Crucial Knowledge 1.1 : Systems Architecture

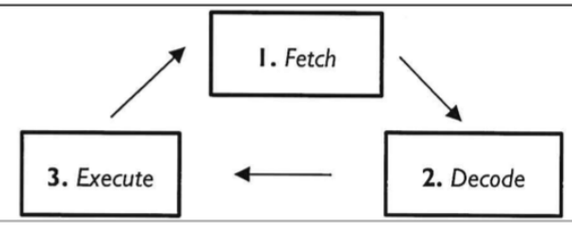
1. The purpose of the CPU

The purpose of the CPU	To manage basic operations of the computer. To be the 'brains' of the computer
The main components of the CPU	Control Unit. Arithmetic Logic Unit. Registers. Cache
Von Neumann Architecture	The architecture that allows for the storage of instructions and data in the same location
The FDE Cycle	The cycle the CPU continuously carries out to process instructions
Binary	The number system used to store instructions and data in the computer
The role of a register in the CPU	It is a place to temporarily hold data and instructions as they are being processed by the CPU.
The PC	The Program Counter keeps the address of the next instruction to be processed
The MAR	The Memory Address Register is used to tell the CPU where to locate data in Main Memory
The MDR	The Memory Data Register is used to store data that is fetched from Main Memory
The ACC	The Accumulator stores results of logic operations and calculations used during processing

2. Common CPU Components and their Function

The Control Unit has two functions	(1) Sending signals to control the flow of data and instructions, and (2) decoding instructions
Cache memory	A small section of extremely fast memory used to store commonly used instructions and data. It is useful as the CPU can access the (fast) cache directly. L1 cache is closest to the CPU, L3 cache furthest
The ALU has the following functions	It carries out mathematical operations / logical operations / shifting operations on data; for example multiplication, division, logical comparisons
An Address	This is a location in he Main Memory (RAM) that stores data or instructions in the Von Neumann Architecture
Buses	Transfer information between the CPU and Main Memory (and other places). For example the Address bus carries memory addresses between the CPU and the RAM

3. The F-D-E (Fetch Decode Execute) Cycle

The F-D-E Cycle repeatedly cycles	
The Fetch Stage	The address is generated by the Program Counter (PC) and is carried to the Memory Address Register (MAR) using the Address Bus. The PC then updates and stores the next memory address, ready for the next round of the cycle. The data or instruction that is in that memory location is placed on the data bus and carried to the processor and is stored in the Memory Data Register (MDR)
The Decode Stage	The data or instruction is then the Memory Data Register (MDR). decoded to find out if it is a piece of data or if it is an instruction to do something such as ADD, STORE, SWITCH, REPEAT etc.
The Execute Stage	The CPU performs the actions required by the instruction. If it is an instruction to control input or output devices the Control Unit will execute the instruction. If it is a calculation then the Arithmetic and Logic Unit (ALU) will execute the instruction. The results of any calculations are recorded in the Accumulator.

4. Performance of the CPU

Cores	CPUs with multiple cores have more power to run multiple programs at the same time.
Clock Speed	The clock speed describes how fast the CPU can run. This is measured in megahertz (MHz) or gigahertz (GHz) and shows how many fetch-execute cycles the CPU can deal with in a second.
Cache Size	The more data that can be held in the cache, the shorter the trips the electric pulses need to make so this speeds up the processing time of each of those billions of electrical signals, making the computer noticeable faster overall.

5. Embedded Systems

Definition	A computer system which forms part of an electronic device	Reasons	They are cheaper to make and smaller than a General Purpose Computer
Re-programmable	Not for different purposes but firmware can sometimes be upgraded	Examples	Washing machine, Smart Oven, Car Engine, Pacemaker

Crucial Knowledge 1.2.1 : Primary and Secondary Storage

1. The purpose of RAM and ROM in a Computer System

The purpose of RAM	RAM is the main memory (also called primary storage) for storing data and programs while they are in use
The purpose of ROM	ROM stores the boot sequence, which is a set of instructions that the computer executes every time it is switched on. ROM is essential since if loads the operating system
We use RAM rather than Secondary Storage	The RAM can be accessed at a much higher speed than the secondary storage. If the CPU was having to communicate directly with secondary storage for the F-D-E cycle the computer would be incredibly slow
Volatility	ROM is non-volatile (it keeps its contents when the power is turned off). RAM is volatile (it loses its contents when the power is turned off)
Primary Storage Devices	Primary storage devices are internal to the system and are the fastest of the memory/storage device category. Typically, primary storage devices have an instance of all the data and applications currently in use or being processed. The computer fetches and keeps the data and files it in the primary storage device until the process is completed or data is no longer required. RAM, ROM, Graphics Card RAM, cache and registers are common examples of primary storage devices
Increasing RAM	This can speed the computer up since there is less need for virtual memory

2. The Need for Virtual Memory

Definition of virtual memory	A temporary storage space taken up on a secondary storage device (e.g. hard disk) to allow more space for running programs and data than can fit in primary storage (RAM)
Use of virtual memory	Open applications / data that are not in current use are 'paged' out to the secondary storage. When they are need they are 'paged' back into primary memory
Advantage of virtual memory	Having virtual memory available allows a computer to run more programs at the same time, or to run larger programs; or to work with much larger amounts of data than could fit in the primary storage (main memory / RAM)
Disadvantage of virtual memory	It is relatively slow compared with RAM. The need to page data in and out of the secondary storage device slows down the computer. It can also lead to 'disk thrashing'

3. Secondary Storage

Difference from primary storage	Primary storage (e.g. RAM, cache) is volatile. Secondary storage is non-volatile. It retains its data when the power is switched off
Cache memory	A small section of extremely fast memory used to store commonly used instructions and data. It is useful as the CPU can access the (fast) cache directly. L1 cache is closest to the CPU, L3 cache furthest
ROM as secondary storage	Not really. ROM is read only. Secondary storage generally needs to be written to as well as read from

4. Common types of storage

Optical	The surface of a CD is covered in microscopic dots. A laser would skim across the surface reading these. As the laser passes over, the pattern on the surface is picked up. If the laser hits a dot it is reflected differently to if there were no dot present. Examples : CD/CDR/CDRW/DVD/BluRay
Magnetic	Magnetic hard drives use silver coloured disks which are covered on both sides with a magnetic film divided into billions of tiny areas. Each one of those areas can be independently magnetised (to store a 1) or demagnetised (to store a 0). The read/write heads would flicker quickly over the surface as it reads and writes the data. Several platters would be installed in one hard drive to give greater storage capacity. Examples : Hard Disk Drive / DAT / Tape Drive / Cassette
Solid State	Solid-state secondary storage does not have any moving parts. Solid state secondary storage stores data using circuit chips. They are sometimes called flash drives. Examples : USB drives / SD Cards / SSD Drives

