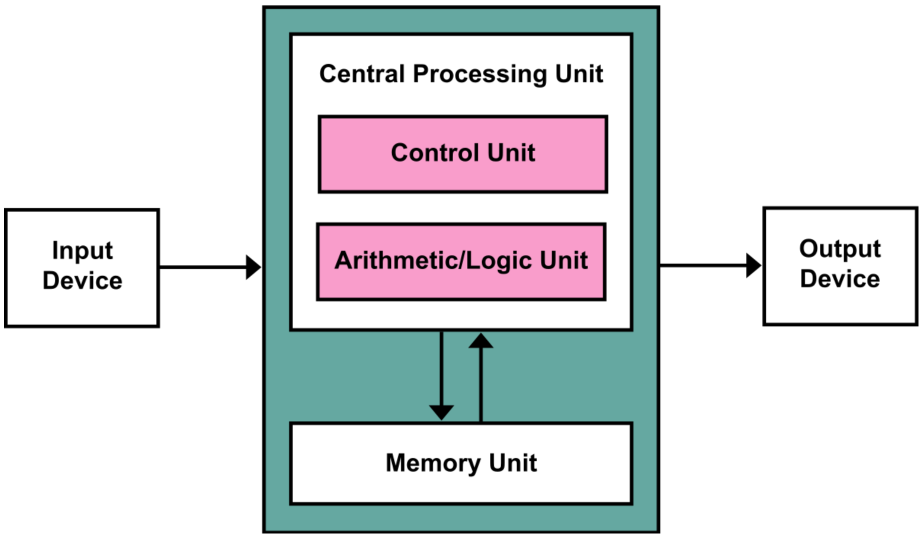
GCSE Computer Science Revision and Workbook

Read, make notes, test yourself using the end-of-chapter questions.

Click the links below to take you to that part of the document. You might have to scroll up or down a little bit to get to the start of the section.

Systems Architecture

The Von Neumann Architecture refers to any computer system that consists of input and output devices, with memory and a central processor. It also refers to both the instructions and the data being stored in RAM together. The processor can tell the difference between instructions and data by knowing how long instructions are (how many bits in length) and that the data must come after the instruction. Almost all modern computers are based on the Von Neumann architecture.

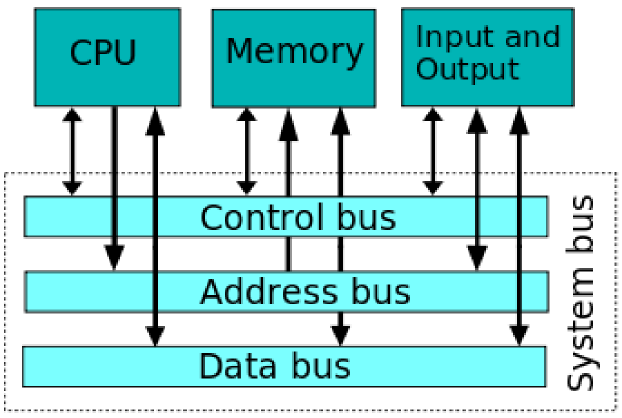


Other architectures store the instructions and data in separate pieces of memory – *weird huh?*

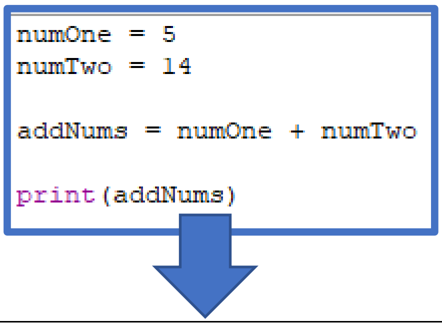
Computers require a way on inputting data, processing data, storing data and outputting data. Internal components are links up via the system bus.

The system bus is a set of parallel wires that carry data and control signals between components connected to the motherboard. The system bus consists of three parts:

- Control bus – carries control signals from the CPU's Control Unit to other components (e.g. RAM)
- Address bus – carries the address of instructions to be fetched from RAM.
- Data bus – carries the actual instruction and data from RAM to the CPU for processing.



Computer programs consist of many instructions. For example; a short block of Python code such as the program below might contain thousands of instructions.



```
numOne = 5
numTwo = 14

addNums = numOne + numTwo

print(addNums)
```

This might be stored in RAM as:

```
01010101 11010111 00001110 00011010 10101010 00110011 00110101 00010001
10001000 10010101 01101010 00000000 11010111 11111111 01011101 00100101 01010101
11010111 00001110 00011010 10101010 00110011 00110101 00010001 10001000
10010101 01101010 00000000 11010111 11111111 01011101 00100101 01010101 11010111
00001110 00011010 10101010 00110011 00110101 00010001 10001000 10010101
01101010 00000000 11010111 11111111 01011101 00100101 11101010 00000000 00000000
```

Instructions are commands that a CPU can recognise and process.

The instruction set contains all of the instructions that a CPU can understand. When the CPU decodes and instruction, it looks up the opcode (the command part of the instruction) in the instruction set to figure out what to do with it.

For example, if the CPU fetched 01010110 out of RAM and then decoded it using the (very small) instruction set below, it would see that the opcode is telling it to ADD a number.

CPU Instruction Set	
Opcode meaning	**Machine code**
INPUT	11110101
OUTPUT	11110001
STORE	11111110
ADD	01010110
SUBTRACT	00110111

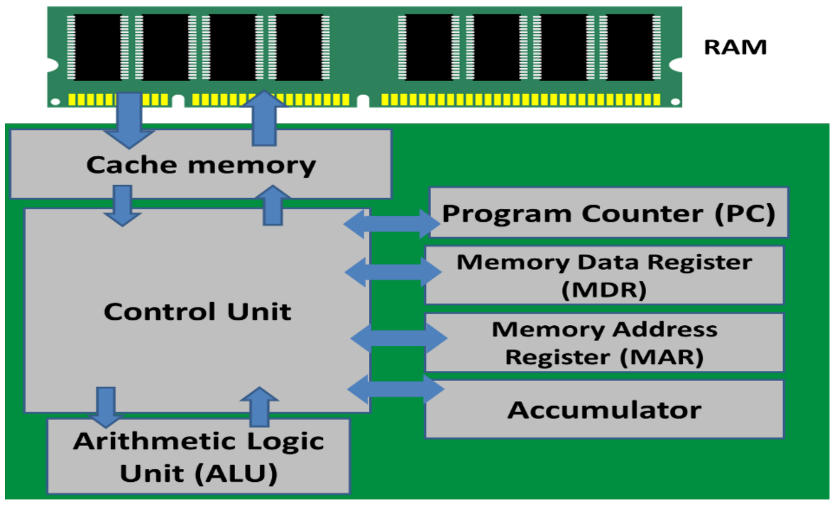
Instructions are made up of two parts; the opcode and the operand. The opcode is tells the CPU what to do and the operand is the piece of data that the opcode is performed on. For example:

INSTRUCTION = **1001**0011

OPCODE: **1001** OPERAND: 0011

The Central Processing Unit (CPU) fetches, decodes and executes instructions.

The CPU consists of register (which are temporary stores for instructions being worked on), the cache, the Control Unit and the ALU.


RAM
Cache memory
Control Unit
Program Counter (PC)
Memory Data Register (MDR)
Memory Address Register (MAR)
Accumulator
Arithmetic Logic Unit (ALU)

CPU Registers (temporary stores for instructions being processed):

- Program Counter – keeps track of where the CPU is in the program. Points to the next instruction in the cycle.
- Memory Address Register (MAR) – holds the address of the instruction to be fetched.
- Memory Data Register (MDR) – stores the instruction about to be executed.
- Accumulator – stores the most recent result of processing.

Main CPU components:

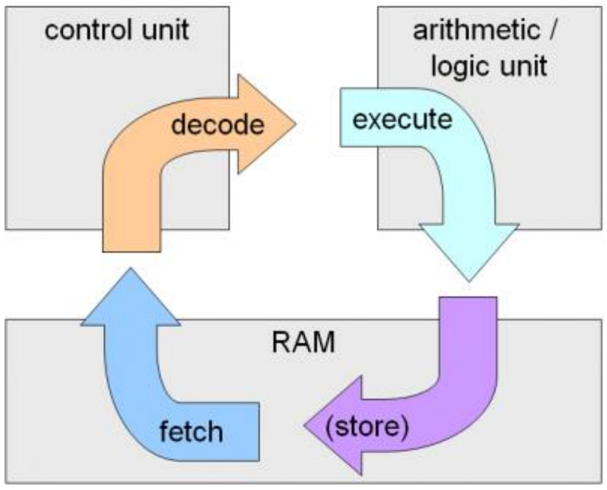
- Control Unit (CU) – controls the fetch, decode, execute cycle. Sends control signals to CPU components and timing signals to coordinate the CPU.
- Arithmetic Logic Unit (ALU) – Carries out calculations and logical comparisons (e.g. greater than, less than).
- Cache – High speed memory, very close to the CPU, stores regularly used instructions for fast access.

The CPU instruction cycle:

When a program is loaded, all the program's instructions are placed into RAM.

1. The Program Counter points to the RAM address of the first instruction.
2. The Control Unit tells the Program Counter to place the RAM address into the Memory Address Register (MAR).
3. The Memory Address Register sends the address down the address bus and locate the instruction in RAM.
4. The instruction travels from RAM, down the data bus and into the Memory Data Register (MDR).
5. The instruction is then decoded by breaking it into two parts (opcode and operand). The opcode is looked up in the instruction set so that the CPU can understand what to do with it.
6. The instruction is then executed by the Arithmetic Logic Unit (ALU) which carries out any calculations and logical comparisons.
7. The result is stored in the Accumulator.
8. The CPU then fetches the next instruction, and so on!

Note: While this has been going on the Program Counter has incremented, ready to point the MAR to the next instruction to be fetched.

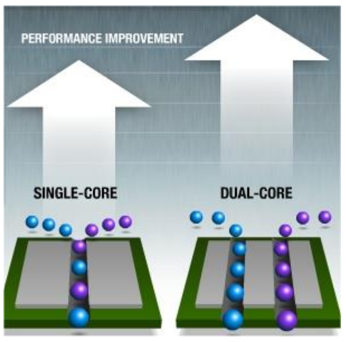


The CPU clock keeps strict timings that the Control Unit uses to keep control over the fetch, decode, execute cycle. The clock speed is the number of instruction cycles per second; that is the number of times the CPU can fetch, decode and execute and instruction per second. One instruction cycle = 1 Hertz (Hz). Modern CPUs often run at over 3 GHz (3 Billion cycles per second).

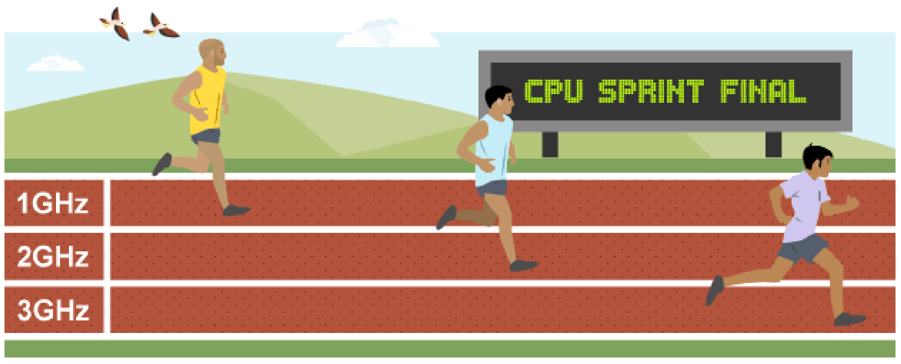
The higher the clock speed, the faster a CPU can process data.

Older CPUs only contained a single core, which meant that it could only process one task at a time. Obviously, most people want to have multiple programs and files open at the same time. CPUs got around this by multitasking. Multitasking is when a CPU rapidly switches between programs, giving a few clock cycles to each program to create the illusion that it is running them at the same time. The user does not usually notice this, but it can cause the computer to slow down if too many programs are open.

Most modern CPUs contain multiple cores. This is like having more than one CPU to process data. More cores mean the CPU can run more programs at the same time.



In theory a dual-core CPU could process two programs at once, or process twice as fast. Although in reality, other factors prevent it from being twice as fast. But more cores definitely means better performance.



Systems Architecture Exam-Style Questions:

1. State what is meant by the Von Neumann Architecture.
2. State the three components of the system bus.
3. State the two parts of an instruction.
4. Describe the difference between an opcode and operand.
5. State the three stages of the CPU instruction cycle.
6. Describe how the instruction set is used to decode opcodes.
7. Describe the function of the CPU's Control Unit.
8. Describe the function of the ALU.
9. Explain how clock speed and the number of cores affect the CPU's performance.

Embedded systems:

Embedded systems are computer systems with a dedicated function within a larger system. For example a computer system that controls a washing machine is an embedded system. The computer chips and sensors inside a car are embedded systems.

Industrial Robots
GPS
GPS Receivers
Digital Cameras
DVD Players
Wireless Routers
Embedded Systems
MP3 Players
Set top Boxes
Gaming Consoles
Photocopiers
Microwave Ovens

Embedded systems only need a small amount of secondary storage (usually solid state) because they only carry out specific functions. They do not need to store lots of large programs or an operating system such as Microsoft Windows. Embedded systems do not need much primary storage (RAM) because they only run one program that performs a dedicated task.

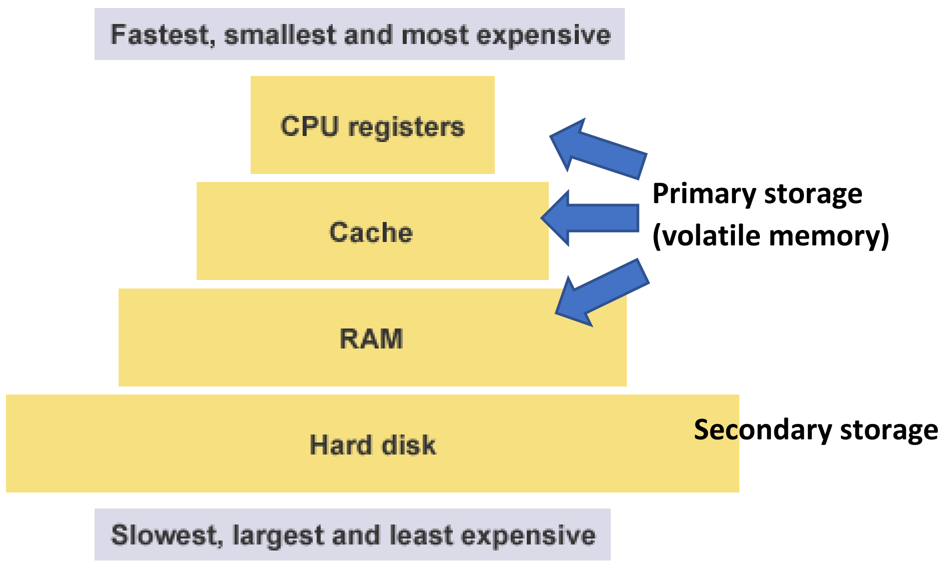
Embedded systems usually have a CPU with a clock speed measured in MHz (millions of cycles per second) rather than GHz (billions of cycles per second) because they do not have to multitask or process as a huge amount of data.

Another reason for "low spec." components is cost (why spend money when you don't need to?) and power consumption. Embedded systems quite often run in low power environments (e.g. off something that uses batteries).

Embedded Systems Exam-Style Questions:

1. Define the term "embedded system".
2. List three examples of embedded systems.
3. Explain why embedded systems use solid state (Flash) storage.
4. Explain why embedded systems only need a small amount of RAM.

Storage and memory:



Fastest, smallest and most expensive
CPU registers
Cache
RAM
Hard disk
Slowest, largest and least expensive
Primary storage (volatile memory)
Secondary storage

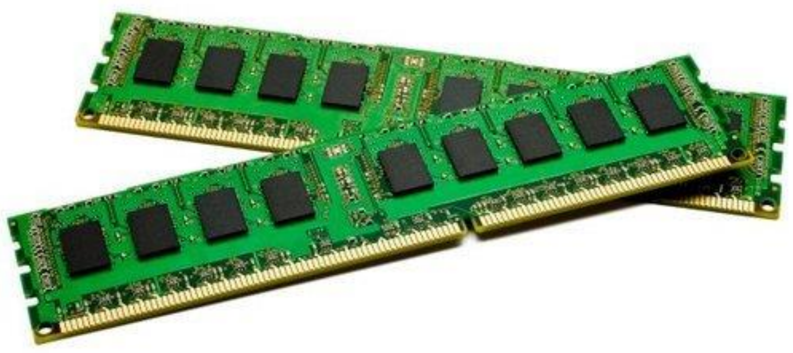
Primary Storage (memory)

Primary storage = Registers, cache and RAM.

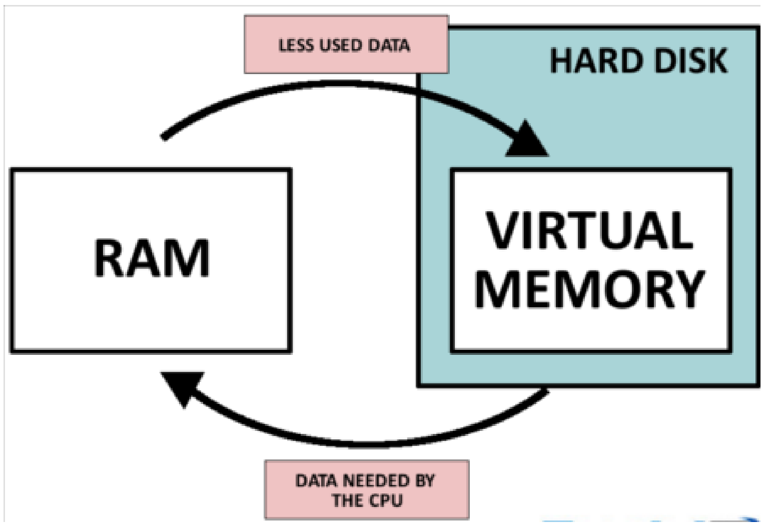
Random Access Memory (RAM) is a type of volatile primary storage that stores programs and files that are currently in use.

