# Data Representation: Compression of Images and Text

| A | Key Vocab |
|---|---|
| **Compression** | Reduction in file size to lessen download times and storage requirements |
| **Lossy** | Compression which loses data (and therefore quality) |
| **Lossless** | Compression which preserves the original data |
| **Metadata** | Data about data |

| B | Representing Text |
|---|---|
| **ASCII** | A 7-bit code which represents a basic *character set* |
| **Extended ASCII** | A *character set* represented by 8 bits instead of 7, in other ways just like ASCII |
| **Unicode** | A modern standard *character set* which uses 16 bits and includes many international characters |
| **Character set** | The complete set of letters and symbols available within a given code |

| C | Representing Images |
|---|---|
| **Bitmap** | The representation of an image by converting it to pixels and each pixel to a binary number |
| **Vector** | The representation of an image by splitting it into shapes and storing each shape as a binary number |
| **Pixel** | The smallest element of an image. One dot of one colour. |
| **Resolution** | The level of detail in an image, measured in pixels (dots) per inch (dpi) |
| **Colour depth** | The number of bits used per pixel to record colour. |
| **File Size** | width (px) × height (px) × *colour depth* |

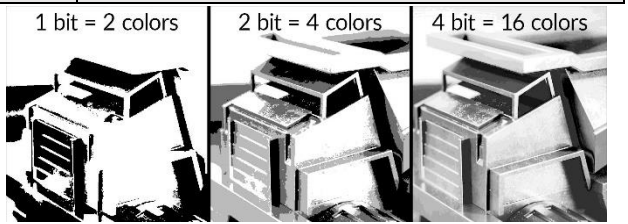
| D | File types | |
|---|---|---|
| **PDF** | document | lossless |
| **PNG** | image | lossless |
| **JPEG** | image | lossy |
| **GIF** | image | lossy |
| **BMP** | image | uncompressed |
| **MPEG** | video | lossy |
| **MP4** | video | lossy |
| **MOV** | video | lossless |
| **MP3** | audio | lossy |
| **WAV** | audio | lossy |

| E | Image metadata |
|---|---|
| Filename | |
| File format | |
| Dimensions | |
| Resolution | |
| Colour depth | |
| Time and Date | |
| Location | |
| Camera settings | |

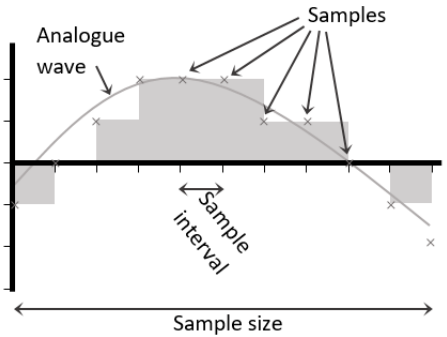
**F** **Lossy compression**

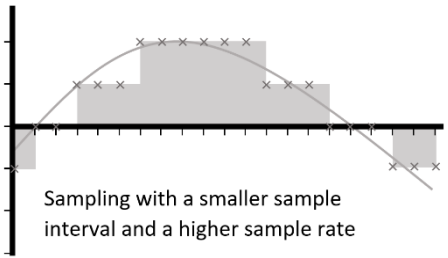
Original: 12KB | Compressed: 1.8KB | Very compressed: 0.56KB

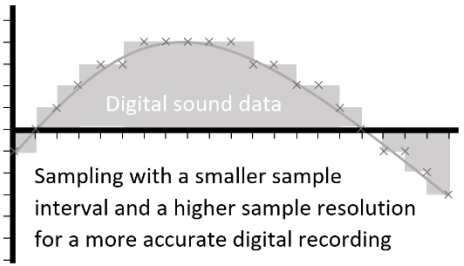
**G** **Colour depth**

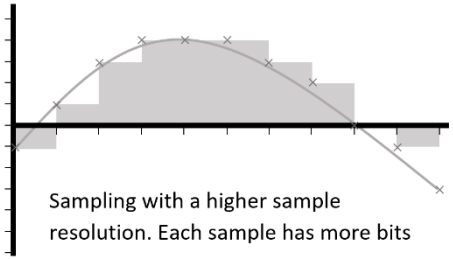


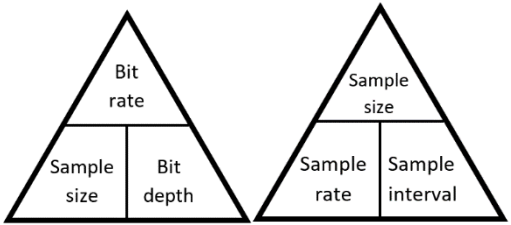
# Data Representation: Compression of Sound




| A | Representing Sound | |
|---|---|---|
| **Digital** | Having discrete values which can be stored as binary | |
| **Analogue** | Having continuously changing values | |
| **Sample** | The smallest element of a recorded sound. A value or set of values which represent a sound at a specific moment | |
| **Sample size** | The number of seconds over which a *sample* was taken | *s* |
| **Sample rate** | The number of times per second the sound is sampled. *Sample size ÷ sample interval* | *Hz* |
| **Bit rate** | The number of bits used to store a second of sampled sound. *Bit depth × sample rate* | *bps* |
| **Sample interval** | The length of time between two samples | *s* |
| **Bit depth / Sample resolution** | The number of bits used to store each sample | *b* |
| **Channel** | An audio file which is intended to be played at the same time as another | |
| **File size** | *Sample rate × bit depth × sample size* | |

# Programming: Basics

| A | Key Vocab |
|---|---|
| **Debugging** | Finding and fixing errors in code |
| **Execution** | When a command or program is run by the processor |
| **Operation** | A mathematical process which takes one or two inputs and produces one output |
| **Programming Language** | A set of instructions and syntax which can be used to make programs |
| **Script** | A small simple program, particularly run on command line interfaces |
| **Sequence** | The order in which a list of instructions is carried out |

| B | Syntax |
|---|---|
| **Comment** | A part of a program which is ignored by the computer but can be read by the programmer |
| **Indentation** | A stylistic approach for writing code. The contents of loops or selection are set a few spaces in from the previous indentation |
| **Syntax** | Rules for the structure of a programming language |

| C | Variables and Constants - Initialisation |
|---|---|
| **Assign** | Give a value to a variable or constant at the beginning of a program |
| **Data Type** | The nature of information used by a computer |
| **Declare** | Set up a *variable* by naming it and allocating memory to it |
| **Initialise** | *Declare* variables and *assign* values at the beginning of a program |

| D | Variables and Constants - Types |
|---|---|
| **Variable** | A named value which can be changed as the program is running |
| **Constant** | A label that refers to a location in memory containing a fixed value |
| **Global** | A *variable* which is used throughout the program |
| **Local** | A *variable* which is defined and used only within a sub program |

| E | Sub Programs |
|---|---|
| **Sub program** | Any section of the program which might be *called* by the main program and is self-contained |
| **Argument** | Data supplied to a *function* or *procedure* when it is *called* |
| **Breakpoint** | The part of a subprogram where it stops and returns to the main program or where the main program stops completely |
| **Call** | An instruction to run a sub program |
| **Function** | A *sub program* which can take any amount of *arguments* and *return* a value |
| **Parameter** | A *variable* which is defined within a *sub program* and which the *sub program* needs to run |
| **Procedure** | A *sub program* which can take arguments but which does not return a value |
| **Return** | To give back a value from a sub program to the main program |