5. Considerations for the Most Suitable Storage Device

Capacity	How much data needs to be stored
Speed	How quickly can the data be stored. How quickly does it need to be read
Portability	Does the device need to be transported? Are weight and size important
Reliability	Is it mission critical? Will it be used over and over again?
Cost	How expensive is the media per byte of storage

6. Typical Uses

Optical	Read only distribution on a large scale (CD/DVD). Relatively small capacity
Magnetic	High data capacity. Reasonably fast. Low cost. Cloud storage on server farms
Solid State	Low power. Small. Rugged. Silent. Very fast. Medium data capacity

Crucial Knowledge 1.2.2 : Data Storage

1. Data units

Bit (b)	The smallest unit of data. 0 or 1
Nibble (N)	4 bits
Byte (B)	8 bits (note the difference between b and B)
Kilobyte (KB)	1000 bytes. Note KB is different from Kb
Megabyte (MB)	1000 KB
Gigabyte (GB)	1000 MB
Terabyte (TB)	1000 GB
Petabyte (PB)	1000 TB

2. Conversions

Binary to Denary
Denary to Binary
Hexadecimal to Denary
Denary to Hexadecimal
Binary to Hexadecimal
Hexadecimal to Binary
Left Binary Shift
Right Binary Shift

3. Operations

Binary addition	You should arrange the two binary numbers above each other so that the columns line up. Start on the rightmost digit and add them. If there are any carries, write them down next to the next left column.
Overflow	If the answer to the left column results in a carry, this is known as an overflow and it causes an overflow error. This can cause problems if a computer program hasn't been written to handle overflows.
Left Binary Shift	Make the number longer, and therefore bigger. Each place it shifts will double the value. A binary left shift of one place (<<1) will double the value, a binary left shift of two places (<<2) with quadruple.
Right Binary Shift	Make the number shorter, and smaller. The right most digit is "lost", so we forget about it. A binary right shift of one place (written as >>1) halves the number, and a binary right shift of two places (>>2) will quarter it.

4. Characters

Individual Characters	Each character is assigned an individual binary code to represent it. The number of bits depends on the 'encoding' used
Character Set	The name given to a collection of characters matching to binary codes. There are many examples.
Choice of Character Set	A character set encoded with more bits allows more characters. This is useful for accents, symbols, emojis, other languages (e.g. Chinese)

5. Examples of Character Sets

ASCII	7-bits to represent characters allowing 127 characters to be represented
Unicode	16 / 24 / 32 bits. Covers many modern and historic languages, as well as lots of symbols which are used in maths and other specialist areas

6. Images

Pixel	The smallest element of a bitmap image. Pixels desk
Vector vs Bitmap	A vector image describes the lines and shapes. A bitmap image consists of rows of coloured dots.
Colour Depth	The number of bits used to represent each pixel in a bitmap image. An 8 bit image can show 2^8 or 256 colours.
Resolution	In a bitmap image resolution is measured in DPI (dots per inch). The higher the resolution the better the picture quality
Metadata	Data that is saved before and after the image to tell the computer how to decode the image. It includes the size in pixels (width x height), the colour depth, the resolution, the GPS location of where the image was taken, etc.
Image size	The size of an image is width x height x colour depth (+10% for metadata)
Factors	Greater colour depth and/or greater resolution will make the file size bigger, and improve the quality of the image; and vice versa

7. Sound

Analogue / Digital	Analogue sound waves must be converted into digital sound waves by taking a sample of the sound at set intervals. This is because computers can only work with digital 'numbers', and not analogue 'sound'
Sample rate	Number of times analogue signal is sampled per second. Measured in Hertz
Bit depth	Number of bits used per sample. Sometimes known as sample resolution
File size	Sample rate x sample resolution x seconds
Factors	Larger sample rate and/or bit depth will make the file size bigger and improve the playback quality; and vice versa. Also, making the duration of the recording longer will make the file size bigger, and vice versa

8. Compression

Compression	Compression is when a file is encoded so it uses fewer bits than the original file format
Lossless compression	Gets rid of unnecessary data to re-present data without losing any information. This process is reversible
Lossy compression	Gets rid of the least essential data. This is an irreversible process: once data is lost it can't be recovered

Crucial Knowledge 1.3.1 : Networks and Network Topologies

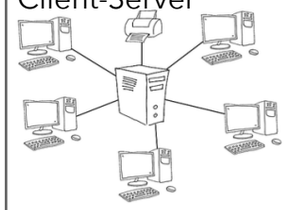
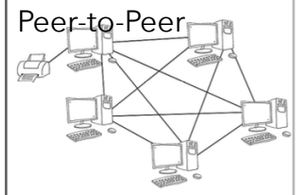
1. Types of Networks

Network	A set of connected computers and other devices (e.g. printers, phones, HomeKit devices) for the purpose of sharing resources
LAN	Local Area Network. Covers a small geographical area (a home, a school, etc.) The infrastructure is often owned by the individual / organisation
WAN	Wide Area Network. Covers a large geographical area. WANs are made up of LANs joined together. The infrastructure is often owned by a Telecoms or other company rater than the individual
Advantages to using a LAN	• Resources (files, etc.) and devices (printers, etc.) can be easily shared across the network • Computers can be configured with the same 'image' so you have the same programs and access to your data from any computer (like in school) • You can control devices (e.g. HomeKit)
Disadvantages to using a LAN	• Security. Malware can spread across a network • Complexity of setting up and maintaining

2. Factors affecting performance of a network

Latency	You can get bottlenecks in parts of your network, either because of a faulty switch, or due to the design of your network. Latency is the term used describe the time it takes data to travel from one designated point to another on the network
Bandwidth	The maximum amount of data transmitted over an internet or LAN connection in a given amount of time.
Transmission Media	WiFi generally has less bandwidth than wired connections. Wired connections (ethernet) can be different speeds (10Mbps, 100Mbps, Gigabit). Switches and routers also have maximum speeds
Concurrent Users	The more users there are on a network the more data is likely being transmitted. This means it can take longer as you have to wait your turn for your packets to travel across the network