Volatile means that when the power is turned off, the contents are lost.

RAM is used because it is much faster for a CPU to access RAM than it is to access secondary storage such as a Hard Disk Drive (HDD).



Virtual Memory is when a portion of the Hard Disk Drive (HDD) is used as though it is RAM (hence "virtual"). When RAM fills up, in order to keep the computer running and allow the user to have more programs open, the operating system will take a snapshot of the programs running in RAM and temporarily place some of them back on the HDD. This slows the computer down a bit, but at least it keeps the programs running. When the user wants to interact the program again, it is quickly placed back into RAM in the state that it was in before.



Cache is very high speed memory next to the CPU that stores regularly used instructions. It’s almost like carrying around tools that you use all the time on a toolbelt so that they are right next to you, ready to use.

Analogy: RAM would be a bit like keeping them in a toolbox just down the hall. Secondary storage would be like keeping them in the shed at the bottom of the garden.

Read-Only Memory (ROM) is a type of storage that can only be read from and not written to. The most common use of ROM is to store the BIOS settings that boot up the computer. There is a small amount of ROM on the motherboard, when you press the “ON” button, the CPU looks in this ROM and uses it to start up the PC.

<u>Secondary Storage:</u>

Secondary storage is non-volatile, long term storage for programs, files and other data. Hard Disk Drives (HDD), Solid State Drives (SSD), optical discs (CDs/DVDs) and USB Flash Drives are all examples of secondary storage.

Secondary storage is needed to store programs and save files for later use. Obviously, RAM cannot be used to store anything that you want to use again in the future!

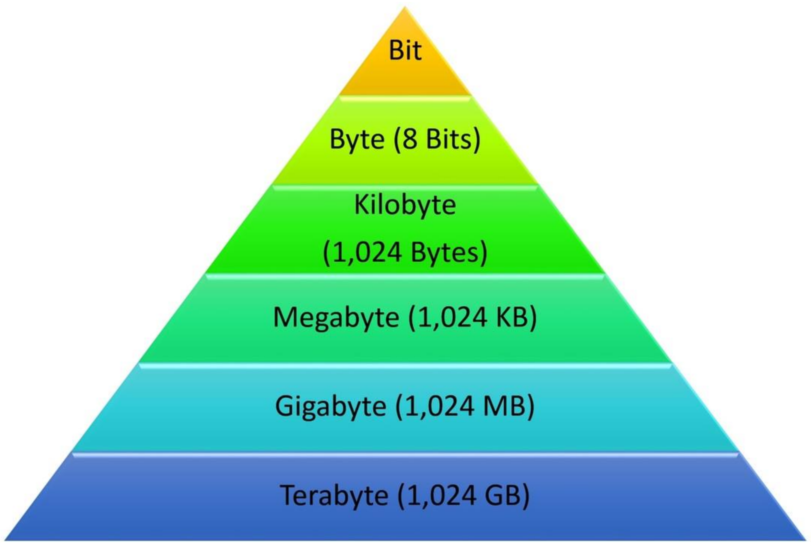
Optical storage includes CDs and DVDs.

Magnetic storage includes Hard Disk Drives. It uses a moving read-write head that passes over magnetised platters. HDDs can be a bit unreliable due to moving parts than can wear out and break.

Solid state includes Flash storage which has no moving parts and is much faster and more durable than a Hard Disk Drive.

Secondary storage is measured usually measured in GigaBytes.

- Bit = 1 or 0 (smallest unit of data).
- Nibble = 4 bits
- Byte = 8 bits
- KiloByte = 1024 Bytes
- MegaByte = 1024 x 1024 (roughly 1 million) Bytes
- GigaByte = 1024 x 1024 x 1024 (roughly 1 billion) Bytes
- TeraByte = 1024 x 1024 x 1024 x 1024 (roughly 1 trillion) Bytes



<u>**Characteristics of secondary storage**</u> that need to be considered when buying devices for certain scenarios:

- capacity (amount of data it can store)
- speed (how fast data can be read and written to it)
- portability (how easy it is to carry about)
- durability (how difficult it is to break)
- reliability (how long it tends to last without going faulty)
- cost (obvious!)

Storage and Memory Exam-Style Questions:

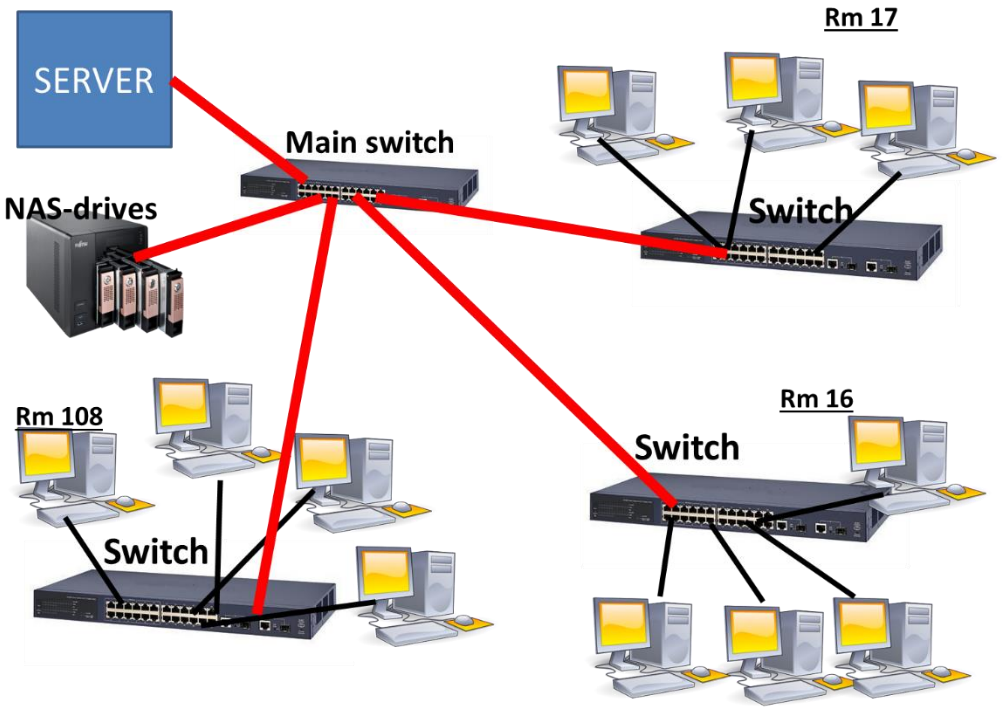
1. List two types of primary storage.
2. List two secondary storage devices.

3. Describe the function of RAM.
4. Describe the function of cache memory.
5. Explain what is meant by virtual memory.
6. State three characteristics of secondary storage.
7. Explain why HDDs are not suitable for use in a SatNav (refer to the characteristics of storage).
8. What is the smallest unit of data?
9. How many bits in a Byte?
10. How many Bytes in a KiloByte?
11. How many KiloBytes in a MegaByte?
12. Explain why optical storage is becoming a thing of the past.

Types of networks:

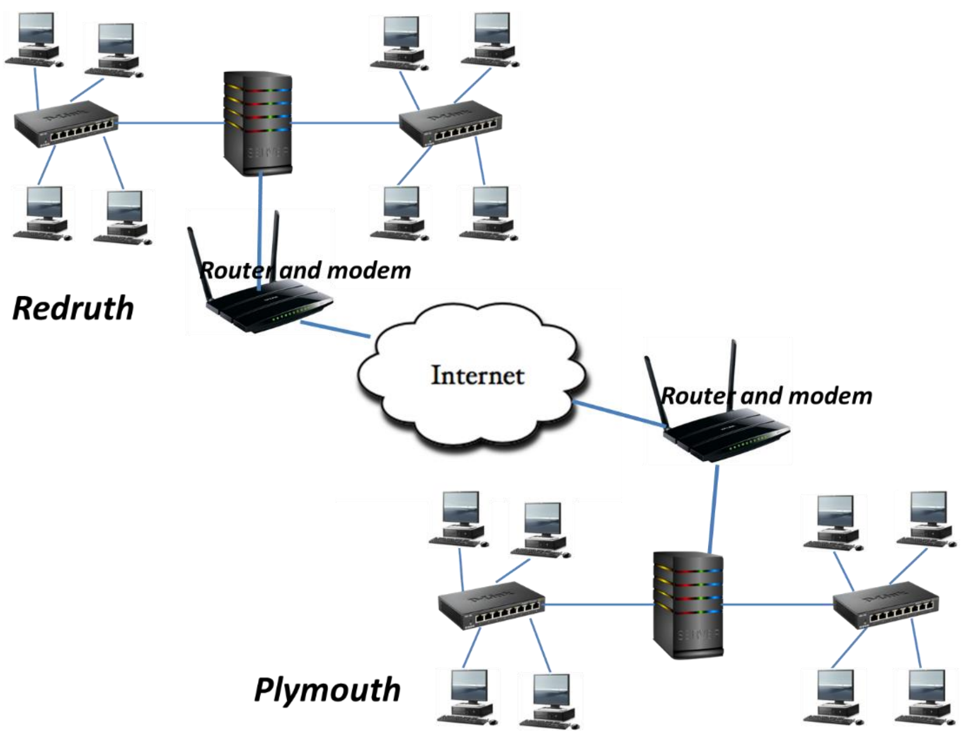
- **LAN (Local Area Network)**

A computer network over a small geographical area, such as an office or a school.



- **WAN (Wide Area Network)**

A computer network over a large geographical area. WANs are usually made up of interconnected LANs.



A Wireless Local Area Network is simply a LAN that you can connect to using Wi-Fi.

<u>Hardware for a Wireless LAN:</u>

- Wireless Access Point (WAP)
- Network Interface Cards (NIC) that support Wi-Fi – think “antenna!”
- A central switch to connect the WAP to.
- If you also want Internet connection you wil need a router and modem.
- Most wireless LANs in the home and small offices are peer-to-peer.
- If you want it to be a client-server Wireless LAN you will need a server.

But what exactly is Wi-Fi?

Wi-Fi (wireless Ethernet) uses short-range, high frequency radio signals to send data between devices on a Local Area Network.

What is a Wi-Fi channel?

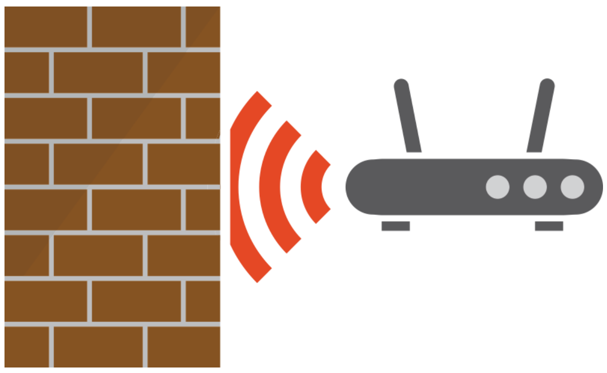
Because Wi-Fi is a type of radio communication, different channels run on different frequencies. Much like the radio or TV channels. If one channel is very busy, then this can slow a network down. Changing to a clearer channel will help to solve the problem.

What does Wi-Fi frequency mean?

Frequency is the number of radio waves per second. Each frequency relates to a certain channel.

Network performance can be affected by a range of factors, including:

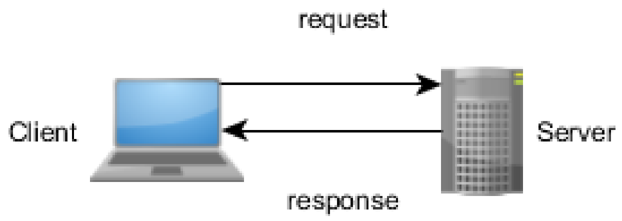
- Amount of traffic (lots of people using the network)
- Interference (from nearby electrical sources)
- Malware (e.g. worms consuming bandwidth)
- The bandwidth of the transmission media (e.g. fibre-optic has a wider bandwidth so allows more data to be transferred per second)
- Server speed (for client-server networks)
- For wireless networks, objects in the way and distance from the WAP can be a problem!



Client-server network:

- Has a powerful central server which provides resources (such as files), a login system and software updates to clients connected to the network.
- Clients request services, the server responds.
- A server usually requires skilled staff to set up and maintain. Servers are also expensive.

The school network is a client-server network. The server provides staff and students with logons and sets file permissions of what staff and student groups can access. For example the G: Drive contents can be read and edited by staff, but students can only view the contents and not make changes. The software is stored by the server and accessed from the classroom computers via the network.



Peer-to-peer network:

Computers are all at the same level and connected to each other. There is no central server. Peer-peer networks are easy to set up, but there is no way to centrally manage them. For example, if software needs to be installed or updated, a technician will have to walk around to each computer and manually install the software. This type of network might be used at home or in a small office.

Types of Network and Wi-Fi (wireless) Hardware Exam-Style Questions

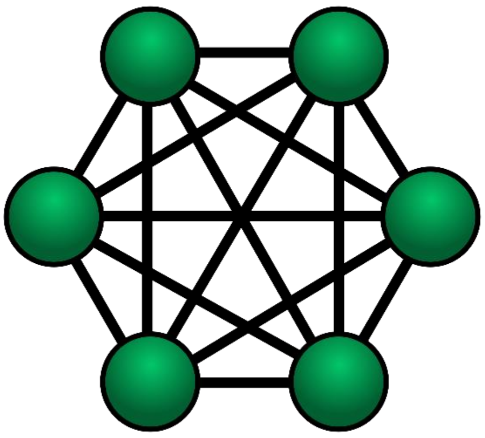
1. Identify two pieces of Wi-Fi LAN hardware.
2. Define the term "Local Area Network".
3. State what is meant by bandwidth.
4. Describe what is meant by a Wi-Fi channel?
5. Why are there different Wi-Fi channels?
6. Explain why a WAP is needed to allow Wi-Fi devices to connect to a LAN.

Network topologies:

A network topology is the physical layout of the network (when viewed from above).

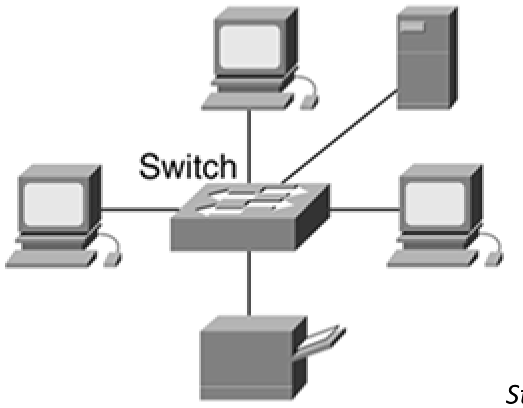
Mesh topology:

- All devices (nodes) connected to all other devices (nodes)
- Very fault-tolerant (if a cable or node fails, all the others can still communicate).
- Lots of cables
- Expensive to set up
- Difficult to add in more computers


Mesh topology

Star topology:

- Uses a central switch, with all devices (nodes) connected to the central switch.
- Easy to add in more devices (simply connect them to the switch).
- A disadvantage is that if the switch breaks the network will fail. One way to get around this is to connect all devices to another switch, so that if one fails, the other can take over.



Star topology

Network hardware:

Wireless Access Points (WAPs) – allow wireless (Wi-Fi) connections to a LAN.

Routers – allow Internet connection (forwards data packets onto the Internet) and can be used to create WANs.

Switches – connect devices over a LAN (using MAC addresses to get the data across the Local Area Network).

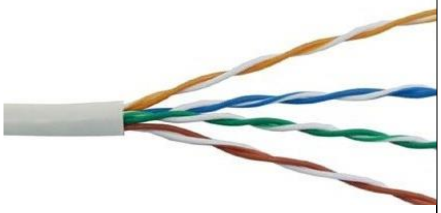
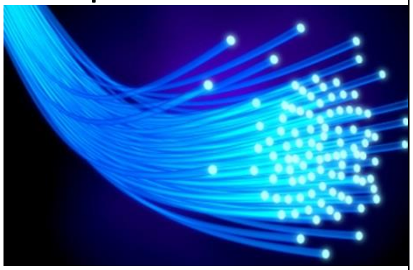
Network Interface Controller/Card (NIC) – a piece of hardware inside each computer that allows it to connect to a LAN. NICs contain the MAC address.

Transmission media – Cat-5 (Ethernet cables) and fibre-optic cables.

Cat-5 cables are copper cables that transmit data across a LAN. They are OK for short distances and low traffic. For example, from a single computer to a switch. Cat-5 cables suffer with interference and signal loss over longer distances.

Fibre-optic cables have a high bandwidth so can transfer more data per second and do not suffer with interference. Fibre-optic can be used for long distance communication and/or where there is high traffic (e.g. from a switch to the server room).

Wireless Access Point (WAP)	**Router**	**Switch**
Allows Wi-Fi connection to LAN.	Allows Internet access. Looks at IP addresses and forwards them on.	Connects devices on a LAN.

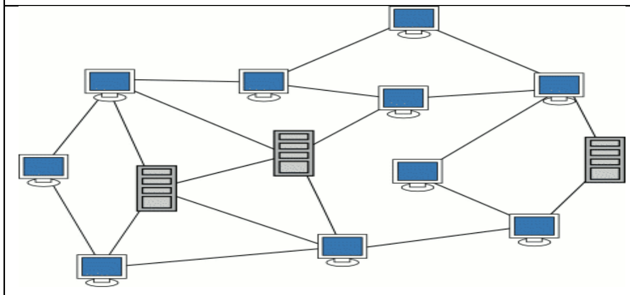
NIC Stores the MAC address and allows devices to plug into a LAN and communicate with it.	**Cat-5 Ethernet (copper) cable**  Low bandwidth cable suitable to low traffic LAN connections.	**Fibre-optic cable**  Suitable for high traffic networks and WAN connections across the long distances. High bandwidth.

Network Topologies Exam-Style Questions

1. Sketch a small client-server star topology, with four computers and a printer.
2. State two advantages of a star topology.
3. State one potential disadvantage of a star topology.
4. Explain why mesh topologies are not suitable for larger networks.
5. Describe the function of a network switch.
6. Describe the function of a router.
7. Explain the role of MAC addresses in a LAN.
8. Which type of transmission media is most suitable for high network traffic and longer distances.