# Programming: Operations

| A | Key vocab |
|---|---|
| **Operand** | A number (or string or Boolean) which is to be operated on |
| **String manipulation** | Operating on strings |

| B | Unfamiliar operations | |
|---|---|---|
| **Concatenation** | Joins two strings together | ":" + "-)" |
| **Exponentiation** | Raises one number to the power of another | 2**3 |
| **Modulus / mod** | Returns the remainder after division | 10 % 3 = 1 |
| **Quotient / floor division** | Returns the whole number part of the division | 10 // 3 = 3 |
| **Unary** | Only has one operand | -7 |

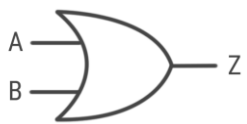
| C | Types of operator | |
|---|---|---|
| **Arithmetic operator** | An operator which turns two numbers into a single number with a mathematical process | **, /, %, //, *, +, - |
| **Assignment operator** | An operator which assigns a value to a name | =, ⇒ |
| **Boolean operator** | An operator which compares Boolean values | AND, OR, NOT |
| **Comparison operator** | An operator which compares two numbers | >, <, >=, <=, ==, != |

| D | Order of operations | |
|---|---|---|
| 1 | **Brackets** | Whatever is in the brackets is resolved first |
| 2 | **Unary** | An operation with only one *operand* |
| 3 | **Indices** | Raising to the power of a number |
| 4 | **Division** | Including *quotient* and *modulus* division |
| 5 | **Multiplication** | × or * |
| 6 | **Addition** | + |
| 7 | **Subtraction** | - |
| 8 | **Comparison** | An operation which returns a Boolean by comparing two operands |
| 9 | **Boolean** | An operation which returns a Boolean by comparing two Booleans |
| 10 | **Assignment** | An operation which assigns a value to a name |

| E | Logic vocab |
|---|---|
| **Boolean algebra** | Mathematical expression of logic circuits |
| **Logic gate** | A component which takes in one or two binary inputs and produces a single binary output |
| **Logic circuit** | A circuit made of a combination of logic gates |
| **Truth table** | A table of inputs and outputs for a logic gate system |

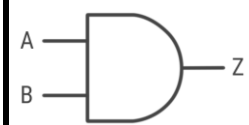
## F Logic gates

**OR gate**



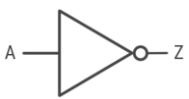

| A | B | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**AND gate**




| A | B | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**NOT gate**




| A | Z |
|---|---|
| 0 | 1 |
| 1 | 0 |

# Programming: Structures

| A | Key Vocab |
|---|---|
| **Iteration** | Repeated execution of a group of instructions |
| **Condition controlled loop** | An iteration statement which repeats until a certain requirement is met |
| **Count controlled loop** | An iteration statement which repeats for a specified number of times |
| **Search** | Find a specific item in a list of data using an algorithm |
| **Selection** | A choice of which branch to take in a program, often with IF statements |
| **Sort** | Arranging a list into an order |
| **Statement** | An instruction or clause in a program |
| **Recursive** | An algorithm which calls itself |

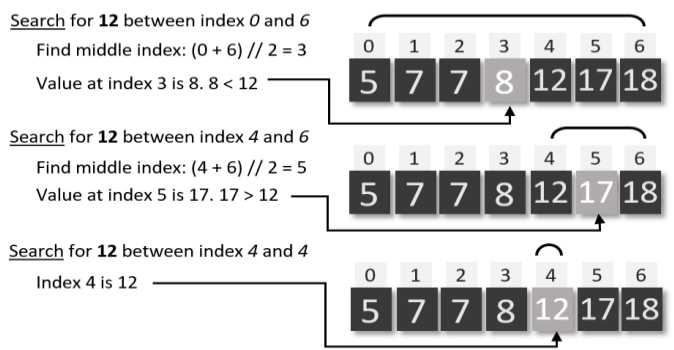
| B | Iteration structures |
|---|---|
| **DO UNTIL** | Iteration structure which has a stop condition at the end of the loop |
| **DO WHILE** | Iteration structure which has a continuation condition at the end of the loop |
| **FOR** | Iteration structure which has an index variable, a step value and a stop condition |
| **WHILE** | Iteration structure which has a start condition at the beginning of the loop |

| C | Selection structures |
|---|---|
| **IF (ELIF) ELSE** | A selection statement which branches the program under certain conditions |
| **SWITCH CASE** | A type of selection statement where there are a number of possible branches |

| D | Search |
|---|---|
| **Linear search** | A search algorithm which starts by looking at the first item in an unordered list, then moves to the second etc. |
| **Binary search** | A search algorithm which starts by looking at the middle term in an ordered list, then if the item is not found, recursively searching on the half of the list with the item in it |

| E | Sort |
|---|---|
| **Bubble sort** | A sorting algorithm which swaps adjacent items in a list if they are not in the right order, before moving onto the next pair. |
| **Insertion sort** | A sorting algorithm which goes through a list by item, removes the item and puts it into the appropriate place in a new ordered list |
| **Merge sort** | A sorting algorithm which splits a list in two, sorts each list recursively, then merges them back together |