3. Network Types

Client-Server 	The network relies on a central server and all the clients (devices) request services from the server such as print services, file services etc. Additional hardware is needed in this type of network: a server. All files can be stored and backed-up centrally on a server which means workers can access files from any computer on the network and the computers can also be updated centrally.
Peer-to-Peer 	All computers have equal status and any computer can act as a client and a server—even at the same time. All computers can request and provide network services. For example, any computer can use a resource physically connected to a different computer. There is no need to buy a dedicated server

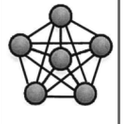
4. Required Hardware

NIC	The Network Interface Card is in each computer/devices and allows connection to other devices on the network. It can allow wired connections, wireless connections, or both
Transmission Media	What connects the computer/devices to each other. Copper cables, fibre optic cables, wireless signals
Switch	A device on the network that receives signals from a computer/device and transmits the signal to its intended recipient
Router	A device used to connect different networks together. For example a home LAN to the internet, or a fibre optic cable to a home WiFi network
WAP	A Wireless Access Point is a device that receives and transmits wireless signals on the network. Often connected to rest of the network by cables

5. The Internet

The Internet	The Internet is a global collection of interconnected networks
DNS	The Domain Name Server is a large directory allowing the Internet Service Provider (ISP) to look up the correct IP address for the desired website
Hosting	If you don't own your own servers and host your website yourself you can use a company to do it for you. They will monitor and maintain their servers they are renting you space on
The Cloud	Data can be stored 'in the cloud'. This means on servers (in server farms) run by big companies. The data can be accessed from anywhere
Web Servers and Clients	Servers provide services (e.g. Web server -> Web pages, File server -> file storage/retrieval). Clients request / use services from a server

6. Star and Mesh Topologies

Star Network	Cheaper than mesh network. Less cabling. Easy to add devices BUT total reliance on central node. If it fails whole network fails		Mesh Network	Full or partial. More cabling than star. Costs more to install. Harder to add a device. Harder to maintain BUT no Single Point of Failure	

Crucial Knowledge 1.3.2 : Protocols and Layers

1. Modes of Connection

Wired	Ethernet is a set of standards (protocols) for how data is transmitted over a wired local area network. It is the most common set of protocols. Data is transmitted in frames
Inside an Ethernet 'frame'	• Preamble of bits used to synchronise transmission • Start frame delimiter to signify start of data part of the frame • Source and destination MAC address • The actual data • Error checking information (cyclic redundancy check - CRC)
Wi-Fi	Wi-Fi is a means of allowing computers, smartphones, or other devices to connect to the Internet or communicate with one another wirelessly within a particular area. It has a range of about 100m, takes quite a lot of power (relatively), and has a high bandwidth (but less than a wired connection)
Wi-Fi advantages and disadvantages	• Users can move around freely • Easier to set up, and less expensive than wired • Speeds are slower than wired networks • Relies on signal strength to the wireless access point (WAP) • Signal can be obstructed • Less secure than wired networks
Bluetooth	Bluetooth is a standard for the short-range wireless interconnection of mobile phones, computers, and other electronic devices. It has a range of about 10m, takes very little power, and has a relatively low bandwidth

2. Wireless Encryption

SSID	Wireless networks are identified by a unique "Service Set Identifier" (SSID). Can be invisible/visible and have a password. The SSID has to be used by all devices which want to connect to that network.
Encryption	Data is encrypted by scrambling the data into cipher text using a "master key" created from the SSID of the network and the password. Data is decrypted by the receiver using the same master key, so this key is not transmitted. Protocols used for wireless encryption include WEP, WPA, WPA2.

3. IP and MAC Addresses

MAC address	Every device on a network has a Network Interface Card (NIC). Every NIC (in the world) has a unique Media Access Control (MAC) address. It is used to route frames on a LAN
IP address	IP Addressing is used to route frames on a WAN (called packets). Every device on the internet has a unique IP (Internet Protocol) address which is assigned to the device by a server. Two main standards (IPv4 and IPv6)
Internal and External IP Addresses	A router will have a unique WAN facing IP address and a LAN facing IP address. Often all devices on a LAN (with unique internal IP addresses) will share a single external IP address

4. Standards

Definition	A set of specifications for hardware/software. Enables products to be compatible with each other and interact with each other
ASCII/Unicode	Character set standards
IEEE	Computer cables standards
HTML	Standard for creating websites
PNG, GIF, MP3	Standards for documents, images, sounds, videos, etc.

5. Common Protocols

TCP/IP	Transmission Control Protocol/Internet Protocol. Used to communicate over LANs and WANs
HTTP / HTTPS	Hypertext Transfer Protocol (secure). Used for webpage requests
FTP / FTPS	File Transfer Protocol (secure). Used for file transfers
POP	Post Office Protocol. Used for receiving e-mail. Downloads e-mail from the server to your device and deletes it from the server
IMAP	Internet Message Access Protocol. Used for receiving e-mail. Keeps e-mails on the server. This allows your device to stay in sync with the server
POP vs IMAP	POP you have your mail on one device since it is deleted from the server. IMAP each device syncs to server so your mail can be on multiple devices
SMTP	Simple Mail Transfer Protocol. Transfers outgoing emails from one server to another / from a email client to a sever

6. Layers

Concept	The concept of layering is to divide the complex task of networking into smaller, simpler tasks that work with each other.
Responsibility	The hardware and/or software for each layer has a defined responsibility. Each layer provides a service to the layer above it
Advantages	Reduces the complexity of the problem into manageable sub-problems. Devices can be manufactured to operates at a particular layer. Products from different vendors will work together.