<u>The Internet and the World Wide Web</u>

The Internet is a global collection of interconnected networks. The Internet is all the physical cabling and other hardware such as routers. The World Wide Web (WWW) is all the websites (software).

The Internet	The World Wide Web
 Interconnected networks spanning the globe.	Websites, social media, web browsers, etc.

<u>Domain Name System (DNS):</u>

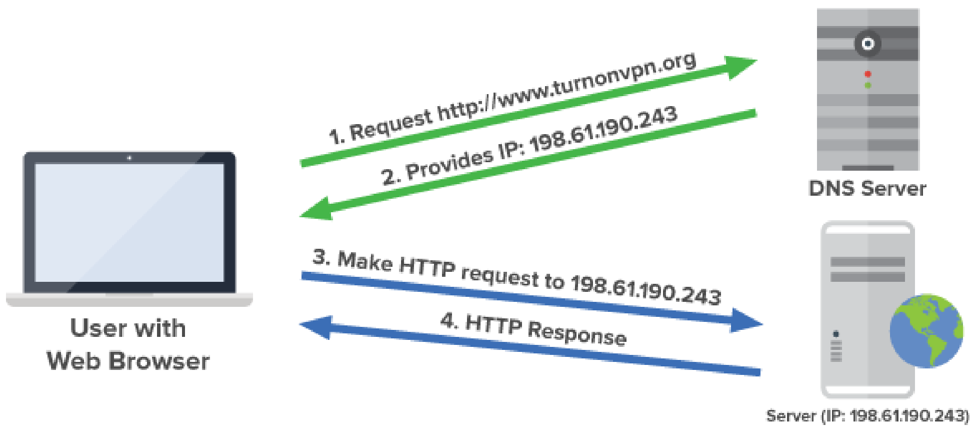
DNS converts human-readable domain names (such as google.com) into computer-readable IP addresses (such as 243.6.83.5)

Why do we need DNS?

Humans can't remember IP addresses! Computers can't read domain names! Therefore there has to be a system to convert them.

How DNS works:

1. User types in a domain name (stuff.com) or a specific URL (e.g. http://www.stuff.com/chickens) into a web browser such as Firefox.
2. DNS then looks for the IP address of stuff.com in web browser's DNS cache (a table of domain names and their matching IP addresses – a bit like a phone book).
3. If DNS doesn't find the IP address here, it goes out to the Internet Service Provider's (ISP) domain name table.
4. It DNS doesn't find the IP address here, it goes to higher level DNS server, and so on…
5. Eventually (if it hasn't found the IP address) it will ask an authoritative name server which will know the IP address.
6. Once it has found the correct IP address, DNS sends the IP address to the web browser which can then use that to locate and communicate with the website (stuff.com).
7. The web browser's own DNS cache will also be updated, so that next time it needs stuff.com it will know the IP address. Clearing your browser history usually deletes these saved IP addresses.


1. Request http://www.turnonvpn.org
2. Provides IP: 198.61.190.243
DNS Server
User with Web Browser
3. Make HTTP request to 198.61.190.243
4. HTTP Response
Server (IP: 198.61.190.243)

DNS in action – converting domain name to IP address.

The Internet and the WWW Exam-Style Questions

1. Describe the key differences between the Internet and the World Wide Web.
2. State the function of DNS.
3. Why is DNS needed?
4. List the key steps of how DNS works.

Web hosting:

Paying a company (such as GoDaddy) to store your website online. They store your HTML files on their servers and allow people to view your website using their web browsers.

Before hosting, you will also need to get your domain name registered (which also allocates an IP address to it and puts the IP address on the Doman Name System so that you website can be found!)

The cloud:

The **cloud** is simply another word for “**online**”. Cloud storage is where you store your data remotely. Usually you would pay a company to store your data.

Advantages:

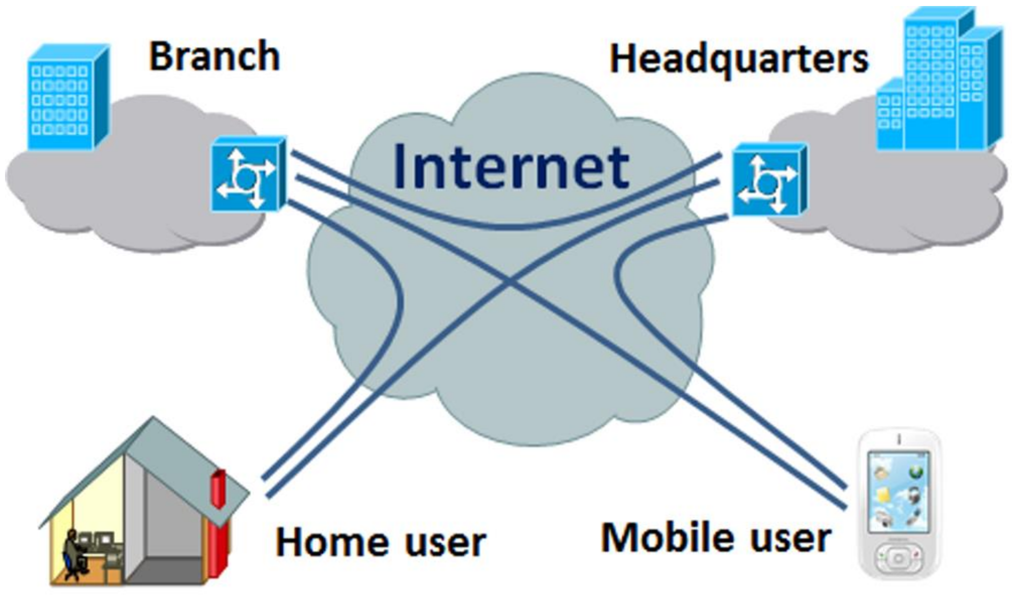
- You can view your files from anywhere in the world.
- You don’t have to buy and manage your own backup hardware.

Disadvantages:

- Can you trust the company storing it? Are there data protection laws in the country where your data is being stored?
- How vulnerable are their systems to attack?
- Have you got a backup plan if the Internet fails or they lose your data?

Cloud computing is when you use software that is being run on a remote computer. For example, a powerful server might be running the software and you will be able to view the results on your own computer. This is great if your Internet is fast and reliable!

A **Virtual Private Network (VPN)** appears to network users like they are connected to a physical LAN or WAN without needing to be anywhere near the physical part of the network. They need to login with a username and password. Once logged in, they can access files, folders and software as though they were physically in the same building as the network. Users can log in from anywhere in the world and the data is sent securely (using encryption) through the Internet. It is helpful to think of it as an “encrypted tunnel” through the Internet. Travelling business people rely heavily on VPNs.



Network protocols, layers and packet switching

Ethernet

Ethernet is a protocol and set of standards for communicating over a Local Area Network. The Ethernet protocol controls how data is sent and received across a Local Area Network. Ethernet also ensures that devices are compatible.

Since Ethernet is a Local Area Network technology, it uses MAC addresses to get data from device to device. MAC addresses are unique “physical” addresses stored on a computer’s Network Interface Card.

The Internet uses a different set of protocols to allow communication:

TCP/IP

- Transmission Control Protocol (TCP) breaks data into equal sized chunks called data packets. Each data packet is given a sequence number and a check digit.
- TCP hands the data packets to the Internet Protocol (IP).
- IP encapsulates each data packet in a wrapper with a header containing the destination IP address and the source IP address.
- IP also routes the data packets across the Internet.
- When they arrive at their destination, TCP takes over again and puts the data packets back together.
- If any are missing or corrupt, TCP will request replacements.

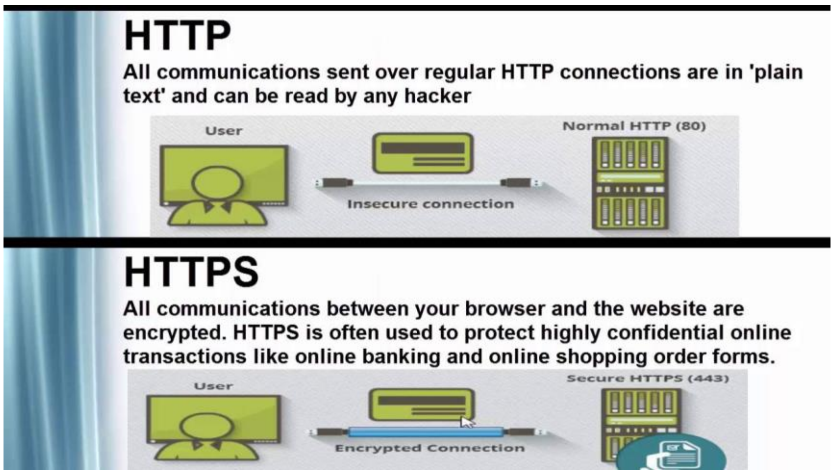
HTTP (Hyper Text Transfer Protocol)

HTTP allows web browsers to request websites from web servers (this runs on port 80). For example, you might type in HTTP://www.google.com/examples

HTTPS (Hyper Text Transfer Protocol Secure)

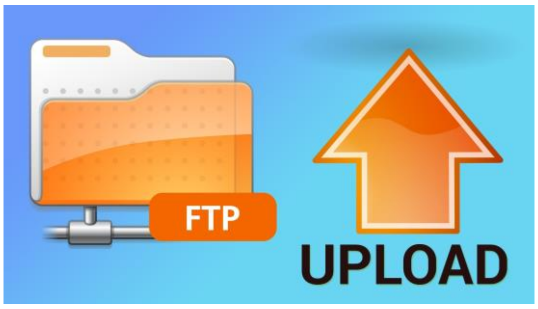
HTTPS is the secure (encrypted) version of HTTP, suitable for exchanging sensitive information, e.g. bank details.

HTTP vs. HTTPS


HTTP
All communications sent over regular HTTP connections are in 'plain text' and can be read by any hacker
User
Normal HTTP (80)
Insecure connection
HTTPS
All communications between your browser and the website are encrypted. HTTPS is often used to protect highly confidential online transactions like online banking and online shopping order forms.
User
Secure HTTPS (443)
Encrypted Connection

FTP (File Transfer Protocol)

FTP allows large files and folders to be transferred across the Internet. Website developers might use this to upload their HTML files to be hosted on a web server. Or movie fans might use it to illegally download movies!



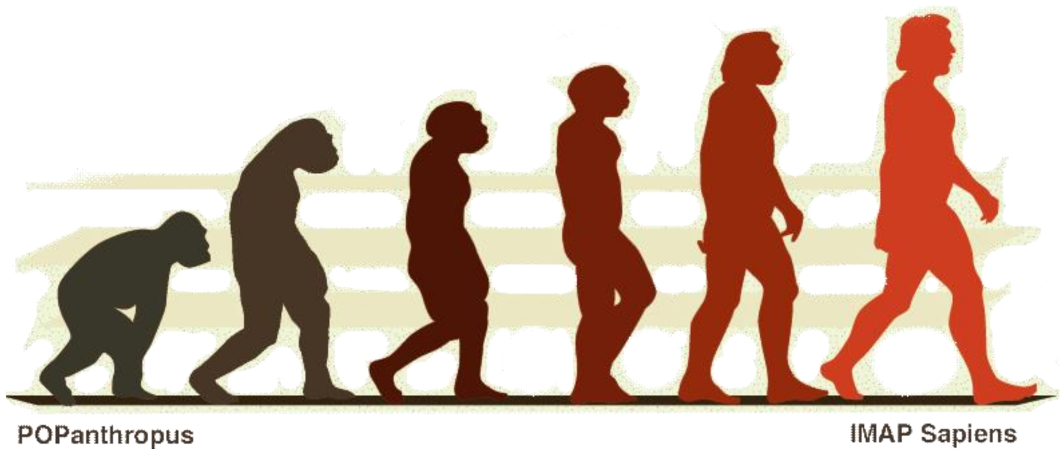
<u>The three email protocols:</u>

POP (Post Office Protocol)

Allows users to download an email ONCE ONLY. When it has been downloaded, it is removed from the email server. This is only OK for people that check their emails on one device.

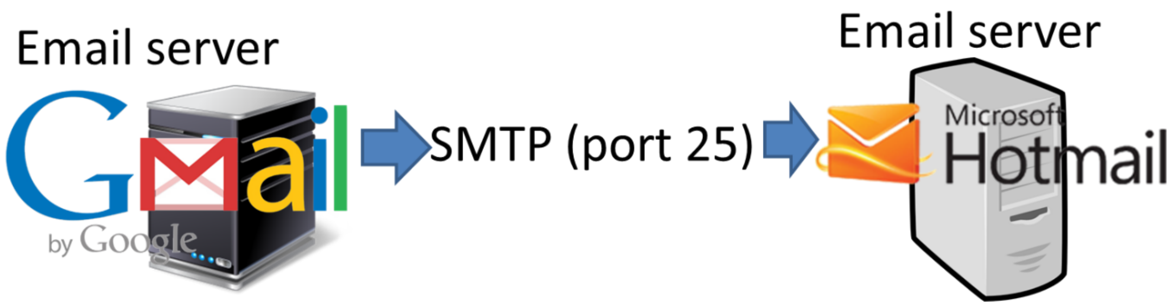
IMAP (Internet Message Access Protocol)

This is a modern email protocol that allows users to view their emails on the email server and download as many emails as they want on multiple devices.



SMTP (Simple Mail Transfer Protocol)

This is used for sending emails and for transferring emails from one email server to another. For example, it will send an email from Hotmail's email server to Gmail's email server.

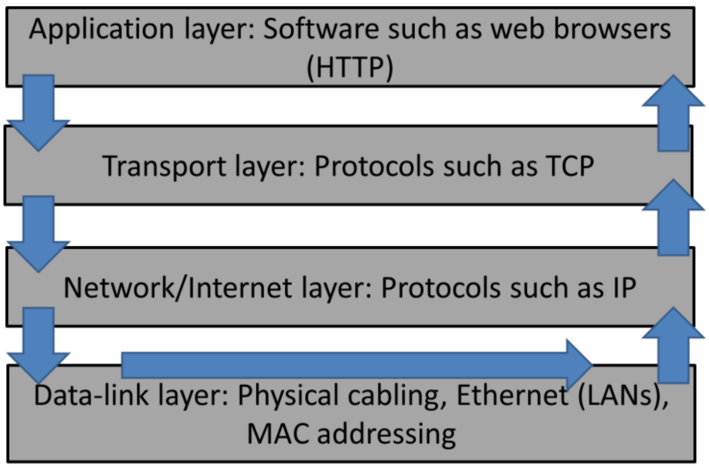


Network layers

The TCP/IP four-layer model is a something that network engineers use to understand and troubleshoot network problems.

When data (for example an image) is sent across a network, it goes down through the layers and has wrappers put around each data packet, then on the receiving side, the data packets move up through the layers and the wrappers are removed. Putting data in these "wrappers" is known as encapsulation.

TCP/IP four-layer model:


Application layer: Software such as web browsers (HTTP)
Transport layer: Protocols such as TCP
Network/Internet layer: Protocols such as IP
Data-link layer: Physical cabling, Ethernet (LANs), MAC addressing

How does the TCP/IP four-layer model work?

Basically, it's a different way of looking at TCP/IP…

The user interacts with a web browser in the Application Layer and sends an email.

The data is handed down to the Transport Layer where TCP lives.

TCP breaks the email data down into data packets, labels them with sequence numbers and check digits and hands them down to the Network/Internet Layer where IP lives.

IP encapsulates them with in a wrapper with the IP address on each packet.

IP hands down to the Data-link layer which gets them across the physical network.

The data packets are handed back up the layers, until eventually they are displayed to the user in the Application Layer.

Packet switching

Packet switching is also a different way of looking at TCP/IP. All packet switching means is that data is sent as individual chunks (data packets) that are routed through a network. Each data packet might take a different path depending on the amount of network traffic. Routers can detect busy routes and send data packets around different paths to get them to their destination.

The main benefit of packet switching is that all devices can SHARE the transmission media, there are no dedicated connections between machines.

A very rough analogy is that of car vs train travel. If a group of people wanted to travel to London, they could each get in their own car and decide on which route they wanted to travel. They would not need to take the same route. But they would still get there and meet up again. This is similar to the idea of packet switching. If they took the train, they would only be able to take a single route along a dedicated line (the railway!).

Data packets

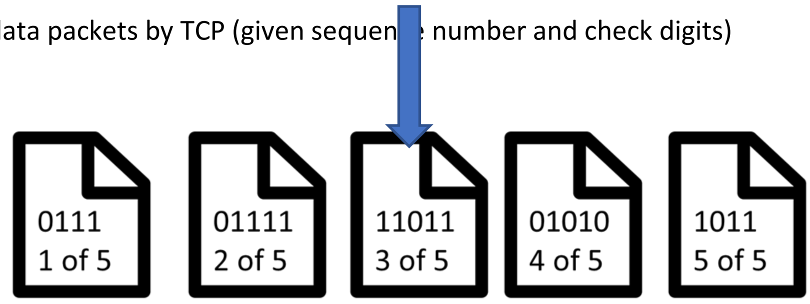
Data packet structure

Source address	Destination address	Sequence number	Data	Check digit
254.65.2.18	192.45.17.2	5 of 1056	10101010 10111010 00101011	49495501

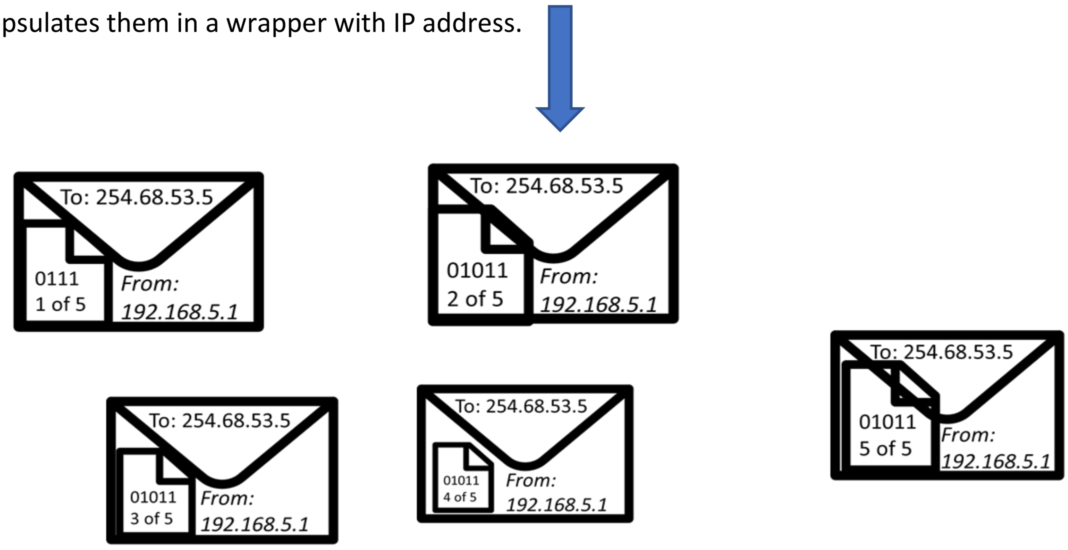
TCP/IP basic idea

Someone want so to email a frog picture:

Broken down into data packets by TCP (given sequence number and check digits)



IP encapsulates them in a wrapper with IP address.



