St Gabriel's Curriculum Map - Subject: Computer Science

Curriculum Intent	Develop the mind-set of a computer scientist through equipping students with the skills to participate in a rapidly changing world. The curriculum journey incorporates challenging and engaging topics, giving students the opportunity to develop their capability, creativity and subject knowledge. It also capitalises upon, and feeds into, learning within other subject disciplines including mathematics, science and design and technology. It is these opportunities that enable students to develop and hone skills that can be applied in day-to-day life.

KS2	Time	7	8	9	KS3	10	11	KS4	KS5	Careers
Declarative Knowledge: Apply their understanding of how to strengthen, stiffen and reinforce more complex structures understand and use mechanical systems in their products for example, gears, pulleys, cams, levers and linkages Understand and use electrical systems in their products for example, series circuits incorporating switches, bulbs, buzzers and motors Apply their understanding of computing to program, monitor and control their products.	HT1	Software Skills Uses a variety of software to manipulate and present digital content: data and information. Shares their experiences of technology in school and beyond the classroom. Talks about their work and makes improvements to solutions based on feedback received.	Spreadsheet modelling Understand what a relational spreadsheet is, and understands the benefits of storing data and using formulas to calculate and extract information Performs more complex searches for information e.g. using Boolean and relational operators. Analyses and evaluates data and information, and recognises that poor quality data leads to unreliable results, and inaccurate conclusions.	On a Rotation Photoshop Undertakes creative projects that collect, analyse, and evaluate data to meet the needs of a known user group. Effectively designs and creates digital artefacts for a wider or remote audience. Considers the properties of media when importing them into digital artefacts. Documents user feedback, the improvements identified and the refinements made to the solution.	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	1.1. Computer Architecture Explore how computers work internally and understand the piece of hardware that completes all processing in the computer. 1.2. Memory storage and data Representation understanding of how data is represented in a computer, by understanding different number systems and describing how sound, images and text are translated. Students will also understand how storage devices are used to save work and their characteristics. They will understand the difference between types of primary storage and how they affect the working of a computer system	2.1. Algorithms explore algorithm design and effectively solve problems by designing a solution in either a flowchart, pseudocode, or Exam Reference Language. Students should also be able to recognise common algorithms used by a computer including searching and sorting Mock revision – exam technique	Develop their capability, creativity and knowledge in computer science, digital media and information technology. Develop and apply their analytic, problem-solving, design, and computational thinking skills Understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to report a range of concerns	Level 3/Level 6 qualifications (e.g. Pearson BTEC Level 3 National Extended Diploma in Computing, SVQ IT Professionals - SQCF Level 6) may be accepted as an alternative to A Levels/Highers. A-Level Computer Science	Software Development: App developer Web developer Games Developer Design: Web designer UX design Games Designer Engineering: Software Engineer Machine leaning engineer Software Tester Analysis : Forensic computer analyst Cyber security analyst Systems analyst Average salary: £44,154
	HT2	Computer Systems Understand the input Output system and develop knowledge on the components that make up a computer and how	Computer Systems Understands the von Neumann architecture in relation to the fetch-execute cycle,	App Development Designs a solution to a problem that depends on solutions to smaller instances of the			Y11 Mocks Paper 1 and part of Paper 2			