Crucial Knowledge 1.4 : Network Security

1. Forms of Attack

Malware	Software written in order to infect computers and commit crimes e.g. fraud or identify theft. Malware exploits vulnerabilities in software
Types of Malware	Malware is term that covers (among other things) viruses, trojans, worms, ransomware, spyware and adware
Phishing	Online fraud technique used by criminals. It is designed to get you to give away personal information such as usernames, passwords, bank details, credit card details… Achieved by disguising as a trustworthy source in an electronic communication, e.g. an email or fake website.
Brute Force Attack	A trial and error method used to decode encrypted data (such as passwords). Uses every combination until it hits upon the correct one.
DOS Attack	Denial of Service attack. Floods a server with useless traffic causing the server to become overloaded and unavailable
DDOS Attack	Distributed Denial of Service Attack. Using multiple computers (zombies) in a Botnet to undertake a DOS attack
Data Interception and Theft	Stealing information from an unknowing victim's computer in order to get confidential information, or to compromise their privacy. E.g. to sniff usernames and passwords
SQL Injection	A technique used to view or change data in a database by inserting additional code into a text input box, creating a different SQL command
Zero Day Attack	An attack using an unknown and undocumented vulnerability in software code (unknown to the code owner)

2. Threats posed to Networks

Malware	• Files are deleted, become corrupt or are encrypted. • Computers crash, reboot spontaneously and slow down. • Internet connections become slow. • Keyboard inputs are logged and sent to hackers.
Phishing	• Accessing a victim's account to withdraw money, or purchase merchandise and services. • Open bank accounts, credit cards, cashing illegitimate cheques. • Gain access to high value corporate data. • Financial services can blacklist the company
Brute Force Attack	• Theft of data. • Access to corporate systems.
(D)DOS Attack	• Loss of access to a service for customers • Lost revenue • Lower productivity • Damage to reputation
Data Interception and Theft	• Usernames and passwords compromised • Disclosure / theft of corporate data
SQL Injection	• Contents of databases can be output, revealing private data. • Data in the database can be amended or deleted. • New rogue records can be added to the database.
People	Many system vulnerabilities are caused by people being careless: • Not installing operating system updates. • Not keeping anti-malware up to date. • Not locking doors to computer rooms. • Not logging off or locking their computer. • Leaving printouts on desks. • Writing passwords down on sticky notes attached to computers. • Sharing passwords. • Losing memory sticks / laptops. • Not applying security to wireless networks. • Not encrypting data.

3. Identifying and Preventing Vulnerabilities

Malware	• Security software (Spam filter, Anti-virus, Anti-spyware, Anti-spam) • Enabling OS and security software updates. • Staff training • Backup files regularly onto removable media.
Phishing	• Strong security software. • Staff training: awareness of spotting fake emails and websites. • Staff training: not disclosing personal or corporate information. • Staff training: disabling browser pop-ups.
Brute Force Attack	• Network lockout policy, Using progressive delays. • Staff training
(D)DOS Attack	• Strong firewall and packet filtering • Properly configuring servers and auditing and monitoring systems
Data Interception and Theft	• Encryption and using virtual networks • Staff training and computer use policies
SQL Injection	• Validation on text boxes • Database permissions

Crucial Knowledge 1.5 : Systems Software

1. Definitions

Systems Software	Systems Software is the software used to control the hardware of the computer. It is contrasted to application software which is used to enable the user to perform tasks and create content and products
Operating System	An operating system is a piece of system software that communicates with the hardware of the computer and allows other programs to run. It is comprised of system software, or the fundamental files your computer needs to boot up and function
Peripherals	Peripherals are controlled by software called device drivers. Standard drivers (mouse and keyboard) are included in the operating system, however more specialist peripherals may need drivers programmed by the manufacturer which convert signals into machine code and are installed separately
Utility Software	Utilities are programs that are installed to perform a specific function, usually to improve the efficiency or security of a computer system

2. The Function of Operating Systems

What does an Operating system do?	An operating system manages all of the software and hardware on the computer. Most of the time, there are several different computer programs running at the same time, and they all need to access your computer's central processing unit (CPU), memory, and storage. The OS co-ordinates this activity
Interaction	A user interacts with the computer by means of an interface provided by the operating system

3. Types of Interface

GUI	A Graphical User Interface provides windows, icons, menus, (mouse or other) pointer… Sometimes calls WIMP. It is visual, interactive, and intuitive. Optimised for mouse/touch input
CLI	A Command Line Interface is text based. It uses less resources than a GUI. It is more efficient but harder to learn. Often repetitive processes can be automated with scripts
Menu	A Menu Interface presents successive menus to the user with options to choose at each stage. Often used with buttons on a keypad. (Think calculator when you press the 'MENU' button)
Natural Language	A Natural Language Interface responds to questions in a spoken language. They are not always reliable but are improving all the time. (Think Siri or Alexa)

4. Features Often Provided by an Operating System

Multitasking	Running multiple applications at the same time by giving each application a small time-slice of processor time. This allows more than one program to be held in memory at a time, and data shared between them such as copy and paste. It also enables you to listen to music on your PC at the same time as word processing for example
Memory Management	When programs are loaded, the operating system decides where they are held in memory. Over time the memory becomes fragmented as programs are loaded and closed because they use different amounts of memory. The operating system must keep track of different program fragments. When the memory is full, the operating system uses virtual memory
Device Drivers	Translates operating system instructions into commands that the hardware will understand. Each peripheral will need a device driver and many common ones are built into the Operating System
User Management	Providing for different users to log into a computer. The operating system will retain settings for each user, such as icons, desktop backgrounds etc. Each user may have difference access rights to files and programs. A client server network may impose a fixed or roaming profile for a user, and manage login requests to the network.
File Management	Data is stored in files. An extension to the filename tells the operating system which application to load the file into. Files can also be placed in folders for ease of organising

5. Examples of Utility Software

Encryption	Encryption utilities use an algorithm to scramble plain text into cipher text. It can be decrypted and read again with a Key
Defragmentation	Defragmentation utilities reorganise files on a hard disk, putting fragments of files back together, and it collects together free space. This reduces the movement of a read/write head across the surface of the disk, which speeds up file access. Solid state drives should not be defragmented (it is unnecessary as they have no moving parts. It also reduces their lifespan)
Compression	Compression utilities reduce the size of a file so that it takes up less space, and is quicker to download/upload.Compressed files must be extracted before they can be read. Compression is lossy or lossless
Backup	Backup utilities take a copy of the data and place it elsewhere (disks, tapes, cloud, etc.). Backups can be either full (backup everything) or incremental (back up changes since the last backup).