IP delivers them, then TCP reassembles original image and requests any missing/corrupt packets.

Network Protocols Exam-Style Questions

1. Define the term “protocol”.
2. List the three email protocols.
3. What does FTP stand for?
4. What is FTP used for?
5. Explain why IMAP is an improvement over POP.
6. What is SMTP used for?
7. Which protocol is used for secure transmission of data to a web server? (For example, accessing a banking website).
8. Briefly explain how TCP/IP is used to send an image across the Internet.
9. List the four layers of the TCP/IP Four Layer Model.
10. Sketch and label a data packet.

System security

Threats posed to networks:

There are a number of threats that can cause harm to a network or that can be used to steal data. The main threats that you need to know about are explained in this section.

Virus:

A malicious piece of software that infects other files. It can only replicate if the file is opened and run, e.g. if it is inside a .exe file and the user runs it or if it is a macro in an Excel file and the user opens the Excel file. A virus will replicate and infect other susceptible files, but requires human computer users to spread them from computer to computer. For example, accidentally sharing an infected file with someone via USB stick.

It is called a virus after the biological viruses that spread by taking over human cells. Likewise, a computer virus needs to infect other files to spread and survive.

(Search for "The Concept Virus" to read about a real virus).

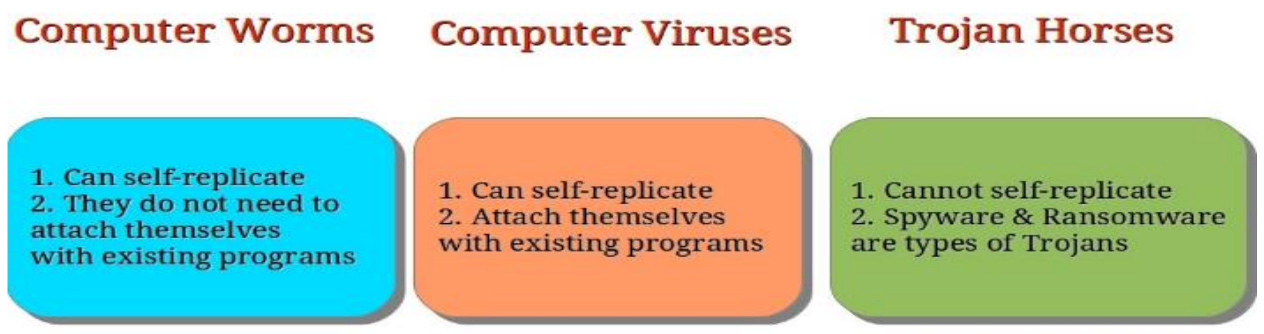
Worm:

A complete standalone piece of malicious software that can spread by itself without any human interaction. It can replicate and often the aim of a worm is to replicate itself thousands of times and send itself between computers and networks to consume all of the bandwidth and grind the network to a halt. A worm does not need to infect other files, it is a complete program and can be very sophisticated.

(Search "Stuxnet Worm" to read about a real worm).

Trojan:

A piece of malicious software that tricks users into downloading by making it look like it is something useful. For example, you might download a program that claims to clean your system up and improve the performance of your PC. You download it in good faith, but what you've actually downloaded is a Trojan! A Trojan could create a backdoor in your system, allowing hackers to access your PC remotely. It could be a keylogger which will monitor your keystrokes and upload them to a hacker (including you typing in your usernames, passwords and bank details!). It could be used to turn your camera into a spying device, etc. Ransomware is another potential threat from Trojans!



Computer Worms	Computer Viruses	Trojan Horses
1. Can self-replicate 2. They do not need to attach themselves with existing programs	1. Can self-replicate 2. Attach themselves with existing programs	1. Cannot self-replicate 2. Spyware & Ransomware are types of Trojans

Denial of Service (Dos) attack:

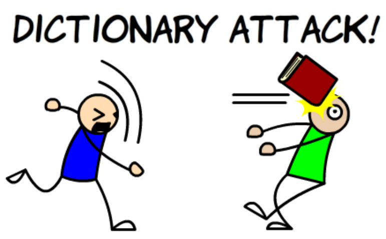
Any attack on a system that denies you (and others) of a service. For example, hacking and crashing an email server so that you can't access your emails. Bombarding a website with IP data packets until it crashes or so that the bandwidth is consumed and no-one else can access it.

Distributed Denial of Service (DDoS) attack:

Infecting hundreds or thousands of computers with Trojans that can all be triggered at once to send a flood of IP data packets to a server and overwhelm it and crash it.

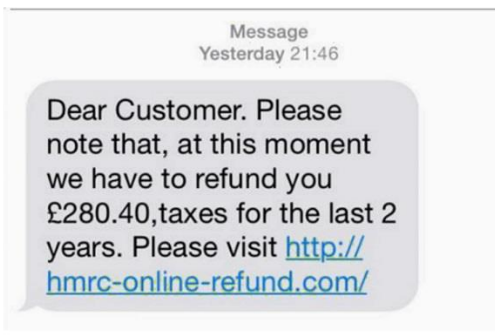
Brute Force attack:

Guessing thousands (or millions or even billions!) of password combinations until they crack the password. This is sometimes called a "dictionary attack" because it goes through all the words of the dictionary first. Obvious defence against this is to choose long passwords with a mixture of characters, symbol, numbers, uppercase and lowercase letters. Another defence is to time-out after three incorrect login attempts. An even better defence would be to use 2-factor authentication, e.g. sending a text message to your phone with a 6-digit PIN to enter along with your password.



Phishing:

Phishing scams are emails that are sent to numerous email addresses in the hope that some of the recipients will fool for the scam. It could be trying to trick them into downloading a Trojan, or tricking them into giving away usernames, passwords, or even bank details! Obvious defence is to NEVER give out any personal information and report suspicious activity.



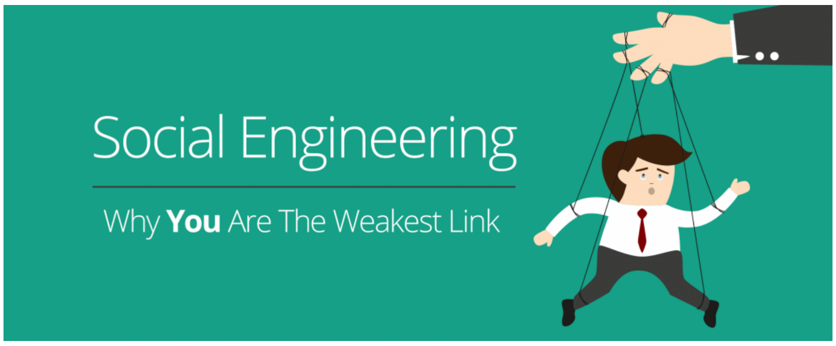
Message
Yesterday 21:46

Dear Customer. Please note that, at this moment we have to refund you £280.40,taxes for the last 2 years. Please visit http://hmrc-online-refund.com/

Social Engineering:

Any method that hackers use to trick PEOPLE into either downloading malicious software or giving out personal information. Phishing is a form of social engineering. Kevin Mitnick is a famous hacker that used to either telephone or actually walk into companies claiming to be an IT technician and fool people into giving him their usernames and passwords! This technique works far more than you might think!

People are a weak-point in even the most secure computer systems!



SQL Injection:

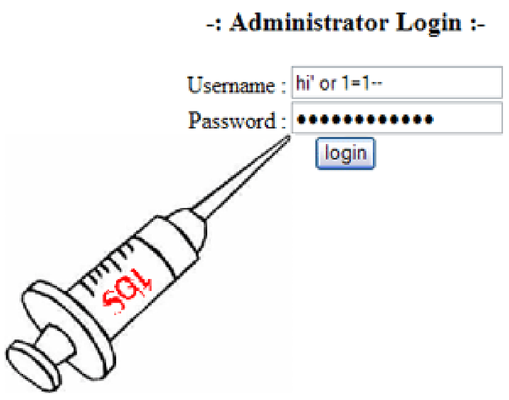
SQL injection is when a hacker types SQL code into the input box of a form and submits it. The SQL code could allow a hacker to view usernames and passwords stored in a database.

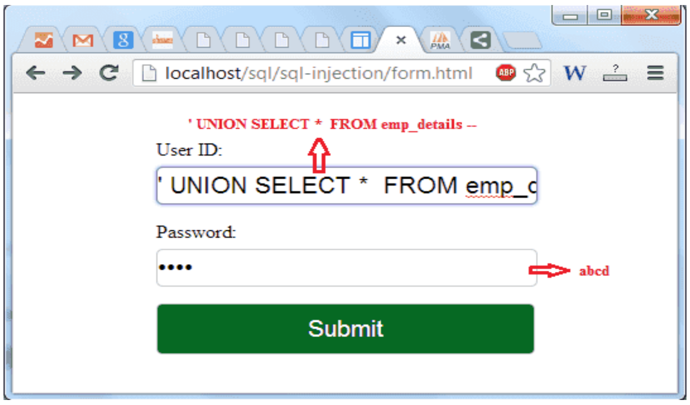
For example: If someone entered this into a username field:

```
SELECT * FROM users WHERE 1 = 1;
```

This might "fool" the database into returning all data from the "users" database table.

A way to defend against this would be to VALIDATE the data. E.g. length check, or check for special characters e.g. ";" that goes at the end of SQL statements.

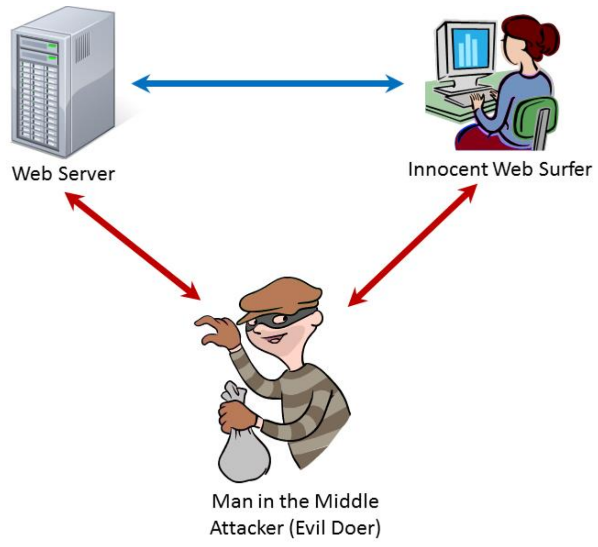



localhost/sql/sql-injection/form.html
' UNION SELECT * FROM emp_details --
User ID:
' UNION SELECT * FROM emp_d
Password:
••••
abcd
Submit

Data interception and theft:

Cyber criminals (hackers) can intercept data as it is travelling across the Internet. For example; as a user logs in to their online bank account their username and password is sent across the Internet. A hacker could intercept this and try to get their details. One way to prevent this is to use encryption which scrambles the data and makes it unreadable to anyone that intercepts it. The data becomes readable again when it reaches the bank's computer. HTTPS is the protocol that allows secure (encrypted) website access.

Cyber criminals might also try and hack into databases and steal sensitive data about customers. They might then sell this data on to make money. Encrypted the database is one way to prevent hackers from seeing sensitive information.



Poor network policies:

Network policies are rules that users must stick to when using a network. It also outlines procedures such as how often to back up data and how complicated passwords should be. Poor network policies make a network more vulnerable to attack.

Identifying and preventing vulnerabilities:

Potential threats to a network must be discovered and prevented BEFORE a cyber-criminal finds them. Testing a network and installing security measures are vital. There are a range of techniques for testing a network and preventing cyber-attack.

Penetration testing:

Penetration testing is when companies pay professional hackers to hack into their systems and find vulnerabilities. Penetration testers (professional, ethical hackers) must only penetration test what the company has agreed with them. For example, a company might want to test how secure their Wi-Fi is, but not want to test how secure their database system is. Penetration testers are only legally allowed to hack into a company with the manager’s permission and a contract drawn up and agreed between the hacker and the company.

Any vulnerabilities (security weaknesses/holes) must be reported by the hacker.

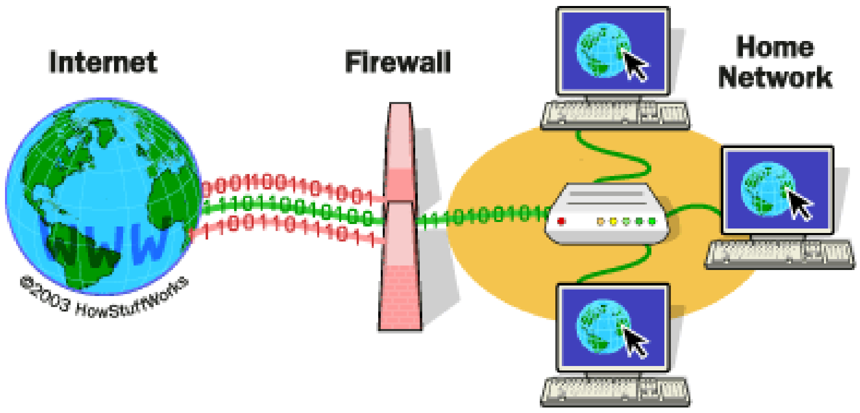
Network forensics:

Network forensics involves monitoring a network and capturing network traffic (data packets) and alerting the administrator to any issues. Network forensics really comes into play when there has been a cyber-crime. For example; if the school network was hacked into and student data stolen, a network forensics team would look at the network logs to see what happened and to try and find the IP address of the computer that the cyber-criminal used. Network forensics can help to see what happened and who did it. This can then be used in court to try and prosecute the criminal and to learn where any holes (vulnerabilities) in the network are.

E SCENE - DO NOT ENTER - CRIME
CRIME SCENE - DO NOT ENTER
ENTER - CRIME SCENE - DO NOT E
5

Firewall:

A firewall blocks unwanted access to the network from the Internet. A firewall monitors incoming data packets and checks for malicious activity. A firewall can block certain ports. Blocking off ports will prevent access to particular programs and protocols from the Internet.



Network Policies:

Network policies are rules and guidelines on what users can and can’t do as well as the software to implement these rules.

- User Access Policy – Making sure all users have accounts with usernames and passwords.
- Acceptable Use Policy – Making sure users use the network appropriately and for example don’t access personal email accounts that might result in viruses being downloaded.
- Backup and Recovery Policy – What happens in the event of a major event happening to the network. E.g. fire, flood, hack, major hardware failure.

Anti-Malware software

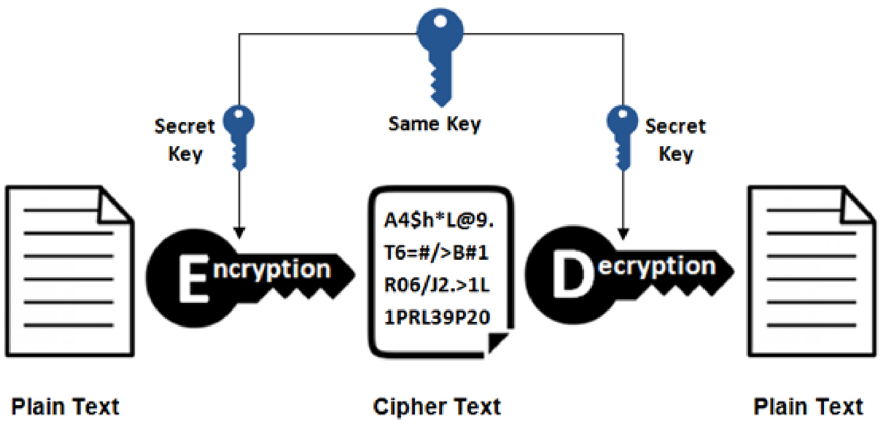
Anti-malware software scans a computer system checking for any malware (including viruses, worms and Trojans). It will notify the user if any are found so that they can be removed.

Encryption

Encryption means scrambling data so that only the intended recipient can understand it. Anyone that tries to steal the data will not be able to read it.

Data is encrypted with a secret key that only the sending and receiving computers have access to. No-one else can get access to the secret key.

An analogy would be that you write a letter on paper and put it in a box. Then you padlock the box and send it to a friend. Both you and your friend have the key to open the padlock. No-one else will be able to unlock and view the letter.



Passwords

Passwords are another method of preventing unauthorised access to a network. Strong passwords that mix together uppercase and lowercase letters, numbers and symbols are most effective. They also need to be long enough so that they cannot be guessed. Brute force passwords attacks can check millions of passwords per second. Having a system that times out after 3 incorrect guesses, or using a text-message PIN code system can help avoid this.

User access levels:

A server manages user groups and permissions. Different user groups have different access levels. For example in a school network the network administrator can get access to all areas of a network and create, read, modify and delete files. A student can only create files in their own user area. A teacher can create files in the G: Drive, but students can only read them. If a hacker got into a student account, they would not be able to inflict much damage; in fact, they would only be able to steal or delete that one student's files.

System Security Exam-Style Questions

1. List three types of malware.
2. Describe the key difference between a virus and a worm.
3. Describe what is meant by phishing (a type of social engineering).
4. Describe how a brute force attack might be used to crack passwords.
5. Describe how network forensics can be used to improve security.

6. Sketch a diagram to show how encryption works.
7. Identify three network policies.
8. Describe one of your chosen network policies in detail.
9. State what is meant by SQL injection.
10. Describe two validation techniques that can help prevent SQL injection.

