Computer Science Department - Year 10

Shirley High Curriculum Map	*The Competent Computer Scientist will develop their knowledge and skills on computer systems, networks, algorithms, programming languages and environments in preparation for their GCSE.*					
	Autumn 1 Theme/Topic/Skill:	**Autumn 2** Theme/Topic/Skill:	**Spring 1** Theme/Topic/Skill:	**Spring 2** Theme/Topic/Skill:	**Summer 1** Theme/Topic/Skill:	**Summer 2** Theme/Topic/Skill: .
	System Architecture Memory and Storage	Algorithms Programming Fundamentals	Programming Fundamentals	Systems Software Boolean Logic Programming languages and Integrated Development Environments	Producing Robust Programs	Computer Networks Connections and protocols Practical Programming skills
Why now?	The learners will build up on KS3 knowledge gained on computer concepts and further develop their understanding of system architecture and the mapping of functionality onto hardware and software. Learn the Von-Neumann Architecture, its data flow characteristics, and determine why it is the chosen CPU architecture.	The learners will strengthen their knowledge and understanding of algorithms and programming concepts learnt in KS3. They will learn trace table values of variables displayed in a table and how they can be used to assist the programmer in identifying syntax errors.	The learners will build upon the Python Programming they learnt in KS3 and how it should be applied when using basic file handling with the 3 stages (open, read-write and close). Perform search of records and stored data. Learn the main functions of the operating system used.	The learners will further their knowledge from KS3 on Logic Gates by completing and editing diagrams to given scenarios. They will build on their programming fundamentals by learning that an IDE is a software, such as Python, which provides tools and facilities used to help a programmer write a program.	The learners will continue learning Programming Fundamentals by creating programs and by showing the steps that must be taken to identify types of syntax error and test them by selecting and using different types of test data.	The learners will learn about networked computer systems, including WANs and LANs. They will develop an understanding of the purpose of networks and how they are connected.
Fundamental Concepts	o Secondary storage o Units (bit, byte, kilobyte, Gigabyte, Terabyte, etc) o Data representation o Data storage o Data compression o The main components of a computer o Structure of the CPU o Fetch-decode-execute cycle o Types of primary memory (RAM, ROM)	o Computational Thinking, Designing, Creating and refining algorithms o Searching and Sorting algorithms o Write a program by using variables, user input, data types, functions, and inbuilt functions.	o Basic file handling in Python o Additional programming techniques o Functions of well-known Operating Systems	o Main Logic Gates (NOT, AND, OR) o High Level Programming Languages o Integrated Development Environment (IDE)	o Defensive design o Stage of Testing o Types of test data	o What is a Network? o Know the three main topologies. o Practical Programming skills
Students will learn	o How data can be converted into a binary format to be processed by a computer o Data Storage o How to convert positive denary whole numbers to binary numbers (up to and including 8 bits) and vice versa. o How to add two binary integers together (up to and including 8 bits) and explain overflow errors which may occur. o How to convert binary integers to their hexadecimal equivalents and vice versa o Binary shifts o How data is stored in computers. o Characters o Images o Sound o Compression o Lossy o Lossless o Purpose of the CPU o Common CPU Components and their function o Von Neumann Architecture o CPU Performance o Embedded Systems The purpose and characteristics of embedded systems o Primary Storage o The need for primary storage. o Why do computers have primary storage? o The need for Secondary storage o The units of data storage	o The Principles of Computational Thinking o Understanding of these principles and how they are used to define and refine problems. o Designing, creating and refine algorithms. o Identify inputs, o processes, and outputs for a problem. o Interpret, correct, complete, and refine algorithms using: ● Pseudocode ● Flowcharts ● Reference language/ high-level programming language o Identify common errors o Trace tables o Search and Sorting Algorithms o The standard searching algorithms: o Binary search o Linear search ¨ Standard sorting algorithms: o Bubble sort o Merge sort o Insertion sort o Learn how to use variables, constants, operators, inputs, outputs and Assignments. o Use of the three basic programming constructs used to control the flow of a program: o Sequence o Selection o Iteration (count- and condition-controlled loops) o The common arithmetic operators o The common Boolean operators AND, OR and NOT. o The use of data types: ● Integer ● Real ● Boolean ● Character and string ● Casting	o The purpose and functionality of operating systems: o User interface Memory management and multitasking Peripheral management and drivers User management File management Utility software Purpose of the identified utility software and why it is required. **Programming Techniques** o The use of arrays (or equivalent) when solving problems, including o both one-dimensional and two-dimensional arrays o How to use subprograms (functions and procedures) to produce o structured code o Random number generation o The use of SQL to search for data o SQL Syntax o The use of arrays (or equivalent) when solving problems, including o both