Crucial Knowledge 1.6 : Ethical, Legal, Cultural and Environmental Concerns

1. Privacy Issues

Implications	• Implications for personal privacy have arisen due to the vast array of cameras and surveillance systems around. • The amount of data that we share and that is recorded about us is growing hugely • Free speech / freedom of expression / right to personal privacy vs. Law and Order / Public security / government's role

2. Cultural Issues

Implications	• The impact of technology in our daily lives (Technology is changing how people live their lives today. We have an ever increasing dependency on technology in the 21st Century) • The digital divide (Access to technology and the Internet is not the same across the world) • Globalisaion (As people around the world become more exposed to technology this impacts on the values and expectations of the people in each country)
Positive Effects	• In the developing world, the rapid spread of technology, fuelled by the Internet has led to positive cultural changes in developing countries. • Easier, faster communication has contributed to the rise of democracy, as well as working towards the alleviation of poverty. • Globalisation can also increase cultural awareness and promote diversity
Negative Effects	• Diffusion of technology must be carefully controlled to prevent negative cultural consequences. • Developing countries risk losing their cultural identities and assimilating themselves into an increasingly westernised world. • Challenges of inequality from the uneven distribution of technology within a country also still remain • Traditionally, most computer applications are designed by developers in North America. These designers unintentionally apply their cultural values and systems of thought whilst developing computer applications

6. Open Source vs Proprietary Source

Open Source	Users can modify and distribute the software. Can be installed on any number of computers. Support provided by the community. May not be fully tested. Users have access to the source code
Proprietary Source	Users cannot modify the software. Protected by CD&P Act. Usually paid for and licensed per user or per computer. Supported by developers. Users do not have access to the source code. Tested by developers prior to release. Although they may run beta programmes.

3. Environmental Impact

Fossil Fuels	Fossil fuels are consumed in the manufacturing of computer devices
Energy	2% of global energy consumption is used by data centres
Disposal	Old computing equipment is often shipped to countries with lower standards for disposal. People trawl through waste looking for metals to be recycled and sold, exposing themselves to danger.

4. Impacts of Digital Technology on Wider Society

Customers	Customers can do more from home with less travelling involved. They can do things 24/7. They can access their data on many devices. Computers can make instant decisions without human involvement. Potentially open to hacking. Less personal
Staff	Job losses as things become more automated. New types of jobs created that didn't previously exist. Up-skilling required
Companies	Less overheads (salary, rent, utility bills) if fewer staff and buildings required. More ways to target potential customers. Increased importance of data protection and security
Local Communities	Local shops may suffer is town centres are more empty. Elderly and vulnerable customers may have nowhere local to go as local services are scaled back

5. Legislation

Data Protection Act (2018) [implementing GDPR]	• Data must be processed lawfully, fairly and in a transparent manner. • Data must only be collected for specified, explicit and legitimate. • Data must be adequate, relevant and limited to what is necessary. • Data you collect must be accurate and kept up to date. • Data you hold must be kept for no longer than is necessary. • Data you hold must be processed in a manner that ensures appropriate security of the personal data. • Data controllers must be able to prove that their data protection measures are sufficient
Computer Misuse Act (1990)	It is illegal to make any unauthorised access to data… …with the intent to commit further offences …with the intent to modify data, e.g. viruses
Copyright Designs and Patents Act (1998)	It is illegal to copy, modify or distribute software, music, videos or other intellectual property without permission from the author

Crucial Knowledge 2.1.1 : Algorithms

1. Computational Thinking

Abstraction	The process of removing unnecessary details and including only the relevant details.It is a method of computational thinking that focusses on what is important in problem solving
Decomposition	The process of breaking a complex problem down into smaller more manageable parts. Dealing with many different stages of a problem at once is much more difficult than breaking a problem down into a number of smaller problems and solving each, one at time.
Advantages of Program Decomposition	• Makes problems easier to solve. Different people can work on different parts of a problem at the same time… • …reducing development time. • Program components developed in one program can easily be used in other programs
Algorithmic Thinking	A way of getting to a solution by identifying the individual steps needed. By creating a set of rules, an algorithm that is followed precisely, leads to an answer. Algorithmic thinking allows solutions to be automated.

2. Input Processes and Output

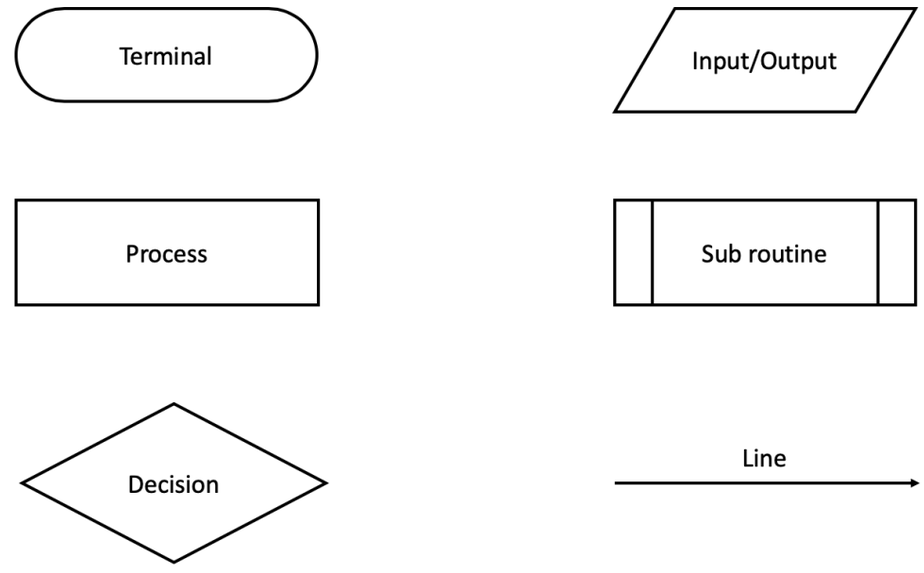
Inputs	• Anything which needs to be supplied to the program so it can meet its goals. • Often input by the user. • Consider an appropriate variable name and data type for the input.
Processes	• Consider what calculations need to be performed while the program is running. • Does data need to change formats or data types
Outputs	• Consider what your program need to output. • Consider what form this output need to take. • Consider an appropriate variable name and data type for any output

3. Structure Diagrams

- Structure diagrams illustrate problem decomposition.
- They can be used for developers to understand a problem to code and to share with users during systems analysis.
- They are produced using a method known as step-wise refinement.
- Break problem down using decomposition into ever smaller components.
- Some areas of the program will needed breaking down more than others.
- The lowest level nodes should achieve a single task.
- These can then be coded as a single module or sub-program.

3. Flowcharts, Pseudocode and OCR Reference Language

Flowchart	A method of representing the sequences of steps in an algorithm in the form of a diagram. Sometimes called a Flow diagram
Structure Diagram	A diagram showing a top-down breakdown of a complex problem
Pseudocode	A text based alternative of representing the sequences of steps in an algorithm. Pseudo-code can be thought of as a simplified form of programming code.
OCR Reference Language	You must be able to read this but you can always use Python in your exams—but be precise



4. Types of Errors

Syntax Error	Syntax errors are errors which break the grammatical rules of the programming language. They stop it from being run/translated
Logic Errors	Logic errors are errors which produce unexpected output. On their own they won't stop the program running

5. Trace Tables

- A vital skill for understanding program flow and testing the accuracy of an algorithm for logic is called "Tracing Execution".
- Examine a printed extract of program code and running thorough the program.
- Take each line at a time and write out in a trace table the current state of each variable. Noting down any output the program produces.
- Each variable present in the program should have its own column in the trace table.
- A new row should be added under any column if the state of a variable changes.
- Trace tables are an excellent way to track down logic errors in a problem.