Systems Software (OS and utilities)

Systems software is software that controls and manages the hardware and performs useful tasks to keep the computer running. System software includes the operating system and utilities (tools).

The operating system is a large piece of software that controls the operation of the computer and allows the user to interact with the computer.

Utilities (tools) are small pieces of software that perform a particular task.

Operating system functions:

- Multi-tasking – allows the CPU to run many programs at the same time (it actually does this by rapidly switching between programs to create the illusion that they are running at once).

- User interface – presents a Graphical User Interface (GUI) to the user so that they can easily interact with the computer.

- Memory management – loads programs into (and out of) RAM and keeps track of where the data is while the program is running. It also controls virtual memory.

- Peripheral management and drivers – allows devices to connect to the computer and allows the CPU to understand the external devices. For example, a mouse sends in a sequence of 0s and 1s, the computer makes sense of this by using a device driver that “understands” what these mean and convert that into an instruction that the computer can understand, e.g. “move cursor up the screen”.

- Security – provides a user logon system and file access permissions (e.g. one user might be able to only access certain files).

Think MUMPS (Multi-tasking, User-interface, Memory management, Peripheral management, Security).

The OS also manages user accounts and file storage.

Utility system software:

Encryption software utility

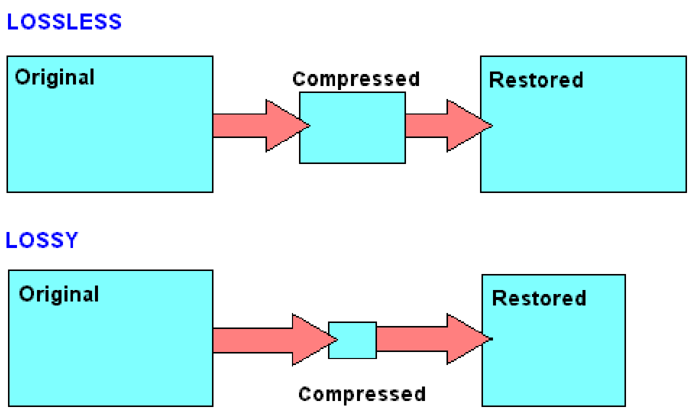
The encryption tool encrypts (scrambles) data using a key that is known only to the sender and receiver. It prevents anyone else from reading the data.

Defragmentation utility

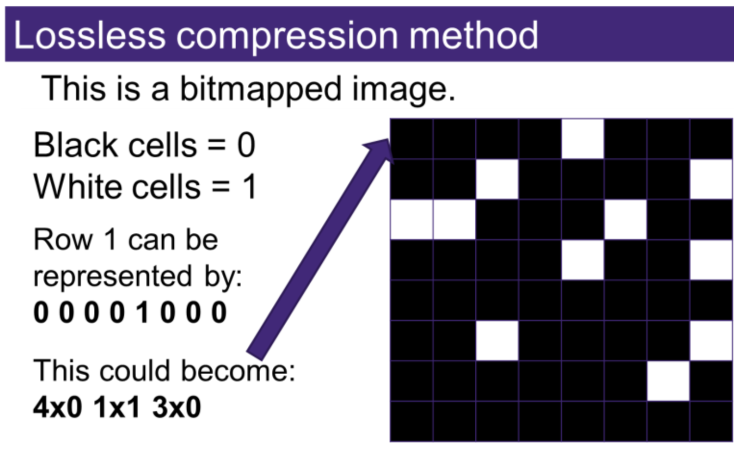
The disk defragmentation tool puts files back into sequential order on the hard disk drive. Over time, files can become fragmented (broken up) on the HDD, which slows the HDD down because it has to locate each piece of the file (the read/write head and platters take time to move into position). Defragmenting means putting the fragments back together to speed up the HDD.

Data compression utility

The data compression tool allows users to shrink file sizes so that files and programs take up less storage space. Lossy compression shrinks the file down, but also loses some of the quality. Lossless compression shrinks the file down, but restores it to its original size and quality after.



Lossless compression uses clever algorithms to shrink and restore files:



Lossless compression method

This is a bitmapped image.

Black cells = 0
White cells = 1

Row 1 can be represented by:
0 0 0 0 1 0 0 0

This could become:
4x0 1x1 3x0

System Software Exam-Style Questions

1. Describe what is meant by “system software”.
2. List two functions of an operating system.
3. Explain the function of memory management.
4. Why are device drivers required to allow peripherals to connect and communicate?
5. List three system utilities.
6. Describe the key difference between lossy and lossless compression.
7. Which type of compression is most suitable for streaming YouTube videos? Why?
8. Explain how the encryption utility can help to keep data secure.
9. Explain how defragmentation can improve a computer’s performance.

Data backup:

Backup (tertiary storage) is required to make sure that if data is lost, then it can be replaced.

Cloud storage, external HDDs and NAS-drives are all types of tertiary storage (backup).

Backup/tertiary storage concepts:

Data redundancy: Storing copies of data on several drives in different locations. If one drive fails, the others can take over.

Full backup: Backing up all of the data. Full backups take a long time to complete and are not done very often.

Incremental backup: Only backing up the changes since the last full backup. These are quick to complete.

In the networks backup policy, they will often state how many times a full back up should be performed and how often an incremental backup should be performed. It will also outline what would happen in the event of a drive failure.

Ethical, legal, cultural and environmental concerns

- Ethical issues (is something morally right or wrong?)
- Legal issues (laws!)
- Cultural issues (what impact is a technology having on cultures? E.g. is the Internet changing the way Amazonian tribes live their lives?)
- Environmental issues (e.g. habitat destruction by mining materials for SmartPhones and what to do with waste electronics goods).
- Privacy issues (surveillance, CCTV, social media use, etc.)

Legislation (laws) relevant to Computer Science:

- **The Data Protection Act 1998**

  Makes it illegal for companies to share or sell your data with permission from you.

Makes it illegal for companies to have poor computer security.

Forces companies to only store relevant information about you. For example, they do not need to know your medical history if all you want is a bank account!

- **Computer Misuse Act 1990**

  Makes hacking and data theft illegal.

- **Copyright Designs and Patents Act 1988**

  Prevents people and organisations from copying each other’s work without permission.

- **Creative Commons Licensing**

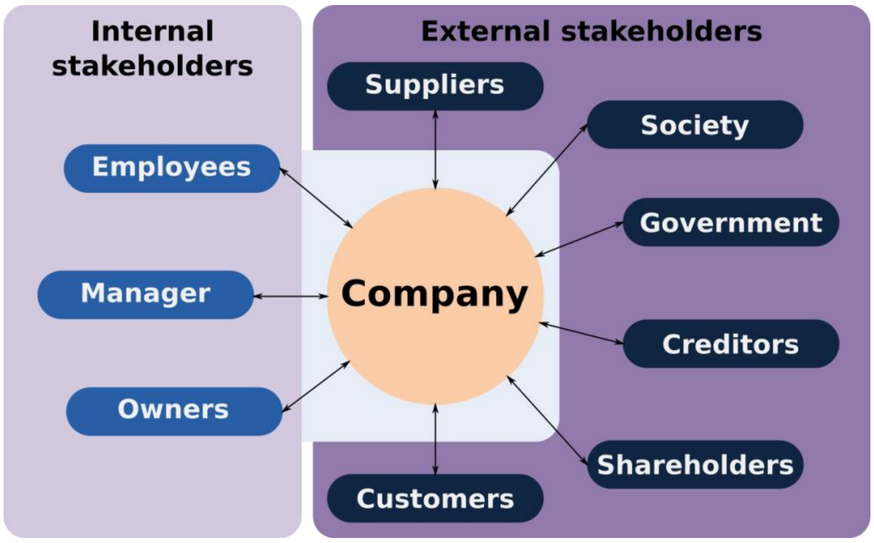
Allows people are organisations to allow others to use their work for non-profit purposes. For example, somebody might allow their art work to be used for educational purposes only.

- **Freedom of Information Act 2000**

  Allows members of the public to request information from large organisations such as the Government. For example, you could request to see how much a local MP gets paid, or where the council has spent the money.

Stakeholders

A person with an interest (usually financial) in a company. For example, the stakeholders for a school are the students, parents, teachers, governors and local businesses.



<u>Open source vs proprietary software</u>

Open source software allows other people to access the original source code. People can view, edit and pass on the source code. Open source software means that a large community of programmers can make improvements to the code. There is some very good open source software, including the Linux operating system (an alternative to Microsoft Windows). Open source software is usually free and a creative commons license that allows users to use it for non-profit is often attached with it.

Proprietary software is owned by a company with the view to making money. Proprietary software is closed source because the company does not want anyone to be able to view the source code. Proprietary software is usually copyright protected.

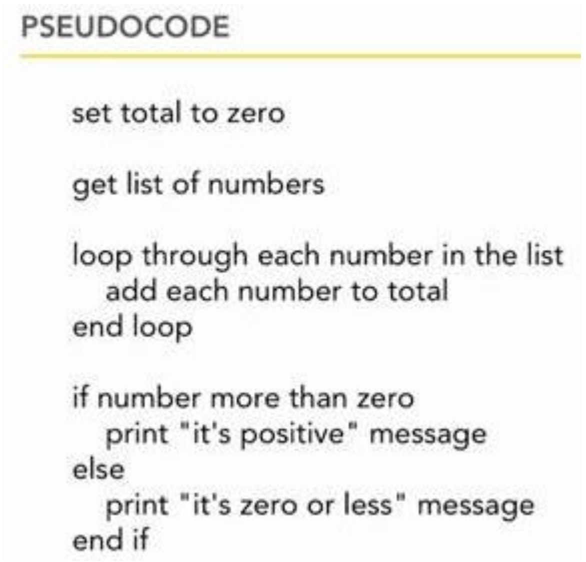
Ethical, Legal, Cultural and Social Issues Exam-Style Questions

1. List three laws (pieces of legislation) relating to computing.
2. Which law is used to ensure that companies' keep your data secure?
3. Which law can be used to prosecute hackers?
4. Which law could be used to view how much the Council spends on company cars for staff?
5. Define "stakeholder".
6. Give two examples of stakeholders in Facebook.
7. Discuss the advantages and disadvantages of creating an open source computer game.
8. Explain why most companies want to create proprietary (closed source) software.

Algorithms

Algorithms are a step-by-step set of instructions to solve a problem. Algorithms can be written as **pseudocode or flowcharts**.

Pseudocode is code written in English. The idea is that programmers can plan an algorithm before they actually start programming it. A group of programmers might know different programming languages, but they will all be able to understand pseudocode.



PSEUDOCODE

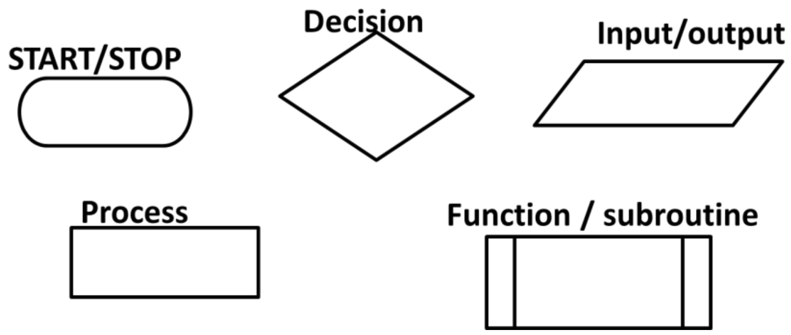
```
set total to zero

get list of numbers

loop through each number in the list
   add each number to total
end loop

if number more than zero
   print "it's positive" message
else
   print "it's zero or less" message
end if
```

Flowcharts are another way to design algorithms that all programmers will be able to understand and implement in their own programming language.



Computational thinking

Being able to think how a computer could be used to solve a problem.

Abstraction – removing unnecessary details. Focusing only on the details you want.

Decomposition – breaking a problem down into smaller solvable sub-problems.

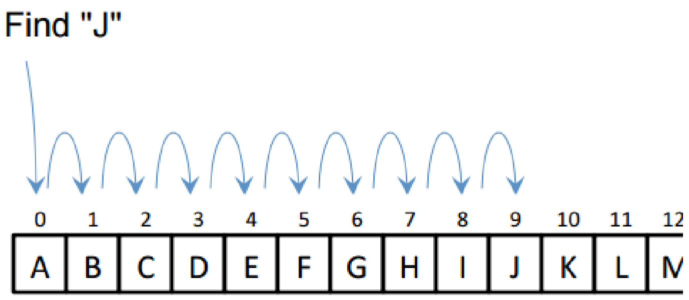
Algorithmic thinking – creating step-by-step instructions to solve a problem using computers.

The London Tube map is a good example of abstraction. The locations are not geographically accurate, because they do not need to be. All it needs to show is how to get from station to station. Details such as distance, buildings/landmarks are not included because they are not needed. The focus is on how to get about on the underground.

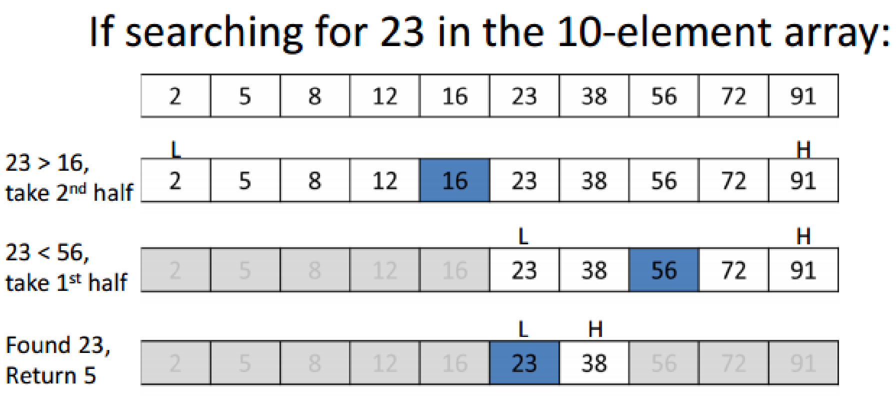
Kensal Rise
Brondesbury
Finchley Road
Kensal Green
Swiss Cottage
Queen's Park
Kilburn High Road
South Hampstead
St. John's
Park
ida Vale
ick Avenue
Royal Oak
rne Park
Paddington
Edgware Road
Marylebone
Baker Street
Edgware Road
Ladbroke Grove

Standard searching algorithms:

Linear search – just looks through each item in the list until it finds it.

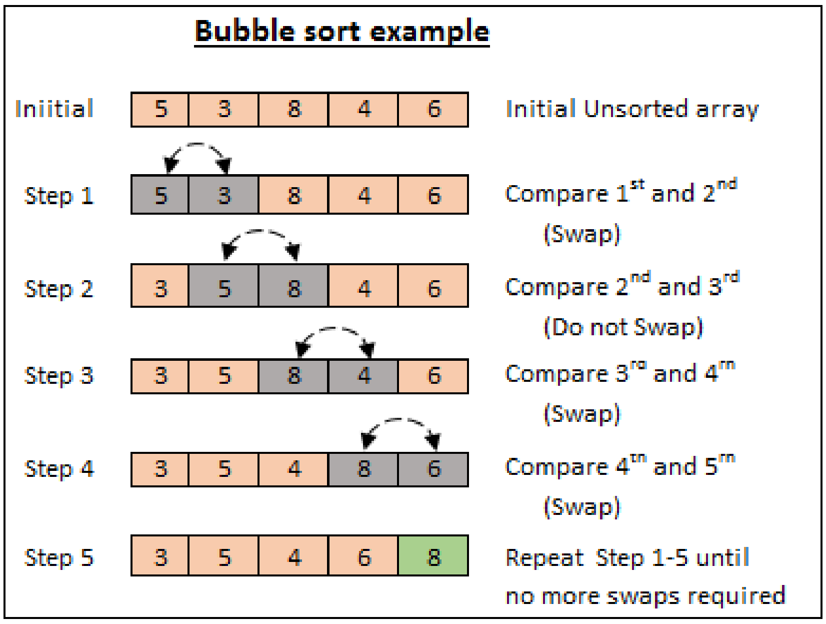


Binary search – finds the middle value of the list. Breaks the list in two, sees if it is above or below the middle. If it is above, it will then discard the bottom half of the list. It will then break the top half in two. Again, it checks if it is above or below, if it is below, it will discard the top half and so on until it finds the item. This can be a much faster way to home in on an item.