**Binary search** F



# Programming: Data and Data types

| A | Key vocab |
|---|---|
| **Alphanumeric** | Containing letters, digits and symbols |
| **Data** | A unit of information without context, measured in bits |
| **Information** | Data, made intelligible by context |
| **Typecast** | Force a variable into a certain data type |

| B | Number Systems |
|---|---|
| **Binary** | Counting system using 1s and 0s. Computers use it because transistors can be used as switches: 1 is 'on' and 0 is 'off'. |
| **Denary** | Our normal numbering system with digits from 0 to 9 |
| **Hexadecimal** | A number system using the digits from 0 to 9 and A to E. Easy to convert to and from binary and easier to read than binary |

| C | Data types | Python |
|---|---|---|
| **Array** | An indexed list of values. The index normally starts at 0. Unlike a Python list, all values have the same data type and the maximum size is normally declared | ['o','m','g'] [6, 0, 8, 1] [0.1, 5.0] |
| **Boolean** | A data type which is either true or false | True, False |
| **Character** | A single alphanumeric symbol | 'B', '@', '8' |
| **Integer** | A data type which is a whole number | 50, -7, 2 |
| **List** | An indexed collection of data in Python | ["a", 2, True] |
| **Real / Float** | A number with a decimal point | 5.0, 3.14, 1.9 |
| **String** | A data type which is a collection of any number of characters | "hello", "", "01273" |

| D | Data measurements | |
|---|---|---|
| **Bit** | A single unit of information. A 1 or a 0. A binary digit. | *b* |
| **Nibble** | Half a byte. Four bits. | |
| **Byte** | Eight bits | *B* |
| **Kilobyte** | 1000 B | *KB* |
| **Megabyte** | 1000 KB | *MB* |
| **Gigabyte** | 1000 MB | *GB* |
| **Petabyte** | 1000 GB | *PB* |
| **Terabyte** | 1000 PB | *TB* |

| E | Binary manipulation |
|---|---|
| **Binary shift** | Adding or taking a zero at the end of a binary number |
| **Left shift** | Adding a zero to the end of a binary number, multiplying it by 2 |
| **Right shift** | Taking a zero from the end of a binary number, dividing by 2 |
| **Binary addition** | Adding binary numbers together |
| **Overflow** | A carried digit which is lost because the number is too big for the space allotted to it. ie 1111 + 0011 = 0010 (4 bit addition) |

# Programming: Translators and Debugging

| A | Translators vocab |
|---|---|
| **Assembly language** | A simple low-level language where opcodes are replaced with mnemonics and the instruction set is small (maybe 9 instructions) |
| **Compiler** | A program which turns source code into object code and saves it as an executable file |
| **Editor** | A program which allows the user to write code |
| **GUI builder** | An IDE for developing a graphical user interface |
| **High-level** (language) | A language which is easy to read and requires a lot of translating before the computer understands it |
| **Instruction set** | The full list of commands available within a language |
| **Integrated Development Environment (IDE)** | Software for writing code, which will usually incorporate an editor, debugging tools, an interpreter and compiler |
| **Interpreter** | A program which translates source code as it is read, stopping if it reaches an error |
| **Linker** | A tool which can combine different compiled codes |
| **Low-level** (language) | A language which is close to the format read by the computer |
| **Machine code / Object code** | Code written in binary |
| **One-to-many** | A language where one written instruction corresponds to a number of actions by the processor |
| **One-to-one** | A language where one written instruction corresponds to one action by the processor |
| **Pretty printing** | A feature of an editor which makes code easier to read by colouring and indenting |
| **Runtime environment** | Everything you need to run a program |
| **Translation** | Conversion of high-level language to machine code |
| **Translator** | A program which converts high-level language or assembly language to machine code |

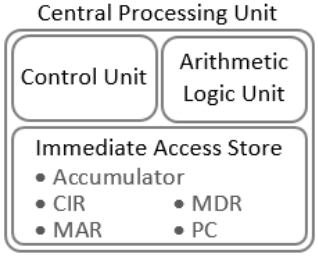
| B | Command breakdown |
|---|---|
| **Opcode** | The part of the instruction which tells the CPU what operation is to be done |
| **Operand** | The part of the instruction which is to be operated on |

| C | A single command at different levels | |
|---|---|---|
| | **Opcode** | **Operand** |
| **Machine code** | 0000 0001 | 0010 1110 |
| **Hex** | 01 | 2E |
| **Assembly** | ADD | 2E |
| **Python** | + | num |
| **Effect** | adds | the value at 0010 1110 (named num) |

| D | Debugging |
|---|---|
| **Trace table** | An offline method of tracking the values of variables through the running of a procedure |
| **Overflow error** | An error produced when a number becomes longer than the number of bits allocated to it. The extra bits are lost. |
| **Logic error** | An error with code where it compiles correctly but produces incorrect results |
| **Syntax error** | An error with the code where the computer can not recognise it as code |
| **Runtime error** | An error which occurs during operation of the program, not during compilation |

# CPU and von Neumann Architecture

| A | CPU structure | | |
|---|---|---|---|
| **Control Unit** | | CU | Communicates with the ALU, immediate access store and main memory to perform the functions of the CPU. |
| **Immediate access store** | | | A collection of registers with specific roles in the CPU |
| 1 | **Accumulator** | | Stores data to be operated on, or the result of any operation carried out by the ALU |
| 2 | **Current Instruction Register** | CIR | Stores the instruction to be used next |
| 3 | **Memory Address Register** | MAR | Stores the address to be used next (all stages) |
| 4 | **Memory Data (or Buffer) Register** | MDR MBR | Stores data which has been retrieved from or is about to be sent to RAM |
| 5 | **Program Counter** | PC | Stores the next address in the program (Fetch stage) |
| **Arithmetic and Logic Unit** | | ALU | Takes two operands from the Accumulator and an operator from the CIR and returns a single result to the Accumulator |




| D | CPU vocab |
|---|---|
| **Boot Process** | Set of instructions required to make the computer start |
| **Clock speed** | The frequency which the CPU runs at, and the number of instructions which can be processed per second (Hz) |
| **Overclock** | Run the CPU at a higher clock speed than its default |

| B | Key vocab |
|---|---|
| **Systems Architecture** | The way the components of a computer are arranged. |
| **von Neumann architecture** | System architecture where the data is stored in the same place as the instructions |
| **Fetch-Decode-Execute cycle** | The cycle followed by the von Neumann architecture |

| C | CPU hardware |
|---|---|
| **Bus** | A connector which transfers data between components. Three types are data, address and control |
| **Cache** | Fast, expensive memory which is loaded from RAM and called by the CPU |
| **Clock generator** | A circuit which produces a square wave, which is the maximum frequency a CPU can perform instructions |
| **Core** | A processing unit which can run simultaneously with others. It will have its own L1 and L2 cache, but share L3 cache and RAM |
| **Single-core** | Only one core |
| **Dual-core** | Two cores |
| **Quad-core** | Four cores |
| **Multi-core** | More than one core |
| **Register** | A section of high speed memory |