Crucial Knowledge 2.1.2 : Searching and Sorting Algorithms

1. Binary Search

The Algorithm	• Calculate a mid-point in the data set.
	• Check if that is the item to be found.
	• If not…
	  • If the item to be found is lower than the mid-point, repeat on the left half of the data set.
	  • If the item to be found is greater than the mid-point, repeat on the right half of the data set.
	• Repeat until the item is found or there are no items left to check.

Requirements / Efficiency	• Requires the data set to be in order of a key field. • Can be done with letters as well as numbers–use alphabetical order • More efficient than a linear search on average

2. Linear Search

The Algorithm	• Starting from the beginning of a data set, each item is checked in turn to see if it is the one being searched for
Requirements / Efficiency	• Doesn't require the data set to be in order. • Will work on any type of storage device. • Can be efficient for smaller data sets. • Is very inefficient for large data sets

3. Bubble Sort

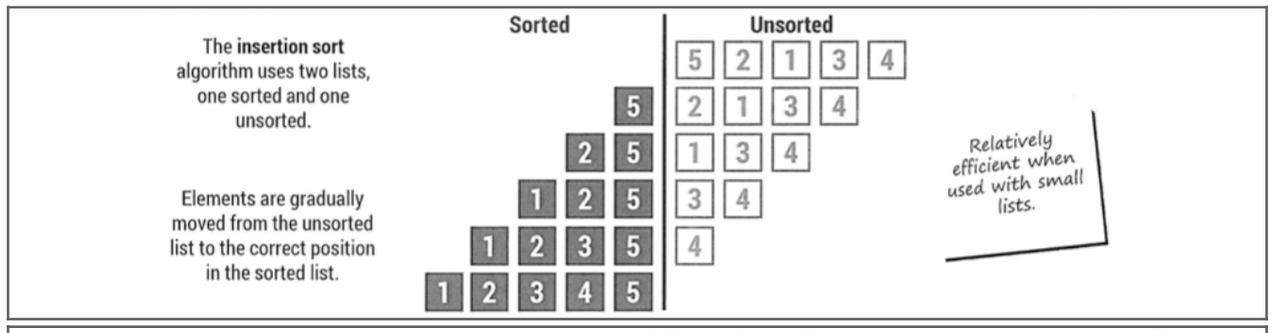
The Algorithm	• Sorts an unordered list of items. • It compares each item with the next one and swaps them if they are out of order. • The algorithm finishes when no more swaps need to be made. • In effect it "bubbles" up the largest (or smallest) item to the end of the list in successive passes.
Efficiency	• This is the most inefficient of the sorting algorithms but is very easy to implement. • This makes it a popular choice for very small data sets

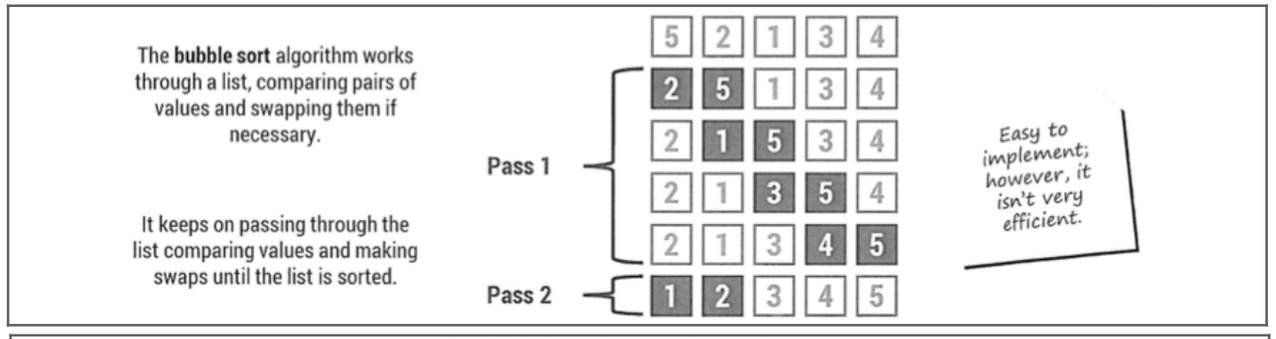
4. Insertion Sort

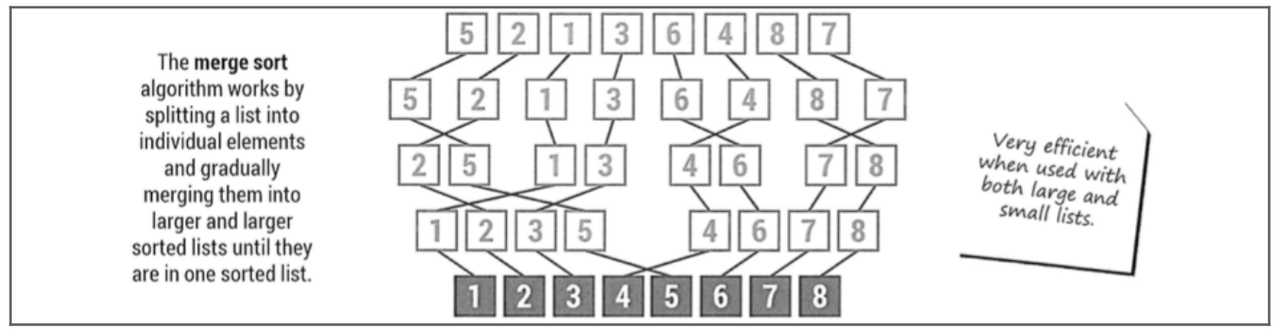
The Algorithm	• The insertion sort inserts each item into its correct position in a data set one at a time.
Efficiency	• It is a useful algorithm for small data sets. • It is particularly useful for inserting items into an already sorted list. • It is usually replaced by more efficient sorting algorithms for large data sets.