one-dimensional (1D) and two-dimensional arrays (2D) o How to use subprograms (functions and procedures) to produce structured code.	o Simple logic diagrams using the operators AND, OR and NOT o Truth tables o Combining Boolean operators using AND, OR and NOT o How to use basic string manipulation o Learn how to use basic file handling operations: o Open o Read o Write o Close Applying logical operators in truth tables to solve problems. **o Languages** o Characteristics and purpose of different levels of programming ● High-level languages ● Low-level languages o The purpose of translators o The characteristics of a compiler and an interpreter common tools and facilities available in an Integrated o Development Environment (IDE): ● Editors ● Error diagnostics ● Run-time environment ● Translators the Integrated Development Environment (IDE)	o Defensive design considerations: o Understand the issues a programmer should consider to o ensure that a program caters for all likely input values o Anticipating misuse o Authentication o Input validation o Maintainability: o Use of sub programs o Naming conventions o Indentation o Commenting o Testing o The purpose of testing o Types of testing: o Iterative o Final/terminal o Identify syntax and logic errors o Select and using suitable test data: o Normal o Boundary o Invalid o Erroneous o Refining algorithms	o Types of network: ● LAN (Local Area Network) ● WAN (Wide Area Network) o Factors that affect the performance of networks. o The different roles of computers in a client-server and a peer-to peer network o The hardware needed to connect stand-alone computers into a Local Area Network: o The Internet as a worldwide collection of computer networks: ● DNS (Domain Name Server) ● Hosting ● The Cloud ● Web servers and clients o Star and Mesh network topologies o Practical Programming skills will be assessed in Component 2 of the qualification, Section B. o The programming task(s) must allow the students to develop o skills within the following areas when programming: ● Design ● Write ● Test ● Refine
Language for Life (Key terms /Vocabulary)	o Solid State o Magnetic o Optical o ASCII o Metadata o Lossy o Lossless o Bit Nibble o Byte o Binary Shift	o Decomposition o Abstraction o Algorithmic Thinking o Pseudocode o Flowchart o Reference Languages/ high- and low-level language o Common errors o Trace tables o Binary Search Linear Search o Bubble Sort	o Encryption software o Defragmentation o Data compression o SQL	o Error diagnostics o Run-time environment o Translators he Integrated Development Environment (IDE)	o Editors o Error diagnostics o Run-time environment o Translators o Normal o Boundary o Invalid o Erroneous	o LAN (Local Area Network) o WAN (Wide Area Network)

		o Merge Sort o Insertion Sort o Procedures and Functions				
Extended writing Opportunities	Where possible, there will be an extended response question to enable the students to demonstrate their ability to construct and develop a sustained line of reasoning.			Learners can expand their knowledge on the 9 marks questions	Learners can expand their knowledge on the 9 marks questions	Learners can expand their knowledge on the 9 marks questions
Maths Across the Curriculum	Binary uses the decimal system Addition to binary numbers.	Logical Operators Venn Diagrams notations	Flowchart	Venn Diagram	Problem Solving	Problem Solving
Links to careers/ aspirations	CAD technician Hardware Designer Machine learning engineer	Application analyst Applications developer Data analyst	Game designer Games developer Software engineer Systems analyst	UX designer VFX artist	Game designer Games developer	Forensic computer analyst
Cultural Capital	Understand that system architecture accepts an input, acts on it, and provides an output, storing the data, how they can use their phones and tablets and gaming consoles to store music, games, etc. Be aware of the Cultural issues of ICT in the digital divide due to the changing nature of employment.	Learners understand that programming languages and instructions are to perform a given task on a computer. Learners must be aware of the importance of the Environmental and Ethical Issues.	Learners understand that computer architectures require their own system software to operate such as Windows, iOS, Linux, etc. Advanced programming experience Ethical issues – Trolling Ethical Issues - Privacy	Computers only understand discrete levels (0, 1) and decisions are made in this way. There are different languages and different programming environments taking into consideration the ethical issues.	Programs require great skill and attention to ensure they perform correctly and reliably. A diverse set of skills would enhance program robustness.	Computers and mobile devices are interconnected via networks (wired / wireless). Great programming skills can perform many tasks and solve many problems. Consideration of diverse backgrounds in communication for appropriate ethics is necessary.
Practical Application of Skills	Component 1 - Computer Systems GCSE	Component 1 - Computer Systems and component 2_Computational thinking, algorithms and programming.	Component 1 - Computer Systems and component 2_Computational thinking, algorithms and programming.	Component 1 - Computer Systems and component 2_Computational thinking, algorithms and programming.	Component 1 - Computer Systems and component 2_Computational thinking, algorithms and programming.	This will be based on applied problems in computational terms, where students may use an algorithmic approach. Practical Programming skills and their ability to design, write, test, and refine programs will be assessed.