Curriculum Progression (Intent)

Long Term Intent Technologies: Computer Science

- Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems.
- Understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem.
- Use 2 or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions.
- Understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be represented in binary and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal].
- Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems.
- Understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits.
- Undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users.
- Create, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability.
- Understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct, and know how to report concerns.
-

	Knowledge and Understanding	Skills
Year 11	1.4.1 Threats to computer systems and networks • Forms of attack: ○ Malware ○ Social engineering, e.g. phishing, people as the 'weak point' ○ Brute-force attacks o Denial of service attacks ○ Data interception and theft ○ The concept of SQL injection 1.4.2 Identifying and preventing vulnerabilities • Common prevention methods: ○ Malware ○ Social engineering, e.g. phishing, people as the 'weak point' ○ Brute-force attacks ○ Denial of service attacks **1.5 System software** 1.5.1 Operating systems	In Year 11, learners continue to use range of techniques in Python to produce robust and maintainable code. **2.1.3 Searching and sorting algorithms** • Standard searching algorithms: ○ Binary search ○ Linear search • Standard sorting algorithms: ○ Bubble sort ○ Merge sort ○ Insertion sort **2.3 Producing robust programs** **2.3.1 Defensive design** • Defensive design considerations: ○ Anticipating misuse ○ Authentication • Input validation • Maintainability: ○Use of sub programs ○Naming conventions ○Indentation ○Commenting

	• The purpose and functionality of operating systems: ○ User interface ○ Memory management and multitasking ○ Peripheral management and drivers ○ User management ○ File management 1.5.2 Utility software • The purpose and functionality of utility software • Utility system software ○ Encryption software ○ Defragmentation ○ Data compression **<u>1.6 Ethical, legal, cultural and environmental impacts of digital technology</u>** 1.6.1 E,L,C and E impact • Impacts of digital technology on wider society including: ○ Ethical issues ○ Legal issues ○ Cultural issues ○ Environmental issues ○ Privacy issues • Legislation relevant to Computer Science: ○ The Data Protection Act 2018 ○ Computer Misuse Act 1990 ○ Copyright Designs and Patents Act 1988 ○ Software licences (i.e. open source and proprietary)	**2.3.2 Testing** • The purpose of testing • Types of testing: ○Iterative ○Final/terminal • Identify syntax and logic errors • Selecting and using suitable test data: ○Normal ○Boundary ○Invalid/Erroneous • Refining algorithms **<u>2.5 Programming languages and Integrated Development Environments</u>** **2.5.1 Languages** • Characteristics and purpose of different levels of programming language: ○ High-level languages ○ Low-level languages • The purpose of translators • The characteristics of a compiler and an interpreter **2.5.2 The Integrated Development Environment (IDE)** • Common tools and facilities available in an Integrated Development Environment (IDE): ○Editors ○Error diagnostics ○Run-time environment ○Translators
Year 10	**<u>1.1 System Architecture</u>** **1.1.1 Architecture of the CPU** • The purpose of the CPU ○ 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 **1.1.2 CPU performance**	In Year 10, students will get to use a range of software to assist with their learning such as Flowol, draw.io, Excel, Word, IDLE. They will continue their KS3 skills of using Python by: **<u>2.1 Algorithms</u>** **2.1.1 Computational thinking** • Principles of computational thinking: ○ Abstraction ○ Decomposition ○ Algorithmic thinking **2.1.2 Computational thinking** • Identify the inputs, processes, and outputs for a problem • Structure diagrams

	• How common characteristics of CPUs affect their performance: • Clock speed • Cache size • Number of cores **1.1.3 Embedded systems** • The purpose and characteristics of embedded systems • Examples of embedded systems **<u>1.2 Memory and storage</u>** **1.2.1 Primary storage (memory)** • The need for primary storage • The difference between RAM and ROM • The purpose of RAM/ROM in a computer system • Virtual memory **1.2.2 Secondary storage** • The need for secondary storage • Common types of storage: ○ Optical ○ Magnetic ○ 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: ○ Capacity ○ Speed ○ Portability ○ Durability ○ Reliability ○ Cost **1.2.3 Units** • The units of data storage: ○ Bit ○ Nibble (4 bits) ○ Byte (8 bits) ○ Kilobyte (1,000 bytes or 1 KB) ○ Megabyte (1,000 KB) ○ Gigabyte (1,000 MB) ○ Terabyte (1,000 GB) ○ 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 **1.2.4 Data storage** **Numbers**	• Create, interpret, correct, complete, and refine algorithms using: ○ Pseudocode ○ Flowcharts ○ Reference language/high-level programming language • Identify common errors • Trace tables **<u>2.2 Programming findamentals</u>** **2.2.1 Programming fundamentals** • 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 **2.2.2 Data types** • The use of data types: ○ Integer ○ Real ○ Boolean ○ Character and string ○ Casting ○ Selection **2.2.3 Additional programming techniques** • The use of basic string manipulation • The use of basic file handling operations: ○ Open ○ Read ○ Write ○ Close • The use of The use of records to store data • The use of SQL to search for data • The use of arrays (or equivalent) when solving problems, including both one-dimensional (1D) and two-dimensional arrays (2D) • How to use sub programs (functions and procedures) to produce structured code • Random number generation **<u>2.4 Boolean logic</u>** **2.4.1 Boolean logic** • Simple logic diagrams using the operators AND, OR and NOT • Truth tables