5. Merge Sort

The Algorithm	• A very efficient method of performing a sort. • Uses a divide and conquer method. • Creates two or more identical sub-problems from the largest problem, solving them individually. • Combines their solutions to solve the bigger program. • Data set is repeatedly split in half until each item is in its own list. • Adjacent lists are then merged back together.
Efficiency	• Works very well for large data sets.







6. For the exam

✓ Understand the main steps of each algorithm
✓ Understand any pre-requisites of an algorithm
✓ Apply the algorithm to a data set
✓ Identify an algorithm if given the code for it
✓ Show all your steps in detail
x To remember the code for these algorithms

Crucial Knowledge 2.2 : Programming Fundamentals 1

1. Key Terms

Variable	A value stored in memory that can change while the program is running
Constant	A value that does not change while the program is running, and is assigned when the program is designed
Operator	A character that represents an action, e.g. "+" is a mathematical Operator
Assignment	Giving a variable or constant a value
Casting	Converting a variable from one data type to another
Input	A value that is entered into the program after the program has started running
Output	A value that produced by the program and either saved or displayed to the user

2. Correct Use of Data Types

Integer	A positive or negative whole number used when arithmetic will be required
Real / Float	A positive or negative decimal number
Character	A single alphanumeric
String	Multiple characters joined together [n.b. use this for credit card numbers]
Others	Some languages have others, e.g. date, picture…

3. The Three Basic Programming Constructs

Sequence	Executing one instruction after another
Selection	Program branching depending on a condition
Interation	sometimes called looping, is repeating sections of code. Condition controlled or count controlled

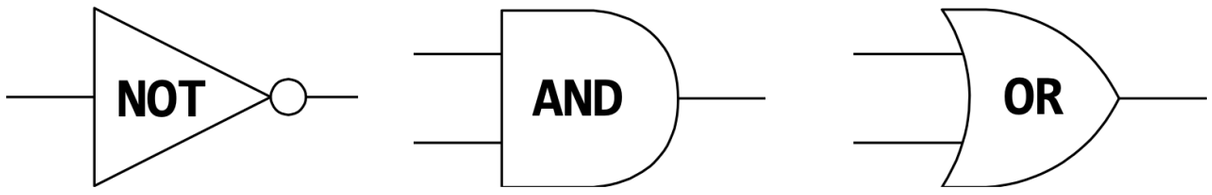
4. Common Arithmetic Operators

+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
MOD	Modulus

5. Common Comparison Operators

==	Is equal to
!=	Is not equal to
<	Is lesser than
>	Is greater than
<=	Is lesser than or equal to
>=	Is greater than or equal to

5. The Common Boolean Operators



6. Basic String Manipulation (general)

string.length	Obtains the length of the string in characters
string.upper	Converts the string to uppercase
string.lower	Converts the string to lowercase
string.left(n)	Gets the left-most n characters of the string
string.right(n)	Gets the right-most n characters of the string
string.substring(a,b)	Gets b characters of the string starting at position a
ASC(char)	Returns the numerical ASCII value of char

Note : this is NOT the way things are done in any particular programming language. In particular Python does things differently

7. Basic File Handling Operations (OCR Reference Language)

myFile=open("…")	Open a file
myFile.close()	Close a file
myFile.readLine()	Read a line from a file
myFile.writeLine()	Write a line to a file
myFile=("…")	Create a new file
string.substring(a,b)	Gets b characters of the string starting at position a
A Workflow	myFile = open ("sample.txt") while NOT myFile.endOfFile() print (myFile.readLine()) endwhile myFile.write("Hello") myFile.close()

Note : this is NOT the way things are done in any particular programming language. In particular Python does things differently

Crucial Knowledge 2.2 : Programming Fundamentals 2

1. Storing Data in Records

In Text Files	• Stored on the secondary storage (hard disk/SSD/flash). • Used to store data when the application is closed. • Useful for small volumes of data. E.g. configuration files. • Each entry is stored on a new line or separated with an identifier such as a comma or tab. • Can require a linear search to find/read data which is slow (if there is no order to the data or record structure). • Structured text files E.g. CSV, XML & JSON are popular for storing and exchanging data between applications
In Arrays and Lists	• Stored in RAM. • Used to store data when a program is running. • Useful for small volumes of data an algorithm is using. • Can be single or multi-dimensional allowing for tables of data to be stored. • Uses indexes to refer to data items. • Efficient algorithms or linear searches can be used to find data
In Databases	• Often stored on remote servers. • Often used to store data shared by many users, e.g. ticket booking system. • Data is stored in records and fields. • Uses advanced data structures to store data efficiently. • Uses very efficient algorithms to search and sort data executed on the servers. • More secure than text files. • The order of the fields in the database in independent of the code
Record Structure	• A collection of related fields. • A field is a variable. • Each field in a record can have a different data type. • Note the dot syntax when using records: record<dot>Field e.g. car1.Make

2. SQL

SELECT	which fields to be returned. * can be used to indicate all fields
FROM	which table. Databases can have more than one table, each with their own unique name
WHERE	records meet a condition. LIKE and % can be used as a wildcard
Example	SELECT name, age, iq FROM person WHERE name LIKE 'FIS%'

3. Arrays

Definition	An array is a series of memory locations – or 'boxes' – each of which holds a single item of data, but with each box sharing the same name. All data in an array must be of the same data type
Use	• Indexes usually start at 0 for the first data item (known zero indexed). • Arrays may be single or multiple dimensions. • Visualise dimensions as a column (single dimension) or table (two dimension) • In Memory two dimensional arrays are still stored in a linear fashion

4. Sub programs

Why Use them	• Larger programs are developed as a set of sub-programs called subroutines. • Structuring code into sub-programs makes the code easier to read and debug. • Each sub-program can easily be tested. • Sub-programs can be saved into libraries and reused in other programs
Functions	Functions return values and create reusable program components.
Procedures	Procedures create a modular structure to a program making it easier to read. They do not return values

5. Random Numbers

Deterministic	Programs that run on computer systems are deterministic – with exactly the same inputs they should produce exactly the same outputs.
Real World	Randomness is easy to produce in the real world – spinning a wheel, rolling a dice and so on are millennia-old techniques but producing the same randomness in a computer program is actually rather tricky
Computer	• Computers do not produce random numbers at all • They use complex mathematical techniques to produce a series of numbers that may appear random but are really only an approximation to randomness (called pseudo-random numbers) • We refer to them as random numbers anyway
OCR Reference Language	myVariable = random (1,6) will produce a random number between 1 and 6