# Hardware

| A | Integral hardware | |
|---|---|---|
| **Central Processing Unit** | CPU | Main processing unit of the computer, comprising the Arithmetic and Logic Unit, the Control Unit and the immediate access store |
| **Network Interface Controller** | NIC | The part of the computer which connects to networks |
| **Hard Disk Drive** | HDD | The storage hardware which stores data permanently |
| **Heat sink** | | A device which draws heat away from any component which is likely to overheat |
| **Graphics Card** | | A piece of hardware which contains the GPU |
| **Graphical Processing Unit** | GPU | Dedicated processor for rendering images |
| **Motherboard** | | The printed circuit board on which the CPU is installed, with connectors to peripherals |

| B | Peripherals |
|---|---|
| **Peripheral** | Input, output or storage device which is not integral to the computer |
| **Input device** | A device which introduces data to the computer |
| Mouse, touchscreen, keyboard, microphone, webcam, scanner, digital camera, controller, accelerometer | |
| **Output device** | A device which displays or transmits data from the computer |
| Speaker, screen, printer, headphones, buzzer, motor | |
| **Storage device** | A device which can hold, read and write data |
| HDD, DVD drive, CD drive, USB stick, SD card reader | |
| **Dongle** | A device which attaches to a networked computer and makes it behave like a WAP |

| C | Network media vocab |
|---|---|
| **Cat 5e/ Cat 6** | Common types of UTP |
| **Coaxial cable** | Single copper wire surrounded by a metallic mesh for shielding |
| **Fibre optic cable** | Glass or plastic cable where data is transmitted as light |
| **Shielding** | Anything which goes around a data carrying wire to absorb interference |
| **Unshielded Twisted Pair (UTP)** | A type of copper wire which is often used for wired networks |
| **Wireless** | Without wires |

| D | Network hardware |
|---|---|
| **Hub** | A device which receives signals and rebroadcasts it to all connected nodes |
| **Repeater** | A device which listens for a signal and then resends it on to help reduce data collisions |
| **Router** | A device which connects networks together, and also splits data into packets, and forwards packets onward |
| **Server** | A computer which provides services for the rest of the network |
| **Switch** | A device which receives data and sends it only to the intended destination |

# Computer Science: Basics

| A | Key vocab |
|---|---|
| **Algorithm** | A set of instructions for a specific task |
| **Application** | A program which has a user interface |
| **Data** | A unit of information without context, measured in bits |
| **Device** | A tool or machine with a particular purpose |
| **Email** | A system of sending message files over the internet |
| **File** | Data, stored and named |
| **General purpose computer** | A computer which is designed to do a variety of jobs |
| **Hardware** | Physical parts of a computing |
| **Image** | Visual stored data |
| **Internet** | A huge network of millions of networks |
| **Memory** | Normally a synonym for RAM |
| **Network** | A collection of computers and other devices (nodes), connected together (by links) |
| **Program** | A series of coded instructions which can be run by a processor |
| **Random Access Memory (RAM)** | Volatile primary storage which contains the data and instructions for any program being currently run, including the OS |
| **Sampling** | Converting an analogue sound signal to a digital signal by recording the sound values at set intervals |
| **Software** | Completed computer programs in general |
| **Storage** | Where data, programs and files are kept semi-permanently |
| **World Wide Web** | The collection of web pages available over the internet |

| B | Computer systems |
|---|---|
| **Control system** | A computer which is used to control machinery |
| **Dedicated system** | A computer which is dedicated to a specific job |
| **Embedded system** | A computer which is dedicated to a specific job as part of a larger device |
| **Real time system** | A system which can guarantee response time to be short and fixed. Useful for safety-critical systems |

| C | LECE |
|---|---|
| **Cyber bullying** | Emotionally abusing someone via social media or other online methods |
| **Cyber security** | Issues surrounding protection of data and computers from the threat of hacking or malware |
| **Digital divide** | The inequality created by the fact that some people have greater access to technology than others |
| **Sharing economy** | Technology enabled renting of services or products such as Uber or AirBnB |
| **Stakeholder** | Someone with an interest |
| **Trolling** | Trying to provoke arguments or upset people online |

# Memory

| A | Secondary Storage: Types |
|---|---|
| **Flash** | A type of SSD which stores information by forcing electrons through a barrier with a large current |
| **Magnetic** | Cheap storage which requires moving parts and writable magnetic disks |
| **Optical** | Cheap storage which requires a laser and a disk |
| **Solid State Drive (SSD)** | Memory with no moving parts |

| C | Primary Storage |
|---|---|
| **Main memory** | Other ways of saying RAM |
| **Primary storage** | |
| **Virtual memory** | Part of secondary storage which is used as main memory when RAM is full |
| **Dynamic RAM** | Single transistor / capacitor RAM which needs to be refreshed every few milliseconds |
| **Static RAM** | 4/5 transistor RAM which can hold data without being refreshed (but does need power) |

| D | Key Vocab |
|---|---|
| **Read Only Memory (ROM)** | Non-volatile memory which cannot be over-written. Generally used for booting |
| **Storage device** | Any hardware which can hold, read and write data |
| **Storage medium** | The type of material or method used to store data |
| **Tertiary storage** | External high-capacity storage |
| **Volatile** | Memory which requires power |
| **Non-volatile** | Memory which persists without power |

| B | Secondary Storage: Qualities | |
|---|---|---|
| 1 | **Capacity** | Amount of data a storage device can hold |
| 2 | **Durability** | How well the device resists damage |
| 3 | **Portability** | How easily the device can be carried |
| 4 | **Reliability** | How well the data resists corruption |
| 5 | **Speed** | How quickly the data can be read from the storage device |
| 6 | **Cost** | Pounds per GB |

| E | The Cloud | |
|---|---|---|
| **Cloud** | Remotely located storage and software, accessed via the internet | |
| | **Advantages** | **Disadvantages** |
| 1 | No need to update application software | Entrusting potentially sensitive data with outsiders |
| 2 | No need to maintain the equipment, software or data | Safety and security of sensitive data is outside your control |
| 3 | No need to employ network managers or other technical staff | The service must be totally reliable |
| 4 | Service provider takes care of backups | Requires internet connection |
| 5 | Easy to share files and collaborate across platforms and locations | |

