Curriculum Map: Computer Science Year 9

	Autumn	Spring	Summer
Content Declarative knowledge 'I Know'	1.2.1 Primary storage (Memory) 1.2.2 Secondary storage 1.2.3 Units 1.1.1 Architecture of the CPU 1.1.2 CPU performance 1.1.3 Embedded system 2.2.1 Programming fundamentals 2.2.3 Additional programming techniques (subtopics: Random number generation, 1D Array) 2.3.1 Defensive design (subtopics: Maintainability) 2.1.2 Designing, creating and refining algorithms (subtopics: Trace tables, Common errors)	1.2.4 Data storage Numbers, Characters, Images, Sound 1.2.5 Compression 2.1.2 Designing, creating and refining algorithms (subtopics: Pseudocode, flowchart, ERL introduction) 2.2.2 Data types 2.3.2 Testing	1.3.1 Networks and topologies 2.2.3 Additional programming techniques (subtopic: String handling, Sub-routines) 2.3.1 Defensive design (subtopic: Maintainability)
Skills Procedural Knowledge 'I know how to'	The need for primary storage ☐ The difference between RAM and ROM ☐ The purpose of ROM in a computer system ☐ The purpose of RAM in a computer system ☐ Virtual memory ☐ The need for secondary storage ☐ Common types of storage: o Optical o Magnetic o Solid state ☐ Suitable storage devices and storage media for a given application ☐ The advantages and disadvantages of different storage devices and storage media relating to these characteristics: o Capacity o Speed o Portability o Durability o Reliability o Cost ☐ The units of data storage: o Bit o Nibble (4 bits) o Byte (8 bits) o Kilobyte (1,000 bytes or 1 KB) o Megabyte (1,000 KB) o Gigabyte (1,000 MB) o Terabyte (1,000 GB) o Petabyte (1,000 TB) ☐ How data needs to be converted into a binary format to be processed by a computer ☐ Data capacity and calculation of data capacity requirements ☐ The purpose of the CPU:	☐ How to convert positive denary whole numbers to binary numbers (up to and including 8 bits) and vice versa ☐ How to add two binary integers together (up to and including 8 bits) and explain overflow errors which may occur ☐ How to convert positive denary whole numbers into 2-digit hexadecimal numbers and vice versa ☐ How to convert binary integers to their hexadecimal equivalents and vice versa ☐ Binary shifts ☐ The use of binary codes to represent characters ☐ The term 'character set' ☐ The relationship between the number of bits per character in a character set, and the number of characters which can be represented, e.g.: o ASCII o Unicode ☐ How an image is represented as a series of pixels, represented in binary ☐ Metadata ☐ The effect of colour depth and resolution on: o The quality of the image o The size of an image file ☐ How sound can be sampled and stored in digital form ☐ The effect of sample rate, duration, and bit depth on: o The playback quality o The size of a sound file ☐ The need for compression ☐ Types of compression: o Lossy o Lossless	☐ Types of network: o LAN (Local Area Network) o WAN (Wide Area Network) ☐ Factors that affect the performance of networks ☐ The different roles of computers in a client-server and a peer-to peer network ☐ The hardware needed to connect stand-alone computers into a Local Area Network: o Wireless access points o Routers o Switches o NIC (Network Interface Controller/Card) o Transmission media ☐ The Internet as a worldwide collection of computer networks: o DNS (Domain Name Server) o Hosting o The Cloud o Web servers and clients ☐ Star and Mesh network topologies ☐ The use of basic string manipulation ☐ How to use sub programs (functions and procedures) to produce structured code ☐ Maintainability: o Use of sub programs

	○ The fetch-execute cycle ☐ Common CPU components and their function: ○ ALU (Arithmetic Logic Unit) ○ CU (Control Unit) ○ Cache ○ Registers ☐ Von Neumann architecture: ○ MAR (Memory Address Register) ○ MDR (Memory Data Register) ○ Program Counter ○ Accumulator ☐ How common characteristics of CPUs affect their performance: ○ Clock speed ○ Cache size ○ Number of cores ☐ The purpose and characteristics of embedded systems ☐ Examples of embedded systems Use of Python programming language to do the following: ☐ The use of variables, constants, operators, inputs, outputs and assignments ☐ The use of the three basic programming constructs used to control the flow of a program: ○ Sequence ○ Selection ○ Iteration (count- and condition-controlled loops) ☐ The common arithmetic operators ☐ The common Boolean operators AND, OR and NOT Use of Python programming language to do the following: ☐ The use of one-dimensional arrays / Lists (in Python) when solving problems ☐ Random number generation Use of Python programming language to do the following: ☐ Maintainability: ○ Naming conventions ○ Indentation ○ Commenting Use of Python programming language to do the following: ☐ Identify common errors ☐ Trace tables	☐ Create, interpret, correct, complete, and refine algorithms using: ○ Pseudocode ○ Flowcharts ○ Reference language/high-level programming language ☐ The use of data types: ○ Integer ○ Real ○ Boolean ○ Character and string ○ Casting ☐ The purpose of testing ☐ Identify syntax errors	
Strategies Conditional Knowledge 'I know when to'	• How different characteristics of secondary storage devices render them suitable (or not) for a variety of storage problems / scenarios • What impact the sizes of hardware such as RAM, HD and cache have on the performance of a PC • How it is the ability of a software to use multiple cores that will determine the performance of a multi-core PC (and not just the fact that it *has* multiple cores) • In programming,	• How and why encoding schemes evolved, and which ones are appropriate for different types of character sets / languages. • When it is required for files to be compressed and what type of compression is appropriate for each file type.	• Why programming solutions are more maintainable when modular in nature. • Where to use functions and procedures effectively