	• 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 **Characters** • 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.: ○ ASCII ○ Unicode **Images** • How an image is represented as a series of pixels, represented in binary • Metadata • The effect of colour depth and resolution on: ○ The quality of the image ○ The size of an image file **Sound** • How sound can be sampled and stored in digital form • The effect of sample rate, duration and bit depth on: ○ The playback quality ○ The size of a sound file **1.2.5 Compression** • How sound can be sampled and stored in digital form • Types of compression: ○ Lossy ○ Lossless **<u>1.3 Computer networks, connections and protocols</u>** **1.3.1 Networks and topologies** • Types of network: ○ LAN (Local Area Network) ○ WAN (Wide Area Network) • Factors that affect the performance of networks	• Combining Boolean operators using AND, OR and NOT • Applying logical operators in truth tables to solve problems

	• The different roles of computers in a client-server and a peer-topeer network
	• The hardware needed to connect stand-alone computers into a Local Area Network:
	  ○ Wireless access points
	  ○ Routers
	  ○ Switches
	  ○ NIC (Network Interface Controller/Card)
	  ○ Transmission media
	• The Internet as a worldwide collection of computer networks:
	  ○ DNS (Domain Name Server)
	  ○ Hosting
	  ○ The Cloud
	  ○ Web servers and clients
	• Star and Mesh network topologies
	• Modes of connection:
	  ○ Wired
	  ▪ Ethernet
	  ○ Wireless
	  ▪ Wi-fi
	• Encryption
	• IP addressing and MAC addressing
	• Standards
	• Common protocols including:
	  ○ TCP/IP (Transmission Control Protocol/Internet Protocol)
	  ○ HTTP (Hyper Text Transfer Protocol)
	  ○ HTTPS (Hyper Text Transfer Protocol Secure)
	  ○ FTP (File Transfer Protocol)
	  ○ POP (Post Office Protocol)
	  ○ IMAP (Internet Message Access Protocol)
	  ○ SMTP (Simple Mail Transfer Protocol)
	• The concept of layers	

Year 9	**Binary representation** Learners will recap binary fundamentals and understand how and why computers only use binary. They will learn how to convert bits, nibbles and bytes of data into decimal (denary) numbers and back into binary sequences. Learners will also learn nibble addition and binary shifts, understanding how this is used in science and data manipulation. Finally, students will appreciate how binary can represent character sets and images through pixels. Learners will appreciate the power and speed of transistors within a computer system. **Networks and Security** Learners will look at what makes a LAN or WAN network and discuss the benefits and drawbacks	Learners will use mental arithmetic to convert between base 10 and base 2 number systems. Pattern recognition, understanding the maximum and minimum values different numbers of bits can produce. Spreadsheet software skills using formatting to generate a pixelated image. Calculations of file sizes

	of having a network of devices. They will learn about the different components as well as study the star and ring topologies. They will be able to describe the advantages and disadvantages of each type of topology and sketch them appropriately for a given scenario. Learners will also look at the various threats to a network and provide both physical and non-physical solutions to these threats; including software, hardware and personnel.	Presentation software skills to produce a presentation of their knowledge of network hardware and/or network security Accurate topology diagrams
Year 8	**Coding** (Block-based to text-based programming) Learners will review block-coding using Scratch to create three different games, utilising the three programming concepts; sequence, selection and iteration. They will learn to use variables in their programs and use various inputs to manipulate their sprites. They will then be given an opportunity to create their own game using the skills they have learnt. Moving on to text-based programming, learners will be introduced to Python. They will first use Python turtle to create various shapes on a screen, using procedures to store them. They will use the random class to generate random integers for sizes, coordinates and angles for their shapes to create patterns on the output. Learners will also be introduced to lists, being able to pick random colours for their patterns. They will continue to develop their understanding of sequence, selection and iteration within Python.	Continued good practice of health and safety in the classroom and computer-based workplace. Using Intergrated Learning Environment to design and develop and multiple choice quiz using external files for differing questions. Using the random function Using software to design csv files Using Excel to review question responses Producing Infographs of data results Evaluation techniques
	They will appreciate the Learning will be introduced to a variety of software required in ICT. They will become confident at using their features and navigating the software effectively and efficiently. System software (Operating System) Word Processing Spreadsheets Presentation Browser 365 – including Teams and Outlook They will continue to develop their understanding and appreciated for e-safety They will start using Python in a limited method to produce a multiple-choice quiz for a user as part of their assessment	Health and safety when using computers Effective software manipulation • Keyboard shortcuts • The ribbon • Fonts and formatting • Headings and footers • Formulae and functions • Highlighting Organisation of files and folders Python IDLE usage Basic Python coding Evaluation techniques