# Networks: Basics

| A | Key vocab |
|---|---|
| **Address** | The direction of where a piece of data should go |
| **Bandwidth** | The amount of data that can be transferred on a mobile network at one time |
| **Channel** | A division of a link (either wired or wireless) |
| **File sharing** | Transferring files across a network |
| **Hotspot** | A location that provides an internet connection |
| **Interoperable** | When two different systems can communicate and use shared data |
| **MAC address** | Unique ID for every device that might join a network |
| **Malware** | Malicious software |
| **Media** | Plural of medium |
| **Medium** | The means of transporting data |
| **Service Set Identifier (SSID)** | ID of the wireless access point |
| **Signal** | A wave or current which conveys data |
| **Traffic** | The amount of data travelling on a network |
| **Virtual Server** | A non-physical server |
| **Wireless Access Point (WAP)** | The point at which a wireless device connects to a network |

| C | Network types | | |
|---|---|---|---|
| **Client-Server** | Network architecture where clients connect to a server | | |
| **Peer to peer (P2P)** | Network architecture where all nodes can act as clients and servers | | |
| **MAN** | Metropolitan Area Network | **VPN** | Virtual Private Network |
| **PAN** | Personal Area Network | **WAN** | Wide Area Network |
| **SAN** | Storage Area Network | **WLAN** | Wireless LAN |

| B | Network specific vocab |
|---|---|
| **Client** | A computer or software which uses services over a network |
| **Server** | A computer which provides services for the rest of the network |
| **Link** | A connection between two nodes in a network |
| **Node** | A device in a network |
| **Local Area Network (LAN)** | A network where all nodes are on a single geographical site |
| **Protocol** | System of rules which must be followed by all parties involved in transferring data over a network |
| **Routing** | Getting data to its destination |
| **Topology** | The way a network is arranged |

| D | Topologies |
|---|---|
| **Mesh** | Nodes are all connected (directly or indirectly) without an intermediate server |
| **Full mesh** | All nodes are involved in the transmission of data without need for an intermediate server |
| **Partial mesh** | A mesh network where some nodes are not connected to each other |
| **Bus network** | Nodes are connected to a "backbone" which is also connected to servers and peripherals |
| **Ring** | Nodes are arranged in a loop, with each node connected to two others |
| **Star** | All outer nodes are connected with one link to a central switch |

# Networks: Protocols and Routing

| A | Protocols | |
|---|---|---|
| **Ethernet** | | Used to connect devices in a LAN |
| **WiFi** | | Used to connect devices wirelessly |
| **Dynamic Host Configuration Protocol** | DHCP | System for reusing IP addresses by reassigning unused ones |
| **Media Access Control** | | For addressing devices permanently, stored in the NIC |
| **File Transfer Protocol** | FTP | For sending files over the internet |
| **HyperText Transfer Protocol** | HTTP(S) | Protocol for transferring HTML files (HTTPS is with encryption) |
| **Internet Message Access Protocol** | IMAP | For email where the client can manage a remote mailbox |
| **Post Office Protocol** | POP | For email. An email is deleted from the server as the client retrieves it |
| **Simple Mail Transfer Protocol** | SMTP | Protocol for pushing email to a server (now becoming obsolete) |
| **Transmission Control Protocol** | TCP | A protocol for splitting packets and reassembling them after transmission, and for checking the data has been correctly delivered |
| **Internet Protocol** | IP | Protocol for packet switching |
| **Transmission Control Protocol / Internet Protocol** | TCP/IP | The protocol for general use of the internet |

| B | Routing |
|---|---|
| **Encapsulation** | Enclosing data inside another data structure to form a single component |
| **De-encapsulation** | Stripping external data from an encapsulated item to extract the original data |
| **Header** | Information at the beginning of a packet including IP addresses of sender and receiver, protocol, packet number and length of packet |
| **Packet** | A division of data which is to be sent over TCP/IP, including a header and trailer. Created by software |
| **Payload** | Data in a packet which is what is meant to be sent |
| **Trailer** | Information at the end of a packet including error correction and end of packet marker |
| **Layering** | A system of rules, organised into an order in which they are applied |
| **Circuit switching** | Method of routing which involves opening a connection between two nodes and sending data in a stream before closing the connection |
| **Packet switching** | Method of routing which involves data being divided up into packets and sent in multiple pathways to the destination |

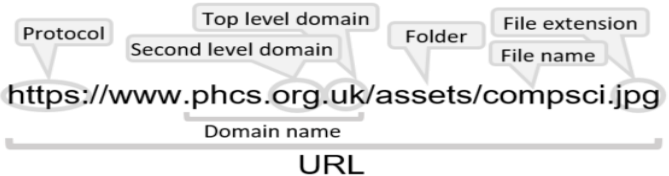
# Networks: Internet and Ethernet

| A | Key vocab | |
|---|---|---|
| **Hypertext Markup Language** | HTML | Language which websites are written in, and which a browser interprets |
| **Cascading Style Sheets** | CSS | File which adds additional styling to HTML files |
| **eXtensible Markup Language** | XML | Text-based data file for use with HTML |
| **Uniform Resource Locator** | URL | A memorable name for a domain |
| **Internet service provider** | ISP | Company which provides access to the internet |
| **Host** | A computer which stores a resource | |
| **Service** | Software which is available to use via a network | |
| **Dynamic IP address** | Temporary IP address assigned by DHCP server on connection to a network | |
| **Static IP address** | Permanent IP address assigned to a computer by the ISP | |
| **Virtual machine** | A machine (or representation of one) used through the cloud | |
| **Virtual network** | A network including virtual machines | |

| A | Domain naming |
|---|---|
| **Domain** | A group of computers on a network which are administered together |
| **Domain Name System (or Server)** | A server which contains a list of IP addresses and their associated URL |
| **Top level** | The last suffix in a URL |

| C | Ethernet |
|---|---|
| **Frame** | Data unit to be sent over Ethernet, including source and destination MAC address and error checking. Sent to all devices connected on a segment. Created by hardware |
| **Segment** | Section of an Ethernet network on a shared medium |

| A | | TCP/IP |
|---|---|---|
| **1** | **Application layer** | Makes data readable to the senders and recipients by using protocols like HTTP, FTP, SMTP etc |
| **2** | **Transport layer** | Breaks down data into packets and applies appropriate headers and trailers according to TCP |
| **3** | **Internet / network layer** | Adds sender's and recipient's IP addresses according to Internet Protocol |
| **4** | **Data link / physical layer** | Breaks data into frames according to Ethernet protocol for passing between nodes of a network and between different networks |