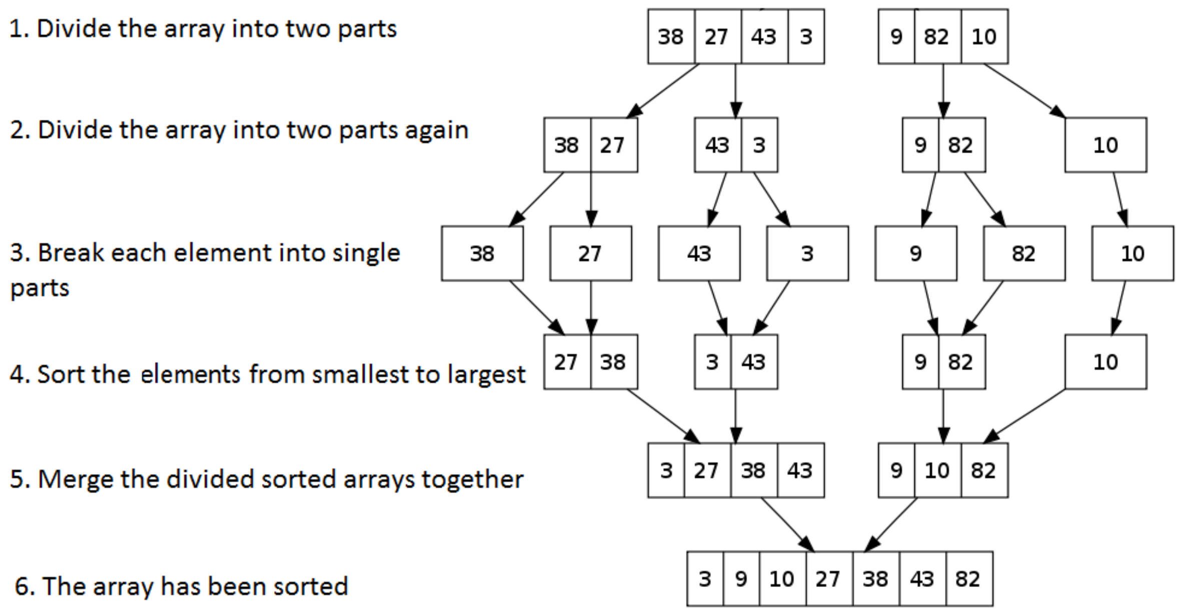


Standard sorting algorithms:

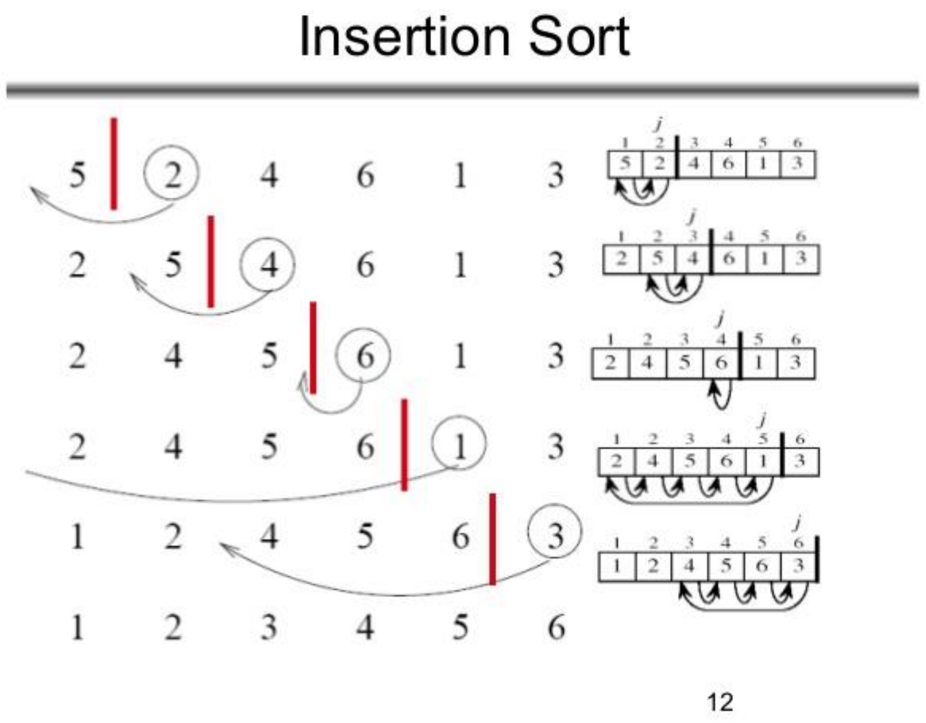
Bubble sort – swaps values in to order with each iteration until they are in order.


Bubble sort example
Iniitial 5 3 8 4 6 Initial Unsorted array
Step 1 5 3 8 4 6 Compare 1st and 2nd (Swap)
Step 2 3 5 8 4 6 Compare 2nd and 3rd (Do not Swap)
Step 3 3 5 8 4 6 Compare 3rd and 4th (Swap)
Step 4 3 5 4 8 6 Compare 4th and 5th (Swap)
Step 5 3 5 4 6 8 Repeat Step 1-5 until no more swaps required

Merge sort – breaks the lists into smaller lists and rearranges and merges them into the correct order.


1. Divide the array into two parts
2. Divide the array into two parts again
3. Break each element into single parts
4. Sort the elements from smallest to largest
5. Merge the divided sorted arrays together
6. The array has been sorted
38 27 43 3 | 9 82 10
38 27 | 43 3 | 9 82 | 10
38 | 27 | 43 | 3 | 9 | 82 | 10
27 38 | 3 43 | 9 82 | 10
3 27 38 43 | 9 10 82
3 9 10 27 38 43 82

Insertion sort – takes the list values out and inserts them into the correct order.



Algorithms Exam-Style Questions

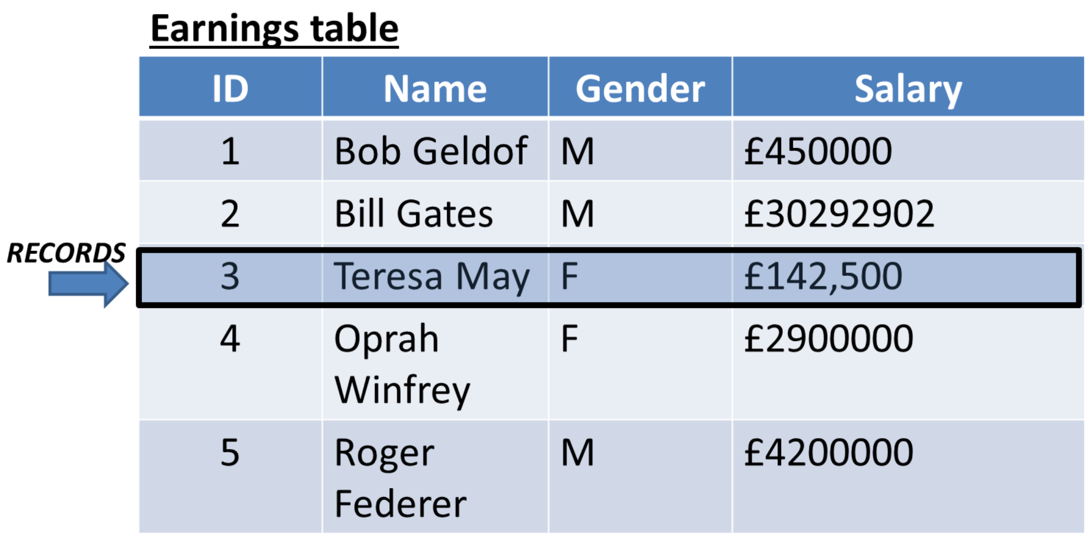
1. State what is meant by “abstraction”.
2. State what is meant by “decomposition”.
3. Describe one example of abstraction.
4. Explain why a software company might want to design an algorithm using pseudocode and/or flowcharts before writing the actual code.
5. State name of the algorithm that breaks a list into two portions and sees whether the value is above or below, then removes the portion where the value does not exist, then repeats this process until it has found the value.
6. State the name of the algorithm that iterates over a list and swaps the order of each pair until it has sorted them.
7. Briefly explain how a merge sort might sort this list: [“Banana”,”Apple”,”Orange”,”Kiwi”,”Satsuma”]
8. Explain why a binary search algorithm is almost always faster than a linear search.
9. Write a binary search algorithm – just attempt it, even if it isn’t fully complete you can still pick up some marks.

Databases and SQL

Databases are large collections of organised data. Facebook uses databases to store all of the user's details and uploaded content. Databases consists of tables where data is stored in records (rows).

Records contain a single row of data about a person or object.

Database table example:



Earnings table

ID	Name	Gender	Salary
1	Bob Geldof	M	£450000
2	Bill Gates	M	£30292902
3	Teresa May	F	£142,500
4	Oprah Winfrey	F	£2900000
5	Roger Federer	M	£4200000

Structured Query Language (SQL) is used to create, update, find and output data from databases.

The basic SQL syntax for searching for and selection data is:

```
SELECT columns FROM tablename;
```

SQL example 1:

```
SELECT * FROM earnings WHERE Gender = "M";
```

This would select all of the records from the Earnings table where the Gender is male.

SQL example 2:

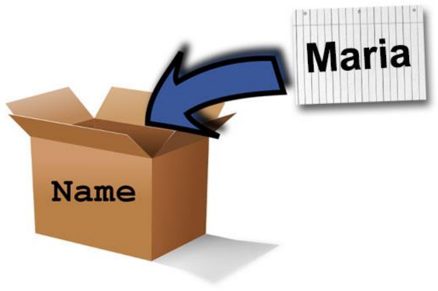
```
SELECT Name, Salary FROM earnings WHERE Salary > "£450000";
```

This would select the Name and Salary of the people that earn greater than £450,000.

Note that SQL injection is when hackers use SQL to try and trick a database into either giving away sensitive information, or allowing them to log into a system illegally.

Programming

Variable: A named container that stores a value in RAM that CAN change.



Constant: A value that CANNOT change.

Data-types: The type of data that is stored in a variable.

String: Alphanumeric characters (e.g. $myPassword123)

Integer: Whole numbers (e.g. 3)

Real/float: Decimal numbers (e.g. 3.14159)

Boolean: TRUE or FALSE

In programming string type-casting is when you change the data-type of a variable.

Binary shifts:

Binary shifts or "bit-shifts" moves binary numbers to the left or right.

A shift of 1 space left will double the number. 1 shift right will halve it.

For example:

If we start with the number 12 in binary:

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

Bit shift left the number 12 by 1 place (<< 1)

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

This gives 24

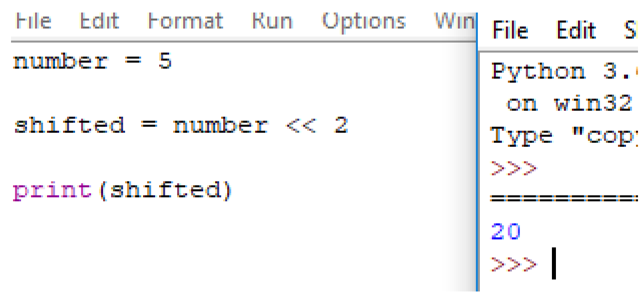
Bit shift left the number 24 by 1 place (<< 1)

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

This gives 48

Each left binary-shift doubles the number.

In code below shows how to bit-shift in Python:



```
number = 5

shifted = number << 2

print(shifted)
```

```
Python 3.
 on win32
Type "cop
>>>
=========
20
>>> 
```

The number 5 is shifted left (<<) by 2, so it doubles and doubles again, giving 20.

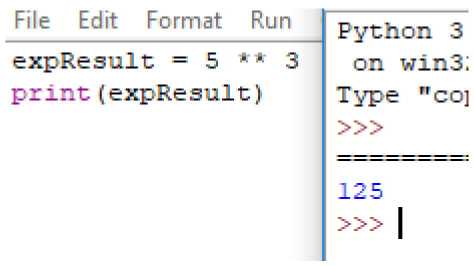
Programming Part 1: Exam-Style Questions

1. Define the term “variable”.
2. Define the term “constant”.
3. List 3 data-types.
4. State the most suitable data-type for a password.
5. Explain why your chosen data-type is the most suitable for a password.
6. State the most suitable data-type for the age of a person.
7. State the most suitable data-type for a measurement that uses decimal places.
8. Which data-type is used for TRUE or FALSE values?
9. Describe the effect of a binary left-shift by 2 places.
10. Describe the effect of a binary right-shift by 1 place.
11. What is 7 << 1?
12. What is 8 << 3?

<u>Mathematical operators:</u>

Addition: +	Example:	5 + 15 = 20
Subtraction: -	Example:	10 – 3 = 7
Multiplication: *	Example:	3 * 8 = 24
Division: /	Example:	21 / 3 = 7

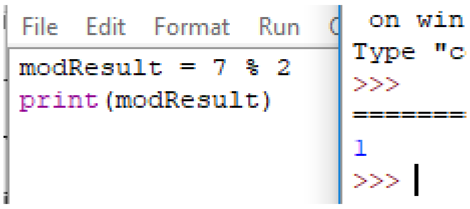
Powers (indices or exponents): EXP or ** Example: 5 EXP 3 = 125 (5 x 5 x 5 = 125)



```
expResult = 5 ** 3
print(expResult)
```

```
Python 3
 on win3
Type "co
>>>
========
125
>>> 
```

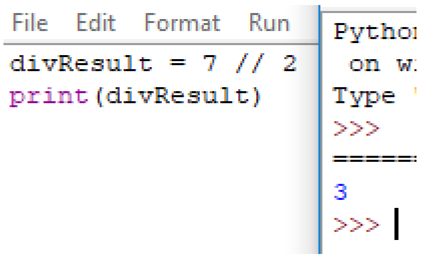
Division that only gives remainder (Modulus division): MOD or % Example: 7 MOD 2 = 1



```
modResult = 7 % 2
print(modResult)
```

```
 on win
Type "c
>>>
=======
1
>>>
```

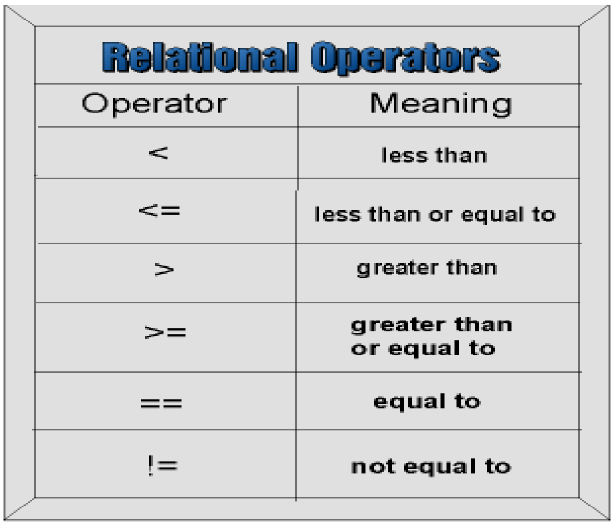
Division that only gives integer part: DIV or // Example: 7 DIV 2 = 3



```
divResult = 7 // 2
print(divResult)
```

```
Pytho
 on w
Type
>>>
=====
3
>>>
```

Logical (relational) operators:



Relational Operators	
Operator	Meaning
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
!=	not equal to

The three basic programming constructs used to control the flow of a program:

- sequence
- selection (IF/ELSE)
- iteration (count-controlled/FOR loops and condition-controlled/WHILE loops)

Data structures:

In programming data structures are ways of storing an organising data in your computer programs.

The main data structure you use in Python are arrays (lists).

Arrays store items of a similar type. For example an array might store a list of animals, or a list of usernames and passwords.

Make sure you are familiar with how to create and use arrays. See the examples below:

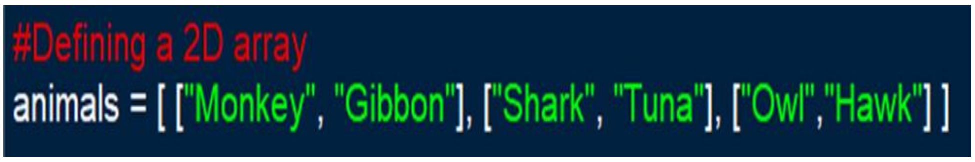
Arrays (or “lists”)

A ***data structure*** that contains *elements* of a similar type.

```
           0         1          2         3        4  ⟵ Index numbers
fruits = [“orange”, “banana”, “grape”, “apple”, “kiwi”]
```

print(fruits[0])

2D arrays are arrays inside of other arrays. 2D arrays are a bit like having rows and columns.

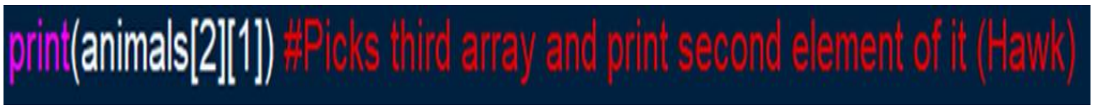


```
#Defining a 2D array
animals = [ ["Monkey", "Gibbon"], ["Shark", "Tuna"], ["Owl","Hawk"] ]
```

Arrays always start at 0, so [“Monkey, “Gibbon”] is array 0.

“Monkey” is at position 0 of the 0th array.

Typing animals[2][1] will select the 3rd array and the 2nd element of that array, giving “Hawk”.



```
print(animals[2][1]) #Picks third array and print second element of it (Hawk)
```

This might help you to understand it:

2D arrays – consist of rows and columns (arrays inside arrays).

```
                     0                        1
                0         1              0         1
users = [ [“Jim”, “password1”], [“Norah”, “abc123”] ]
```

print(users[0,1])   #Could also be written as print(users[0][1])

This would print *password1*

This example shows another 2D array in Python:

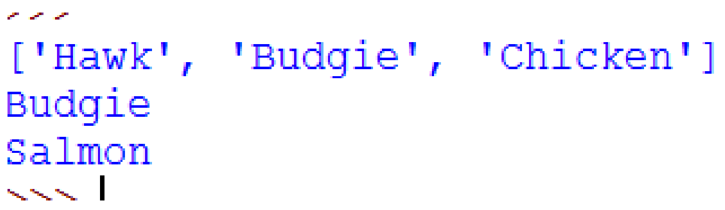
```
animals = [["Hawk", "Budgie", "Chicken"],["Trout", "Cod", "Salmon"]]

print(animals[0])

print(animals[0][1])

print(animals[1][2])
```

Results of the code above:



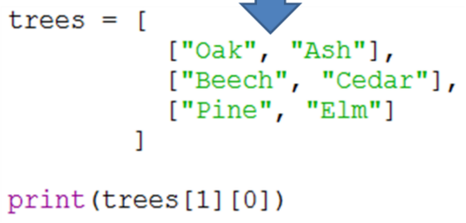
```
['Hawk', 'Budgie', 'Chicken']
Budgie
Salmon
```

2D arrays can be visualised as tables or grids.

```
trees = [ ["Oak", "Ash"], ["Beech", "Cedar"], ["Pine", "Elm"] ]
```

	0	1
0	Oak	Ash
1	Beech	Cedar
2	Pine	Elm

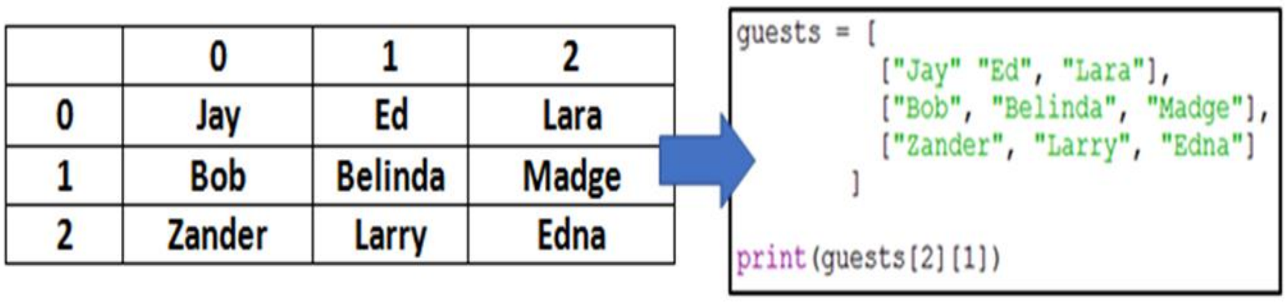
Typing your array like this may help you to see this:



```
trees = [
          ["Oak", "Ash"],
          ["Beech", "Cedar"],
          ["Pine", "Elm"]
        ]

print(trees[1][0])
```

Another example:



	0	1	2
0	Jay	Ed	Lara
1	Bob	Belinda	Madge
2	Zander	Larry	Edna

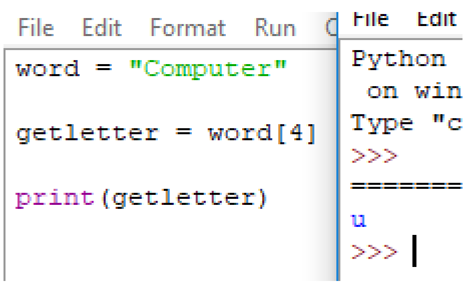
```
guests = [
          ["Jay" "Ed", "Lara"],
          ["Bob", "Belinda", "Madge"],
          ["Zander", "Larry", "Edna"]
        ]

print(guests[2][1])
```

String manipulation: Strings are stored as arrays that start at 0.