	• difference between, and when to use count-controlled and when to use condition-controlled loops. • Difference between, and appropriate usage of types of while loops	• Why it is important to carefully consider if casting is the best way forward in a program as it can lead to loss of data if used incorrectly.	
Key Questions	✓ Why computers have secondary storage ✓ Differences between each type of storage device/medium ✓ Why computers have primary storage ✓ Key characteristics of RAM and ROM ✓ Why virtual memory may be needed in a system ✓ How virtual memory works ✓ What actions occur at each stage of the fetch-execute cycle ✓ The role/purpose of each CPU component and what it manages, stores, or controls during the fetch-execute cycle ✓ The purpose of each register, what it stores (data or address) ✓ Understanding of each characteristic: Clock speed, cache size, number of cores ✓ The effects of changing any of the above characteristics on system performance, either individually or in combination ✓ What embedded systems are ✓ Familiarity with a range of different embedded systems --- ✓ Practical use of fundamental programming techniques in a high-level language ✓ Understanding of each technique ✓ Understand why commenting is useful and apply this appropriately	✓ Why data must be stored in binary format ✓ Familiarity with data units and moving between each ✓ Calculate capacity of devices / for a given set of files ✓ Calculate file sizes of sound, images and text files ▪ sound file size = sample rate x duration (s) x bit depth ▪ image file size = colour depth x image height (px) x image width (px) ▪ text file size = bits per character x number of characters ✓ Conversion of any number between the bases Binary, Denary, Hexadecimal ✓ Understand the effect of a binary shift (both left or right) on a binary number ✓ How characters are represented in binary ✓ How the number of characters stored is limited by the bits available ✓ The differences between and impact of each character set ✓ Each pixel has a specific colour, represented by a specific code ✓ The effect on image size and quality when changing colour depth and resolution ✓ Metadata stores additional image information (e.g. height, width, etc.) ✓ Analogue sounds must be stored in binary ✓ Duration – how many seconds of audio the sound file contains ✓ The effect on audio file size when changing bit depth and sample rate ✓ Bit depth – number of bits available to store each sample (e.g. 16-bit) ✓ Common scenarios where compression may be needed ✓ Advantages and disadvantages of each type of compression ✓ Practical use of the data types in a high-level language within the classroom ✓ Ability to choose suitable data types for data in each scenario ✓ Understand that data types may be temporarily changed through casting, and where this may be useful	✓ The characteristics of LANs and WANs including common examples of each ✓ Understanding of different factors that can affect the performance of a network, e.g.: ▪ Number of devices connected ▪ Bandwidth ✓ The tasks performed by each piece of hardware ✓ The concept of the Internet as a network of computer networks ✓ A DNS's role in the conversion of a URL to an IP address ✓ Concept of servers providing services (e.g. Web server " Web pages, File server " file storage / retrieval) ✓ Concept of clients requesting/using services from a server ✓ The Cloud: remote service provision (e.g. storage, software, processing) ✓ Advantages and disadvantages of the Cloud ✓ Advantages and disadvantages of the Star and Mesh topologies ✓ Apply understanding of networks to a given scenario --- ✓ Practical use of the additional programming techniques in a high-level language ✓ Ability to manipulate strings, including: ▪ Concatenation ▪ Slicing ✓ Arrays as fixed length static structures ✓ The use of functions ✓ The use of procedures
Assessment topics	Oct-Nov assessment (baseline)	Mini tests Feb-March assessment (mid-year)	Mini tests May-Jun assessment (end of year)

Cross curricular links/Character Education	Logical Thinking, Mathematics.	Mathematics. Pattern matching. Science(waves)	Problem solving, Resilience. Relevance of study to software industry practise (e.g. algorithm design, testing)