# Networks: Security

| A | Security policies |
|---|---|
| **Acceptable use** | Policy about what a user might reasonably use IT equipment for |
| **Email** | Policy about what can be sent over email |
| **Incident response plan** | Policy about what to do if there is a security failure |
| **Internet** | Policy about what data is allowed in and out |
| **Password** | Policy about how often passwords should change and what complexity they must be |
| **Remote access** | Policy about how to access the network from off-site |
| **Web** | Policy about what sites can be visited |
| **Wireless** | Policy about how access points are managed |

| C | Malware |
|---|---|
| **Adware** | Software which displays advertising |
| **Key logger** | Spyware which stores every keystroke in a file |
| **Ransomware** | Malware which disrupts the use of a system until a ransom has been paid |
| **Rootkit** | Modifies operating system to avoid detection |
| **Scareware** | Creates alarm and causes the user to follow a malicious link in their panic |
| **Spyware** | Gathers and reports data from the host |
| **Trojan** | Poses as legitimate software and must be installed by the user. Does not self-replicate |
| **Virus** | Hidden in an executable and self-replicates |
| **Worm** | Malware which self-replicates but does not require an executable file |

| B | Preventative Measures |
|---|---|
| **Authentication** | A process for checking the identity of the user |
| **Encryption** | The process of making data unintelligible except to the intended recipient |
| **Key** | The method of decrypting an encrypted message |
| **Public / private key** | An asymmetric encryption technique where the encryption key is public and different to the decryption key |
| **Firewall** | Software and/or hardware which controls traffic between nodes |
| **Network forensics** | Investigation to find the cause of cyber crime |
| **Packet-filter firewall** | Firewall which inspects each packet and drops non-qualifying packets |
| **Penetration testing** | Testing a system by mimicking different forms of attack |
| **Update** | The latest version of a software, including fixes of vulnerabilities |
| **User access level** | The amount of the network that a user has access to |
| **Wifi Protected Access (WPA)** | Encryption of wireless signals |

| B | LECE |
|---|---|
| **Lawful interception** | Checking data as it is transferred between networks by a legitimate entity, typically for purposes of cyber security |

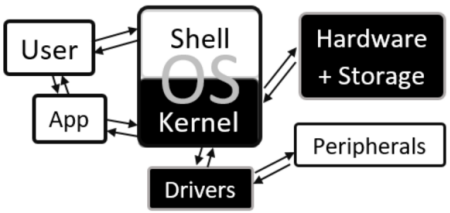
# Networks: Attacks and Data Collisions

| A | Types of attack |
|---|---|
| **Active** | A network attack where the hacker attempts to change data or introduce malware |
| **Backdoor** | An access channel which is opened to outsiders without the users' knowledge |
| **Brute force** | Hacking technique involving trying every possible combination of a password |
| **Data interception** | Picking up data as it is being sent across networks |
| **Denial of Service (DoS)** | An attack which aims to stop a server working by using up all its bandwidth |
| **Hacking** | Accessing someone else's data without consent |
| **Insider** | A network attack where someone within an organisation exploits their network access with malicious intent |
| **Packet sniffing** | A form of data interception where packets are analysed as they are being sent |
| **Passive** | A network attack where the hacker gains access to unauthorised information |
| **Pharming** | Directing a user to a malicious website by an attack on the DNS server |
| **Phishing** | Directing a user to a malicious website from a bogus email |
| **SQL injection** | Malicious code (rather than data) which enters a system through a form field |

| A | Data Collisions |
|---|---|
| **Carrier-Sense Multiple Access with Collision Detection (CSMA/CD)** | System of preventing data collisions on Ethernet. A combination of waiting until the segment is idle and detecting if a collision has occurred |
| **Cyclic Redundancy Check (CRC)** | Error checking technique where a code is generated from the payload and sent in the trailer. The receiver generates the same code from the payload to make sure it is the same as the code in the trailer |
| **Data collision** | When packets are sent over the same segment at the same time, in opposite directions. Data can become corrupted as packets try to pass through each other |
| **Duplex (communication)** | Communication can be in either direction, so collisions are likely |
| **Half-duplex** | Communication can be in either direction, but not at the same time |
| **Simplex** | One directional communication for avoiding data collisions |

# Software: Operating Systems

| A | Roles of an operating system |
|---|---|
| **Memory management** | Allocation of RAM to all running programs using *paging* and *segmentation*. |
| **Multi-tasking** | Running several different programs at the same time by switching between them very quickly (*scheduling*). |
| **User management** | Allowing for different users to have different accounts, security and permissions |
| **Peripheral management** | Allowing for applications to use peripherals and dealing with interrupts |
| **Utility management** | Running and maintaining utilities |
| **CPU management** | Running applications, executing and cancelling processes |
| **User Interface** | The means of communication between the user and the OS |
| **File management** | Providing a file system for storage and retrieval of files |
| **Disk management** | Organisation and maintenance of the hard drive |
| **Library provision** | Making a range of libraries available. |



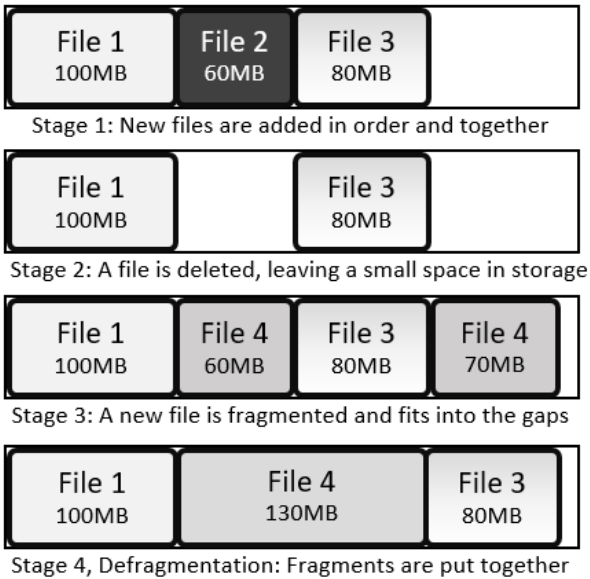

| B | Key vocab |
|---|---|
| **Paging** | Memory management technique which involves splitting RAM up into equal sized pages, and indexing them |
| **Segmentation** | Memory management technique which involves splitting RAM into blocks which fit the gaps |
| **Scheduling** | The process of arranging and controlling various processes when multi-tasking |
| **Multi-user** | When more than one user has access to the same memory, storage or CPU time |
| **Kernel** | The part of the operating system which interacts with hardware on one side and applications on the other |
| **Driver** | Software which interfaces between applications and peripherals |
| **Buffer** | A temporary area of computer memory used to store data for running processes. |
| **Interrupt** | A signal to the OS to stop it running its current program, and instead run a particular driver |
| **Graphical User Interface (GUI)** | User interface based around icons |
| **Command Line Interface (CLI)** | Text-based user interface |
| **Voice User Interface (VUI)** | User interface based around voice |
| **Library** | A suite of supporting programs which are incorporated into an OS and can be used by apps. These apps will have the same look as other apps on this OS. |
| **Static library** | A library where the routines are loaded during translation so they become part of the code. The library does not need to be present on the executing computer |
| **Dynamic Linked Library (DLL)** | A library where the routines are loaded during run time rather than translation. The library must be present on the executing computer |

| C | Prior Knowledge | | | | |
|---|---|---|---|---|---|
| Operating System | Utility | Peripheral | Real time | CPU | System Software |