E.g. "Hello!" would be stored as ["H", "e", "l", "l", "o", "!"]

To output a single character, simply use the same syntax as you would for an array. See example:



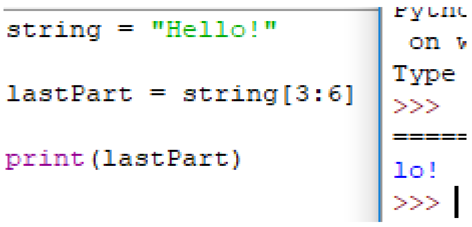
```
word = "Computer"

getletter = word[4]

print(getletter)
```

```
Python
 on win
Type "c
>>>
=======
u
>>>
```

To output a range of character, simply select the range of characters that you want to print. The example below prints the last 3 characters from “Hello!”



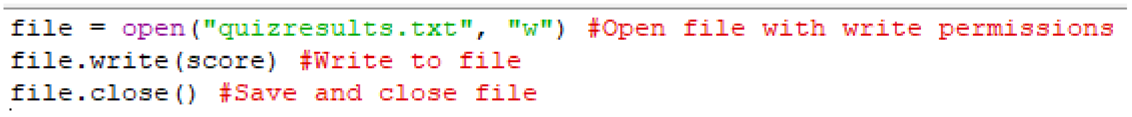
```
string = "Hello!"

lastPart = string[3:6]

print(lastPart)
```

```
Pytho
 on w
Type
>>>
=====
lo!
>>>
```

File handling in Python:



```
file = open("quizresults.txt", "w") #Open file with write permissions
file.write(score) #Write to file
file.close() #Save and close file
```

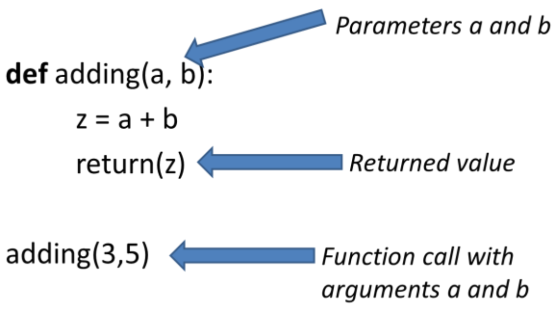
Programming Part 2: Exam-Style Questions

1. Describe what is meant by an array in programming.
2. Give one reason why arrays are useful in programming.
3. Which data-structure could be used to store a grid/table of data?
4. Write an algorithm that will store a username and password entered by the user into a text-file.
5. Write an algorithm that will search for an element in an array.

Sub-routines

Sub-routines (functions and procedures) are blocks of code that perform a particular task inside your main program. Sub-routines help to structure the code, allowing programmers to break problems down and solve them in teams, as well as making the code easier to follow and maintain.

Anatomy of a sub-routine

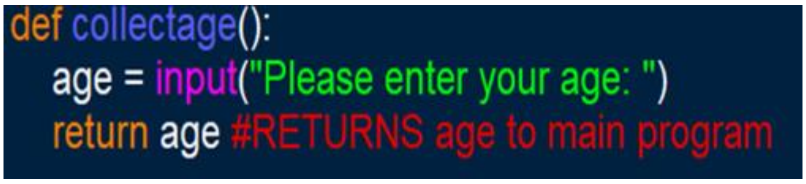


```
def adding(a, b):
    z = a + b
    return(z)

adding(3,5)
```

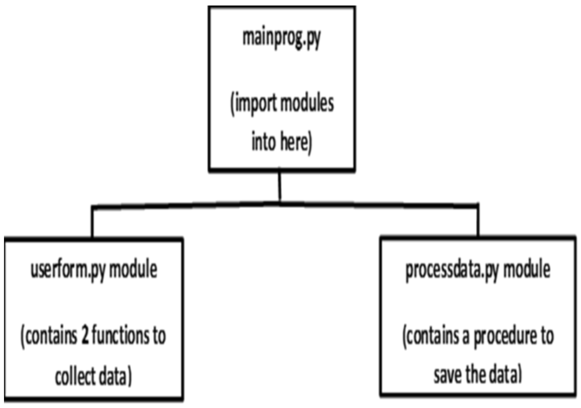
This sub-routine is a FUNCTION because it RETURNS a value

An example of a function in Python:



```
def collectage():
    age = input("Please enter your age: ")
    return age #RETURNS age to main program
```

Structure diagrams can be used to show how a program can be DECOMPOSED into separate SUB-ROUTINES. The example below shows a very simple program:


mainprog.py
(import modules into here)
userform.py module
(contains 2 functions to collect data)
processdata.py module
(contains a procedure to save the data)

Structure diagrams show how the code is structured, they are NOT the same as flowcharts!

Programming Part 3: Exam-Style Questions

1. State what is meant by a sub-routine.
2. Identify two types of sub-routines.
3. Which type of sub-routine RETURNS a value.
4. The sub-routine below add together numbers entered by the user:

```
def functionAdd(numA, numB):
    result = numA + numb
    return result
```

a) How many parameters go into the sub-routine?

b) Which type of sub-routine is this? Why?

5. Give two reasons why sub-routines are used in programming.

6. Explain what a structure diagram is used for and what it shows.

Producing robust programs (defensive design)

A robust program is a program that is fault-tolerant and difficult to break/crash. When producing code, programmers need to think about how users might misuse the program, either intentionally or unintentionally. For example, what if a program asked for a name and they user typed in a number?

Defensive design simply means that programs should be designed to be robust and to not crash if a user does something unexpected.

Defensive design considerations:

- Input sanitisation/validation – validation checks such as length checks and format checks can help to avoid causing errors.

- Planning for contingencies – having a plan in place should something go wrong.

- Anticipating misuse – assume that users will TRY to break the program.

- Authentication – using logon systems with passwords to make sure the user is who they say they are.

Testing

Software testing is an important step in making sure programs are fit for purpose and robust.

Iterative testing is testing that is carried out during production.

Final/terminal testing is carried out just before the final program is released.

Testing techniques:

- Test with erroneous data (intentionally enter data that is not valid to see how the program responds)

- Test with valid data to see if the program works as it should.

- Test with boundary data (data on the edges of what is acceptable) to see if it works as expected.

- Syntax errors should be picked up by the IDE, however logic errors will not be.

- Syntax errors are grammatical errors, e.g. missing off brackets or quote marks.

Logic errors might be for example putting the < (less than) sign when you should have put > (greater than). Logic errors will not be picked up by the IDE, but SHOULD be picked up if testing is thorough!

Code should also be maintainable, so that you and other programmers can easily understand and maintain the code.

There are four ways to make programs more maintainable:

- Comments
- Indentation (neat layout)
- Using sub-routines
- Meaningful variable names

Defensive Design and Testing: Exam-Style Questions

1. Describe what is meant by “defensive design”.
2. Why is it important to ensure that software is robust?

3. Explain, using an example, how an end-user might misuse a program.
4. Define the term “authentication”.
5. State and describe two validation techniques that could be used to ensure that users enter a valid password.
6. The following code check if a user is 18 or above before allowing them to register to use a website:

```
IF age >= 18:
    ALLOW ENTRY
ELSE
    REFUSE ENTRY
```

   a) State one piece of boundary data that could be used to test this program.

   b) State one piece of invalid (erroneous) data that could be used to test this program.

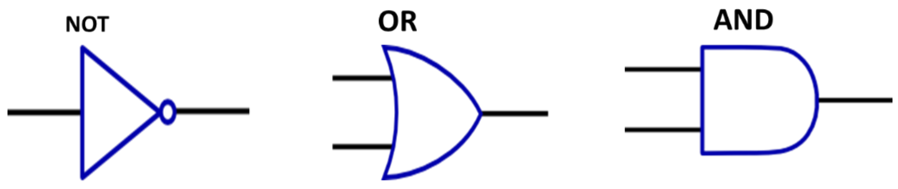
7. Describe the difference between incremental testing and final (terminal) testing.

Computational logic

Logic gates:

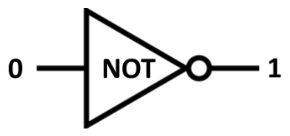
Logic gates take in one or more binary inputs and give a binary output based on the type of logic gate. CPUs are made up of billions of logic gates combined together.

You need to know the THREE main logic gates:



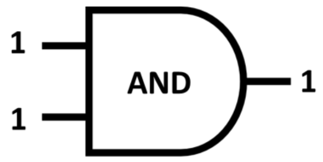
NOT gate:

Simply gives the opposite output (e.g. changes a 1 to a 0)



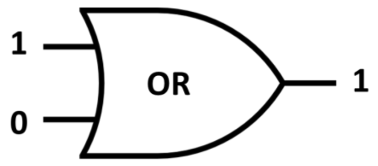
AND gate:

BOTH inputs must be 1 to give an output of 1.

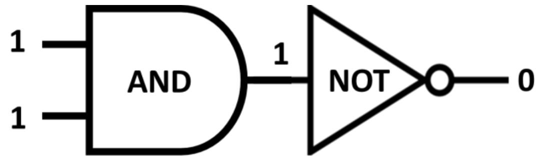


OR gate:

One OR both inputs can be 1 to give an output of 1.

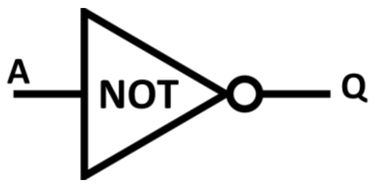


Logic diagrams (also called **logic circuits**) show how logic gates can be combined.

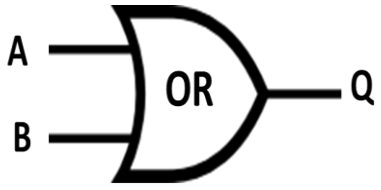


Truth tables

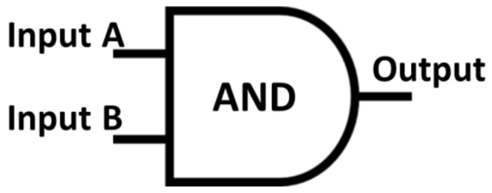
Truth tables are used to show the outputs from logic gates for given inputs. See below:



Input, A	Output, Q
0	1
1	0



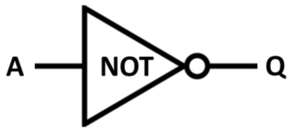
Input A	Input B	Output, Q
0	0	0
0	1	1
1	0	1
1	1	1



Input A	Input B	Output
0	0	0
0	1	0
1	0	0
1	1	1

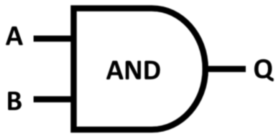
Boolean algebra can be used to represent logic gates using mathematical notation. This is shown below:

$Q = \bar{A}$ *Can also be written as* $\neg A$



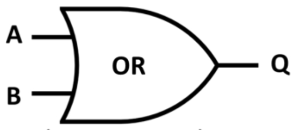
$(Q = NOT\ A)$

Q = A ∧ B



$(Q = A\ AND\ B)$

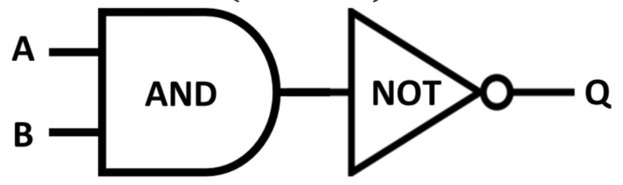
Q = A V B



(Q = A OR B)

Boolean algebra can be combined, for example:

$NOT\ (A\ AND\ B)$

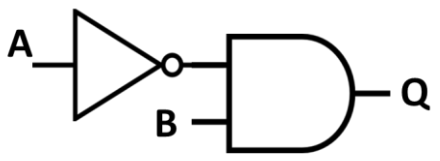


$\overline{A \wedge B}$ $\quad Also: \neg(A\ AND\ B)$

Boolean algebra follows the BIDMAS rules!

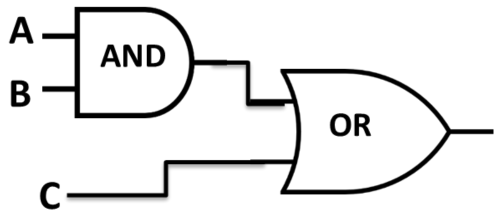
Another example:

¬A ∧ B = (NOT A) AND B =



Another example:

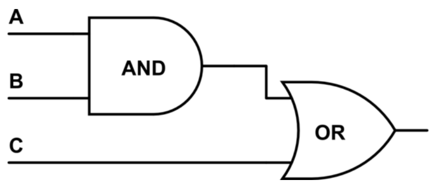
P = (A ∧ B) V C that is… P = (A AND B) OR C



Logic Gates Exam-Style Questions

Draw the logic circuits for each of these:

1. $Q = A \wedge B$
2. $Q = A\ V\ B$
3. $Q = \neg(A\ V\ B)$
4. $Q = A\ V\ B\ V\ C$
5. $Q = (A \wedge B)\ V\ C$
6. $Q = (A\ V\ B) \wedge C$
7. $Q = A\ V\ (B \wedge C)$
8. Write out in words and then write the Boolean algebraic expression:



9.

Copy and complete the truth table for an AND gate:

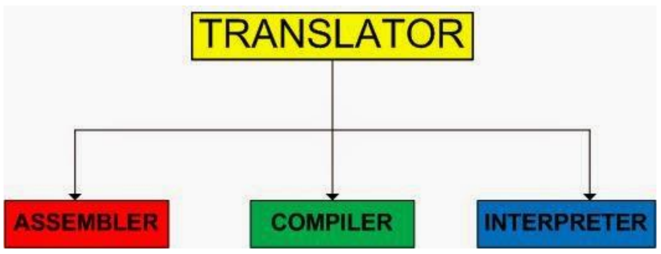
Input, A	Input, B	Output, Q
0	0	
1	0	0
0	1	
1	1	

Translators

Translators convert high-level code such as Python (which is understandable by humans), into machine code (binary) which is understood by computers.

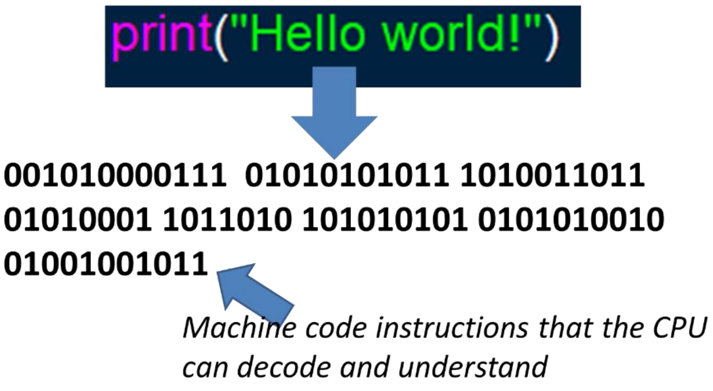
High-level languages are human-understandable because they are written in English. Low level languages are much closer to the CPU and less easy for humans to write. Both high level and low level languages need to be translated into machine code for computers to process.

Types of translator:



The **translator** converts high-level source code into machine code (binary) that the CPU can understand.

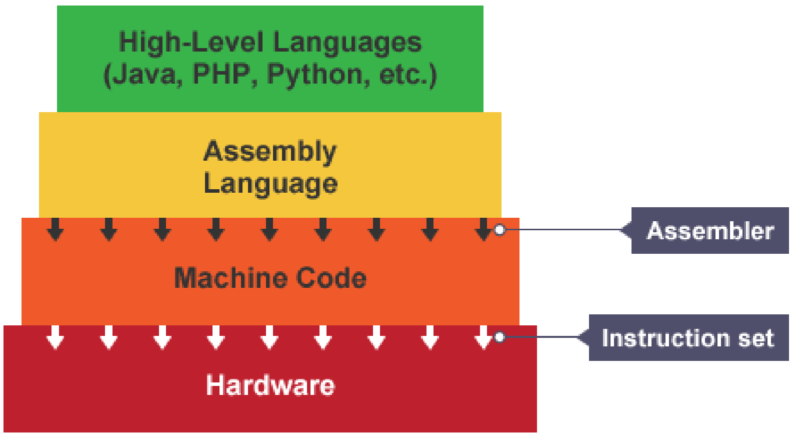
The **translator** works in the background.



Interpreter – translates code line-by-line into machine code (binary). With interpreters, each time you run the program it has to be converted into machine code.

Compiler – translates all of the code in one go and stores it as an executable file. Once the code is compiled, it can be run by the processor whenever it is needed. Most desktop software and computer games are compiled.

Assembler – translates low level assembly language into machine code. Assembly language is low level because there are fewer layers between the programming language and the hardware. In assembly language each command (or "mnemonic") that the programmer types in directly relates to an instruction. The assembler convers each command into the binary instruction.