Crucial Knowledge 2.3 : Producing Robust Programs

1. Input Validation

Validation	Does not ensure that the data entered is correct, just that it is possible and sensible
Type Check	The input is in the correct data type. E.g. Integer, Real, String
Range Check	The input is within a correct range. E.g. Between 1 and 2
Presence Check	Some data has been entered. E.g. Reject blank inputs
Format Check	The input is in the correct format. E.g. dd/mm/yyyy
Length Check	The input has the correct number of characters. E.g. 8 or more chars
Why use input validation?	• The program is more robust • The program is more user friendly • To prevent further errors occurring later in the algorithm

2. Anticipating Misuse

Division by Zero	In mathematics, there is no number which when multiplied by zero returns a non-zero number. Therefore the arithmetic logic unit cannot compute a division by zero.
Communication Error	Online systems require connections to host servers. If this connection is dropped, unable to be established or the server is overloaded, it could potentially cause a program to crash or hang when loading/saving data.
Peripheral Error	Any peripheral may be in an error mode (e.g. paper jam)
Disk Error	Programs that read and write to files must handle exceptions, including: • The file/folder not being found. • The disk being out of space. • The data in the file being corrupt. • The end of the file being reached
Authentication	• Username and password to access systems. • Password recovery by e-mailing to an authenticated e-mail address. • Encryption of data files. • Check for human and not bot attempting access (e.g. reCAPTCHA)

6. Refining Algorithms

What do we mean by refining?	• Code should anticipate all inputs and it should deal with 'bad' data, or missing data, and not crash. • It should ensure prompts to the user are helpful and that the input can only be of the correct type
How to refine	Many languages have exception handling commands

3. Maintainability

Comments	These explain the purpose of the program, or a section of code. They may also explain any unusual approaches or temporary 'fixes'
White Space	Make each section of the code stand out. Use spaces so code is not cramped up and hard to read
Indentation	Mandatory in Python but use indentation to show the flow of the program
Variable Names	Use sensible variable names that have some meaning as to what they are being used for
Sub Programs	Use Procedures and functions to structure the code and eliminate duplicating portions of it
Constants	Declare constants at the top of the program

4. Testing

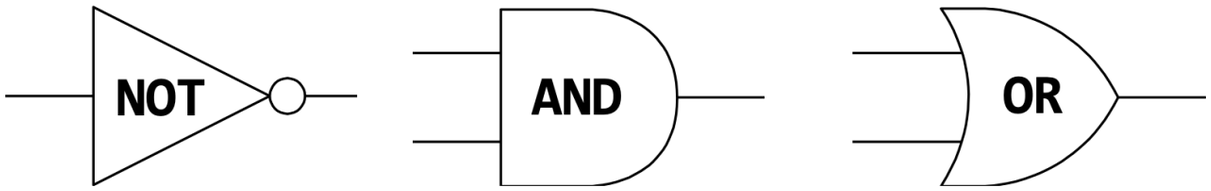
Reasons for Testing	• To ensure there are no errors (bugs) in the code. • To check that the program has an acceptable performance and usability. • To ensure that unauthorised access is prevented. • To check the program meets the requirements
Iterative Testing	• Each new module is tested as it is written. • Program branches are checked for functionality. • Checking new modules do not introduce new errors I not existing code. • Tests to ensure the program handles erroneous data and exceptional situations.
Final / Terminal Testing	• Testing that all modules work together (integration testing) • Testing the program produces the require results with normal, boundary, invalid and erroneous data. • Checking the program meetings the requirements with real data.

5. Suitable Test Data

Normal Inputs	Data which should be accepted by a program without causing errors
Boundary Inputs	Data of correct type on the edge of accepted validation boundaries
Invalid Inputs	Data of the correct type but outside accepted validation checks
Erroneous Inputs	Data of the incorrect type which should be rejected by a computer system. This includes no input being given when one is expected

Crucial Knowledge 2.4 & 2.5 : Boolean Logic, Programming Languages and IDEs

1. Logic Gate Symbols



~A A ^ B A & B A v B A | B

2. Truth Tables

A	NOT A
0	1
1	0

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

3. Levels of Programming Languages

Machine Code 1st Generation	• Binary representation of instructions in a format that the CPU can decode and execute. • Have an operation code (opcode) instruction and address or data to use (operand).
Low-Level Languages 2nd Generation	• Written in Assembly language. • Translated by an assembler into machine code. • Used for embedded systems and device drivers where instructing the hardware directly is necessary. • One instruction translated into one machine code instruction. • The code works on one type of processor only. • The programmer works with memory directly. • Code is harder to write and understand. • Memory efficient. • Code is fast to execute.
High-Level Languages 3rd Generation	• Source code is written in languages as Python, C++. • Translated by a compiler or interpreter into machine code. • Makes the writing of computer programs easier by using commands that are like English. • One source code instruction translates to many machine code instructions. • Code will run on different types of processors. • The programmer has lots of data structures to use. • Code is quicker and easier to understand and write. • Less memory efficient. • Code can be slower to execute if it is not optimised.

4. Translators

Assembler	Assembles' assembly language into machine code. Translates the whole code before execution
Compiler	Translates source code from high-level languages into object code and then into machine code ready to be processed by the CPU. The whole program is translated into machine code before it is run.
Compiler Advantages	• No need for translation software at run-time, and no need to share original source code • Speed of execution is faster because code is usually optimised.
Complier Disadvantages	• You cannot compile the program if there are syntax errors anywhere in it which can make it tricky to debug. • If you change anything you need to recompile the code
Interpreter	Translates source code from high level languages into machine code ready to be processed by the CPU. The program is translated line by line as the program is running.
Interpreter Advantages	• Easy to write source code because the program will always run, stopping when it finds a syntax error. • Code does not need to be recompiled when code is changed, and it is easy to try out commands when the program has paused after finding an error.
Interpreter Disadvantages	• Translation software is needed at run-time, so you beed to share the original source code. • Speed of execution is slower because the code is not optimised

5. Integrated Development Environments

Debugging Tools	• Breakpoints – stopping at a line of code during execution. • Stepping through lines of code one at a time. • Tracing through a program to output the values of variables.
Run Time Environment	• Output window. • Simulating different devices the program can run on.
Usability Functions	• Navigation, showing/hiding sections of code. • Formatting source code often in different colours. • Text-editor functions • Illustrating keyword syntax and auto-completing command entry.
Translator	Some IDEs have an inbuilt translator to test the program and make small alterations before compiling the final program into an executable file for distribution