# Software: Utilities

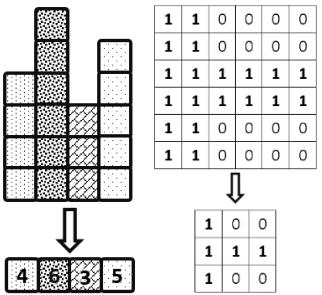
| A | Utilities |
|---|---|
| **Anti-malware (software)** | Software which prevents malicious software entering the system, identifies it when it is there and removes it |
| **Auto update** | A utility which makes sure the utilities are up to date |
| **Backup** | A copy of data and programs in case they are lost |
| **Compression software** | Software which removes redundant data to reduce file size |
| **Defragmentation** | Reorganise the files on a hard drive so they are all stored together, reducing the time the heads have to spend moving around |
| **Disk check** | Search the hard drive for bad links and record those areas as unusable |
| **Encryption software** | Software which encodes data to be stored or transferred |
| **System cleanup** | Identify and remove unused or redundant files |

| B | Fragmentation and Defragmentation |
|---|---|




| C | Backup Types |
|---|---|
| **Full backup** | All files and folders are copied when backing up |
| **Incremental Backup** | All changes since the last incremental backup are saved. To restore, start with the full backup and then restore each incremental backup successively |
| **Differential Backup** | All changes since the last full backup are saved. To restore, start with the full backup, then restore the latest differential backup |
| **Backup plan** | A scheme of when and how to back up data |

| B | Compression |
|---|---|




| D | Prior Knowledge |
|---|---|
| Operating System | |
| Utility | |
| Compression | |
| Encryption | |
| Malware | |

# Software: Basics

| A | Key vocab | |
|---|---|---|
| **Basic Input Output System (BIOS)** | | Software stored in ROM responsible for booting up a computer system |
| **Platform** | | The hardware and operating system for which software is designed |
| **System software** | | Software which is necessary for the running of other software, comprising *utilities* and the *OS* |
| **1** | **Operating System (OS)** | A collection of programs which tell hardware what to do |
| **2** | **Utility** | A single-purpose program for system maintenance |
| **3** | **Firmware** | Software that is stored permanently in a device |
| **Software repository** | | A server which contains open source software which is available for download |
| **Package management software** | | Software which downloads and updates files from a repository |
| **Batch file** | | Series of command line instructions stored in a single file |
| **Run time** | | The period during which a program is executing |
| **Instruction** | | A command that a processor can recognise and follow |
| **Source code** | | A program as it was written in high-level language |

| C | Legal and Ethical Vocab |
|---|---|
| **Copyright** | A legal right that prevents others from copying or modifying intellectual work without permission |
| **Intellectual property** | A piece of non-physical work which has been created and is owned by someone |
| **Patent** | A licence which protects intellectual property |

| B | Legislation |
|---|---|
| **Copyright, Designs and Patents Act, 1988** | Legislation which protects intellectual property by banning its unauthorised copying or redistribution |
| **Computer Misuse Act, 1990** | Legislation against hacking and disruptive behaviour on computers |
| **Data Protection Act, 1998** | Legislation which prevents storing of data about an individual which is excessive, unlawfully sourced, unsafely stored or inaccurate. |
| **Freedom of Information Act, 2000** | Legislation which gives rights for individuals to find out about data held about them |
| **Communications Act, 2003** | Legislation against malicious communication and using someone's internet without their permission |
| **Waste Electrical and Electronic Equipment Regulations, 2013** | Legislation regulating the disposal of electrical equipment |

| C | Legal and Ethical vocab |
|---|---|
| **Open-source** | Software where access to the original code is available to anyone |
| **Proprietary** | Software whose source code is kept hidden to avoid loss of profit |
| **Public Domain** | Intellectual works which are not copyrighted and are free to use |
| **Creative Commons** | Organisation which issues licences which allow the public partial or total access |
| **Licence** | A legal agreement about how a piece of software can be used or distributed |

# Software Development Cycle, Defensive Design and Computer Systems

| A | Software development stages |
|---|---|
| **Analysis** | Looking at a problem, decomposing it into sub problems, abstracting into essential points and spotting patterns, then writing success criteria for solving the problem |
| **Design** | Planning the solution to a problem, including pseudocode for algorithms and validation for data entered |
| **Development / Implementation** | Practical application of a design and its subsequent development |
| **Testing** | Making sure a program works under various conditions |
| **Documentation** | Clear evidence of and information about a product or activity |
| **Evaluation** | Judgement of the success of a product with reference to the success criteria written in the analysis |

| B | Software development processes |
|---|---|
| **Input** | Any method of introducing data to a computer |
| **Output** | Any display or transmission of data from a computer |
| **Process** | A change of state of a computer which does not involve an input or an output |
| **Execution order** | Input ⇒ Process ⇒ Output |
| **Planning order** | Output ⇒ Input ⇒ Process |

| D | Software development vocab |
|---|---|
| **Defensive design** | An approach to programming which tries to anticipate and protect against any problems through a combination of *authentication*, *sanitisation*, *validation*, *maintenance* and *testing* |
| **Maintainability** | The ability for code to be updated and repaired easily |
| **Auto-documentation** | A programming tool which helps to create summary information about a program |

| C | Defensive Design vocab |
|---|---|
| **Authentication** | A process for checking the identity of the user |
| **Maintenance** | Following procedures to keep code easy to read and error free |
| **Data validation** | As data is inputted, it is checked to make sure it is the correct data type, length, format etc |
| **Error trapping** | Planning for invalid inputs or unexpected results |
| **Input sanitisation** | Removing unwanted characters from entered data to protect against SQL injections |

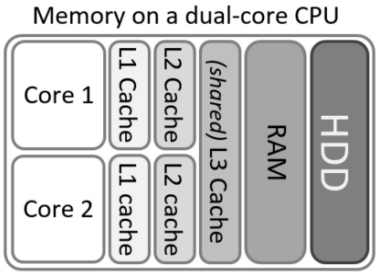
# Software: Computational Thinking, Testing and Data Checking