Types of Translators: Exam-Style Questions

1. Describe what is meant by a "high-level programming language".
2. Describe the function of a translator.
3. List three types of translator.
4. Explain the key difference between an interpreter and a compiler.
5. Discuss whether an interpreter or a compiler would be better for translating source code prior to releasing to customers.
6. What is an assembly language "mnemonic", give an example.
7. Why does Python code have to be translated?

Tools of the IDE

An **Integrated Development Environment (IDE)** is software that helps programmers to write code. The features/tools of an IDE are given below:

Code highlighter – colour highlights keywords and key syntax to make the program more readable.

Run-time environment – allows programmers to run and test the code before release.

Auto-completion – Suggests programming key words and sub-routine names and auto-completes them.

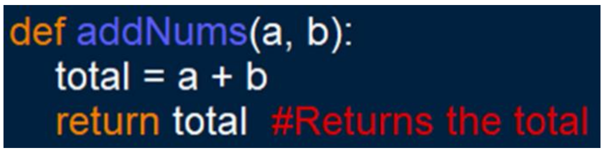
Translator – Translates the high-level source code into machine code.

Editor – Allows the programmer to type and edit the code.

Error diagnostics/debugger – Detects syntax errors and tells the programmer which line the error is on.

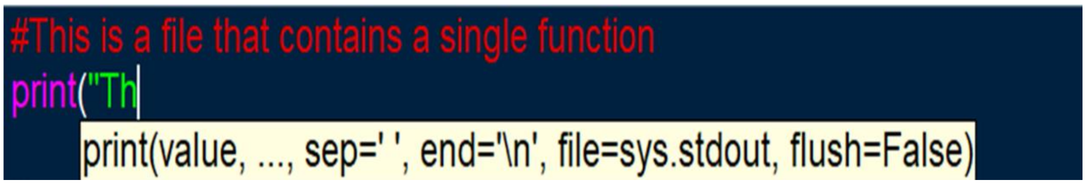
Think CRATEE!

Code highlighter in action:



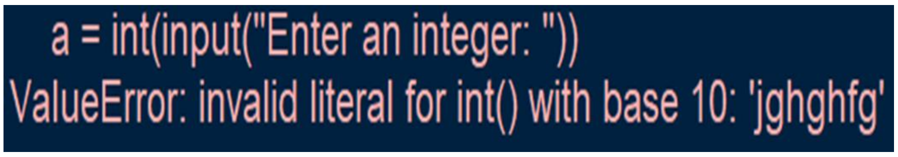
```
def addNums(a, b):
    total = a + b
    return total  #Returns the total
```

Auto-completion in action:



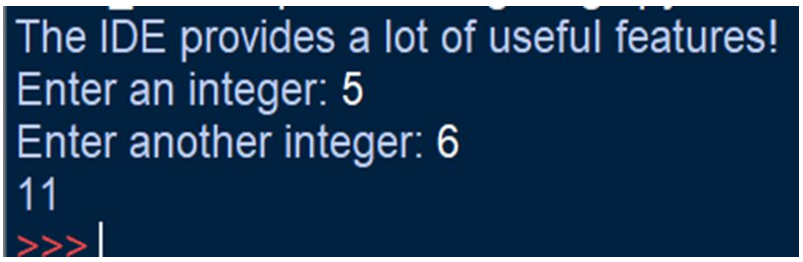
```
#This is a file that contains a single function
print("Th
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)
```

Error-diagnostics/debugging in action:



```
    a = int(input("Enter an integer: "))
ValueError: invalid literal for int() with base 10: 'jghghfg'
```

Run-time environment in action (allowing us to run and test the program):



```
The IDE provides a lot of useful features!
Enter an integer: 5
Enter another integer: 6
11
>>>
```

Tools and Features of the IDE Exam-Style Questions

1. State what IDE stands for.
2. Describe what an IDE is.
3. List three features/tools of an IDE that helps programmers to write code.
4. Describe how the debugger/error diagnostic feature helps programmers to identify and fix syntax errors. Use an example.

Data representation

All data, including images, sound, video and text must be represented as binary (1s and 0s) because this is all the computer can understand. Processors are made up on tiny transistor switches that can either be ON or OFF (1 or 0). The analog world must be sampled and converted to binary for storage and processing.

Binary to denary

To convert binary to denary, simply add up the headings with a "1" underneath them:

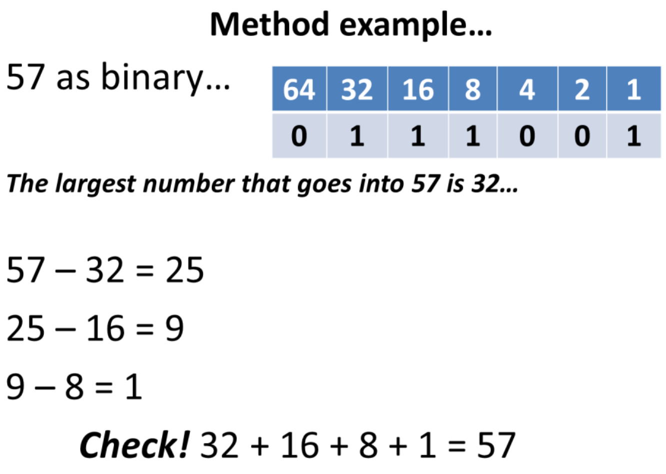
128	64	32	16	8	4	2	1
0	0	0	0	0	1	0	1

= 5

Denary to binary

To convert denary to binary, find the largest of the binary headings (128, 64… etc.) that will go into the number. Subtract is away from the number and put a "1" under the heading. Then find the next largest number, subtract it away, put a "1", repeat until you get to 0.

The following example shows how to convert 57 into binary:



Method example…

57 as binary…

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

The largest number that goes into 57 is 32…

57 – 32 = 25

25 – 16 = 9

9 – 8 = 1

Check! 32 + 16 + 8 + 1 = 57

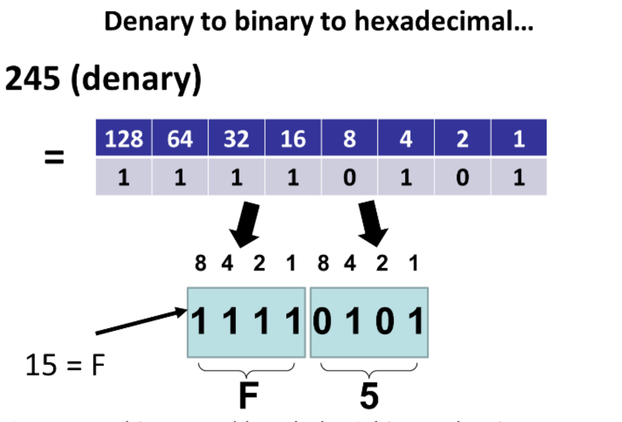
Denary to binary to hexadecimal

You will need to learn and practise the method. It is important to show workings in an exam.

To go from denary ("our numbers") to hexadecimal, convert to binary first. You will also need to write out the denary-hex table:

Conversion "table":

A = 10 | B = 11 | C = 12 | D = 13 | E = 14 | F = 15



Denary to binary to hexadecimal…

245 (denary)

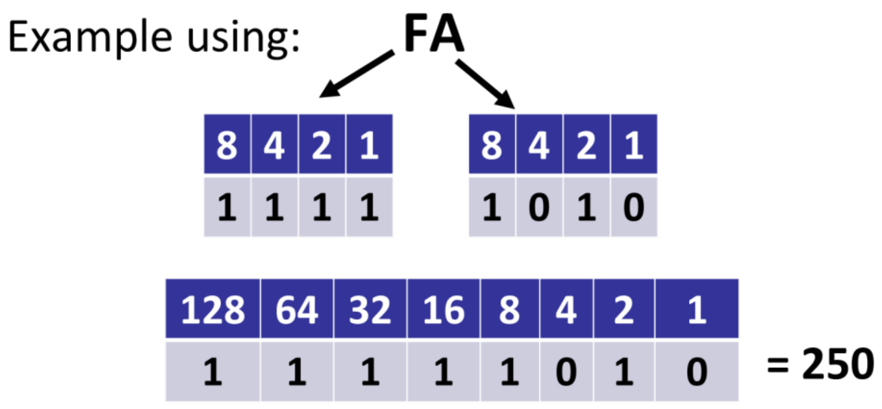
=

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

Convert to binary and break the 8-bit number into two 4-bit numbers. Find the value in hexadecimal by using the conversion table.

Hexadecimal to binary to denary

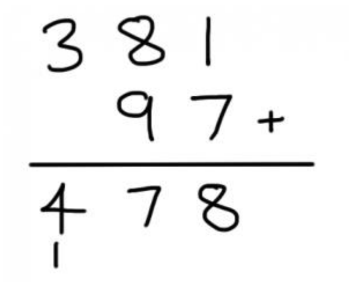
To convert hexadecimal to denary, convert to binary first. It’s basically the reverse of going from denary to hexadecimal:



Binary addition

Standard denary column addition is something you should be familiar with from primary school:

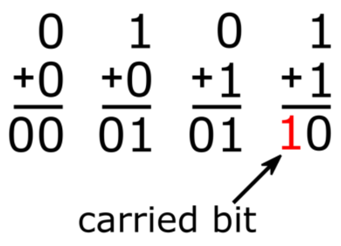
If the two added numbers is above 9, then it carries across to the next column.



Binary addition is similarly fairly simple, so long as you follow the rules. You need to carry the 1 if it’s a 10 or 11. There are a number of YouTube videos to help you with this.

Remember:

0 + 0 = 0 0 + 1 = 1 1 + 0 = 1 1 + 1 = 10 1 + 1 + 1 = 11



Example:

Number 1	0	0	0	1	1	1	0	1
Number 2	1	0	0	0	1	0	1	1
Answer	1	0	1	0	1	0	0	0
Carry			1	1	1	1	1	

If you add together two 8-bit numbers and end up with 9-bits, this is an overflow error because the computer will cut off the last bit and give the wrong answer.

Characters

Characters are simply letters, numbers and symbols. Strings are a data-type that store lines of characters, e.g. usernames and passwords.

A character set is the set of all characters (letters, numbers, symbols) that a computer can understand.

ASCII and Unicode are two methods of encoding and storing characters. Every character is given a sequence of binary to represent it.

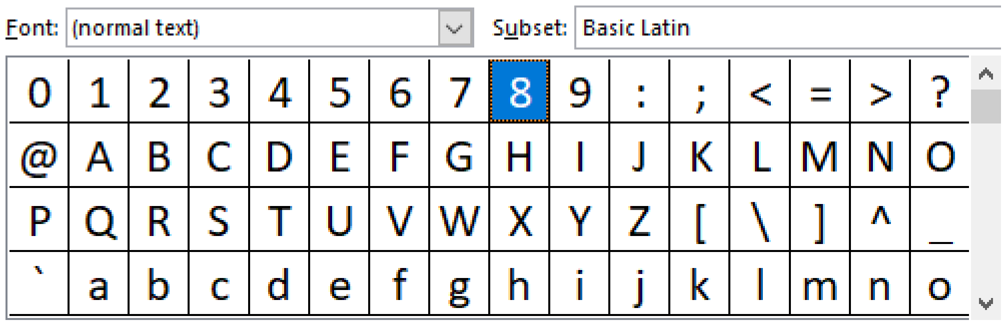
ASCII used 7-bits which could store $2^7 = 128$ characters. ASCII was later upgraded to 8-bits which could store $2^8 = 256$ characters. This was OK during the 1950s/60s when only America and Britain had computers and they so people could type English. If you include English uppercase and lowercase letters and symbols and numbers this is only just enough.

Letter	**ASCII character code**
A	01000001
B	01000010
C	01000011

Unicode (16-bit and later 32-bit) was brought in to allow other languages, with all their various letters and symbols to be able to be represented in a computer system.

The higher the number of bits, the larger the set of characters that can be represented.

Example of PART OF a character set:

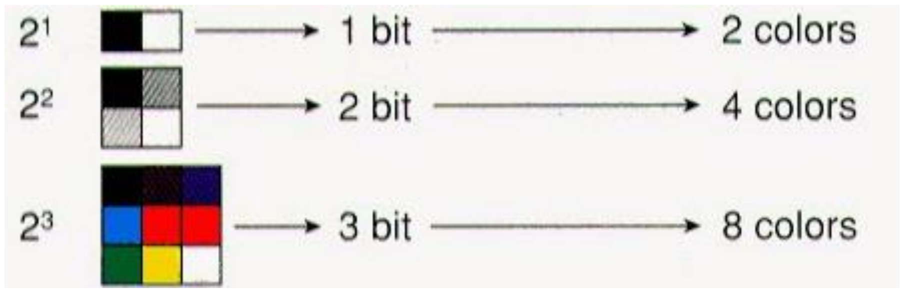


Font: (normal text) Subset: Basic Latin

0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o

Images

Images are represented as a series of pixels which are encoded with binary. Each pixel can only be one colour, the colour is given by the binary code. The higher the bit depth (colour depth) the more colours can be represented.



The resolution is the number of bits per area of an image.

Higher colour depth (bit-depth) and higher resolution = better quality, but larger file size.

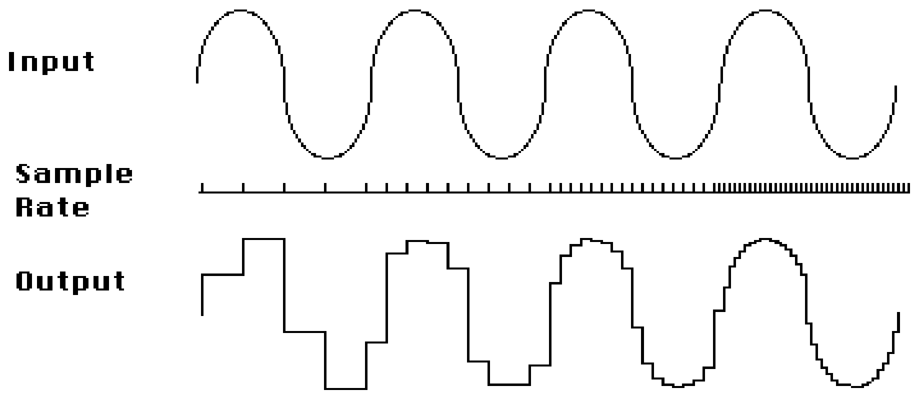
Sound

Sound/audio also has to be captured, converted to binary and stored. This is carried out in a series of steps, the example below should help you to understand the process:

- Musician plays guitar.
- Sound vibrations travel through the air and are picked up by the microphone.
- The microphone converts the vibrations into electrical signals.
- The electrical signals are sampled a fixed number of times per second.
- Each sample is stored as a series of 1s and 0s.

Sample rate/sampling frequency:

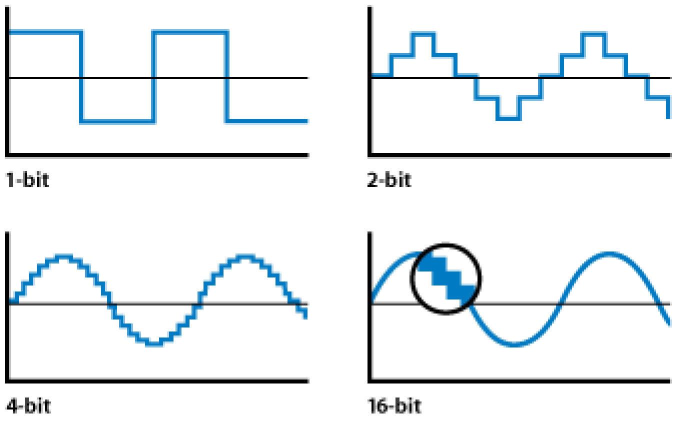
The number of samples of the sound taken per second. The image below shows that as the sampling frequency increases the analog wave is more accurately represented:



Bit depth or "sample size":

The number of frequencies/pitches that can be represented.

The higher the sampling frequency/sample rate and bit-depth/sample size the better the file will represent the original analog sound.



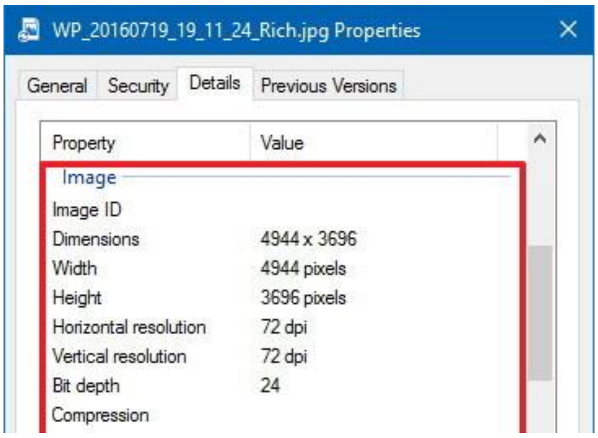
Bit rate:

The amount of data per second that is required to stream/play the sound back again. Obviously, the higher the sampling frequency and the larger the bit-depth/sample size the more data will need to be processed per second to play the audio file (.mp3).

Metadata:

Image files and sound files (and other files) also contain metadata. Metadata is data about data. For example, an image file might contain metadata that gives the resolution and colour depth.

The file type, author, creation date, sample rate, etc. are all types of metadata. Computers need metadata to understand how to display the file properly.



Property	Value
Image	
Image ID	
Dimensions	4944 x 3696
Width	4944 pixels
Height	3696 pixels
Horizontal resolution	72 dpi
Vertical resolution	72 dpi
Bit depth	24
Compression	

Data Representation Exam-Style Questions

1. Define the term “character set”.
2. Explain why modern computer systems use Unicode, instead of ASCII.
3. State what is meant by the resolution of an image.
4. Describe the effect of increasing the colour depth (bit depth) of an image.
5. Explain how increasing the sample rate improves the quality of an audio (sound) file.
6. Explain what is meant by the bit depth of an audio recording.
7. Explain what is meant by bit rate (playback rate).
8. List two pieces of metadata that might be stored along with an image file.
9. Convert 1001 to denary.
10. Convert the hexadecimal character E to binary and then to denary.
11. Convert 5C (hexadecimal) to binary and then to denary. Show ALL workings.
12. Convert F3 (hexadecimal) to binary and then to denary. Show ALL workings.