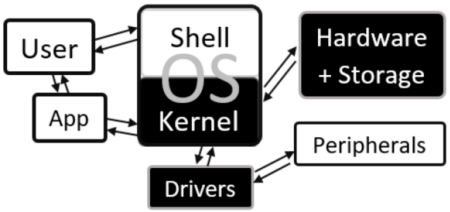
| A | Computational Thinking |
|---|---|
| **Abstraction** | A model or representation removing the inessential elements of a situation to focus on the essential elements |
| **Algorithmic thinking** | Approaching a problem by breaking it into steps which need to be followed in order |
| **Decomposition** | Breaking apart a complex problem into smaller manageable parts |
| **Computational thinking** | Approaching complex problems with a mix of abstraction, decomposition, pattern recognition and algorithmic thinking |
| **Pattern recognition** | Identifying situations with the same essential elements |
| **Program flow** | The order in which statements are executed which is affected by selection, iteration and sequencing |
| **Testing** | Making sure a program works under various conditions |

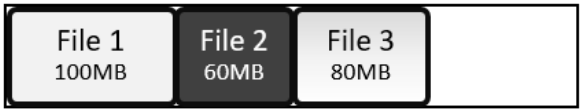
| B | Types of test |
|---|---|
| **Fault Tolerance** | Testing with illegal or out-of-range inputs |
| **Functional** | Testing with a selection of inputs which are chosen to be both normal and extreme |
| **Integration** | After a subroutine has been tested in isolation, testing to see that it works with the main program |
| **Iterative** | Testing every module before moving on |
| **Parametric** | Testing of individual subroutines |
| **Regression** | Testing after any changes have been made to see they have not made unexpected changes elsewhere |
| **User Acceptance** | Testing with users to see if they interact with the program as expected |
| **Final** | Functional testing on a high level to make sure the program works as expected |

| C | Testing vocab |
|---|---|
| **Erroneous** | Test data which should not be accepted by a program |
| **Valid** | Test data which is in range and should be handled |
| **Invalid** | Test data which is out of range and should be trapped |
| **Extreme** | Test data on the border of validity |
| **Test Plan** | Carefully chosen inputs and their expected outputs which will be used in testing |

| D | Data checking |
|---|---|
| **Check digit** | A digit which is calculated from an original number. It can be recalculated after transfer or input to make sure no errors have been introduced |
| **Check sum** | A number used to check if a packet of data has been sent correctly |
| **Parity check** | A binary check digit which is a 0 if the number of 1s is even and 1 if the number of 1s is odd (or vice versa) |

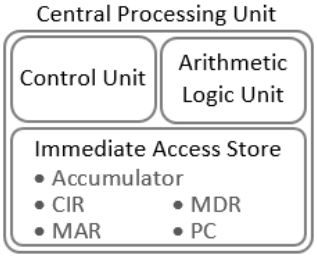


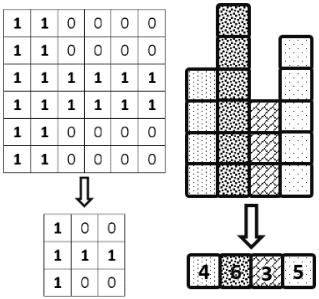




Stage 1: New files are added in order and together

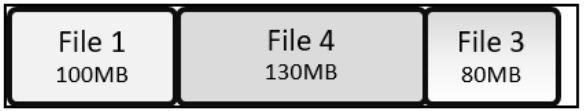

Stage 2: A file is deleted, leaving a small space in storage



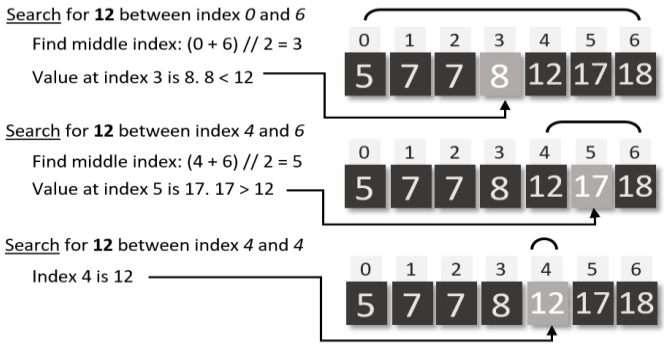



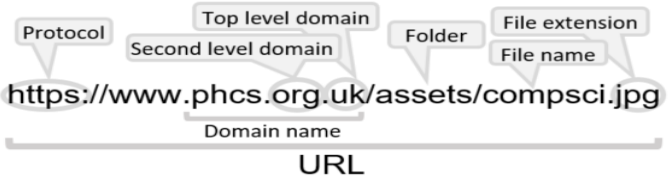


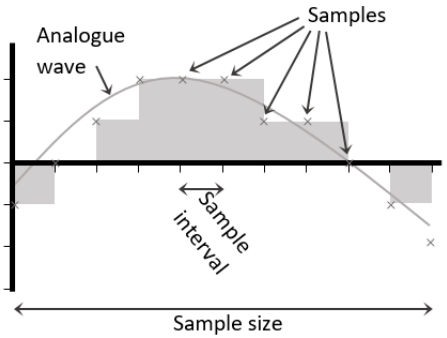
Stage 3: A new file is fragmented and fits into the gaps

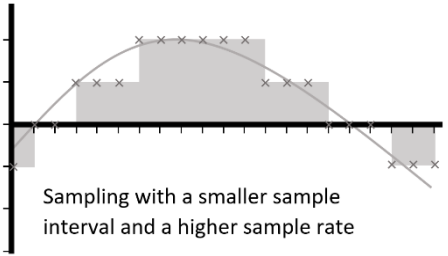



Stage 4, Defragmentation: Fragments are put together

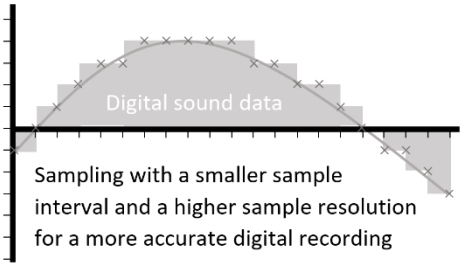


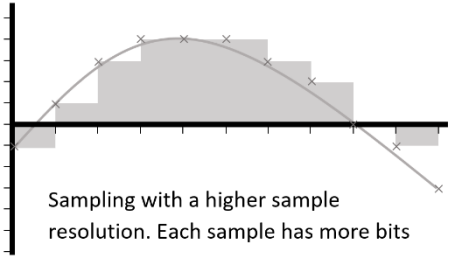


Sampling with a smaller sample interval and a higher sample rate




Sampling with a smaller sample interval and a higher sample resolution for a more accurate digital recording



Sampling with a higher sample resolution. Each sample has more bits