		they interact with each other. Knows that computers collect data from various input devices, including sensors and application software. Understands the difference between hardware and application software, and their roles within a computer system.	including how data is stored in memory. Understands the basic function and operation of location addressable memory	same problem (recursion). Understands that some problems cannot be solved computationally. Designs and writes nested modular programs that enforce reusability utilising subroutines where ever possible. Understands the difference between 'While' loop and 'For' loop, which uses a loop counter. Understands and uses two dimensional data structures. Performs operations using bit patterns e.g. conversion between binary and hexadecimal, binary subtraction etc. Understands and can explain the need for data compression, and performs simple compression methods. Has practical experience of a small (hypothetical) low level	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					
	HT3	Visual Programming Designs solutions (algorithms) that use repetition and two-way selection i.e. if, then and else. Uses diagrams to express solutions. Uses logical reasoning to predict outputs, showing an awareness of inputs. Creates programs that implement algorithms to achieve given goals. Declares and assigns variables. Uses post-tested loop e.g. 'until', and a sequence of selection statements in programs, including an if, then and else statement. Knows that computers collect data from various input devices, incl	Python Understands the difference between, and appropriately uses if and if, then and else statements. Uses a variable and relational operators within a loop to govern termination. Designs, writes and debugs modular programs using procedures. Knows that a procedure can be used to hide the detail with sub-solution (procedural abstraction).			1.3. Networks understanding of how a network is constructed and how different components communicate within the network to allow it to function. They will be exposed to the different services that are available by storing data online. Students will understand how data is communicated between different branded devices and manufacturers 1.4. Network Security Explore the risks to networks for companies and other stakeholders. They will understand how to prevent or reduce these risks from happening and the preferred methods. 1.5. System Software	2.3 Programming Robust Recap Programming and explore how to produce well designed programs which are maintainable. They will learn how to debug their programs and test to ensure all possible outcomes.			

St Gabriel's Curriculum Map - Subject: Computer Science

Procedural Knowledge: Use research and develop design criteria to inform the design of innovative, functional, appealing products that are fit for purpose, aimed at particular individuals or groups Generate, develop, model and communicate their ideas through discussion, annotated sketches, cross-sectional and exploded diagrams, prototypes, pattern pieces and computer-aided design Select from and use a wider range of tools and equipment to perform practical tasks [for example, cutting, shaping, joining and finishing], accurately Select from and use a wider range of materials and components, including construction materials, textiles and ingredients, according to their functional properties and aesthetic qualities Investigate and analyse a range of existing products Evaluate their ideas and products against their own design criteria and consider the views of others to improve their work Understand how key events and individuals in design and technology	HT4	Visual to Textual Programming Understands that algorithms are implemented on digital devices as programs. Designs simple algorithms using loops, and selection i.e. if statements. Uses logical reasoning to predict outcomes. Detects and corrects errors i.e. debugging, in algorithms. Uses arithmetic operators, if statements, and loops, within programs. Uses logical reasoning to predict the behaviour of programs. Detects and corrects simple semantic errors i.e. debugging, in programs.	Python continued Understands that iteration is the repetition of a process such as a loop. Recognises that different algorithms exist for the same problem. Represents solutions using a structured notation. Can identify similarities and differences in situations and can use these to solve problems (pattern recognition). Understands that programming bridges the gap between algorithmic solutions and computers. Has practical experience of a high-level textual language, including using standard libraries when programming. Uses a range of operators and expressions e.g. Boolean, and applies them in the context of program control. Selects the appropriate data types.	programming language. Understands and can explain Moore's Law. Understands and can explain multitasking by computers.		understanding of the purpose of an operating system and that a computer can't function without this software.	2.4. Programming Languages produce logic gates to show how a computer manages circuits and predict outputs. Students can show outputs of logic gates through truth tables. 2.5. Programming Languages recognise the difference between types of programming languages and the support that is given to programmers through an IDE. They should be able to link the process of designing, creating and how the computer reads the program to facilitate effective testing and executing of the given program.	.		

have helped shape the world	HT5	Logic Gates Understands the relationship between binary and electrical circuits, including Boolean logic. Understands how and why values are data typed in many different languages when manipulated within programs	Data Representation Understands how numbers, images, sounds and character sets use the same bit patterns. Performs simple operations using bit patterns e.g. binary addition. Understands the relationship between resolution and colour depth, including the effect on file size. Distinguishes between data used in a simple program (a variable) and the storage structure for that data			1.6. Ethics explore the wider impact of computing on people, the law, industries, growth in Computing and environmental issues. 2.2. Programming understanding of programming techniques which will allow them to create independently programs to solve a problem. Students should also explore textual programming and should be given ample time to create programs to meet the specification requirements. Mock preparation – whole paper 1 apart from 6 mark ethic question	Computer Science Exam preparation & Exam			
	HT6	Ethics and the Law Understands the ethical issues surrounding the application of information technology, and the existence of legal frameworks governing its use e.g. Data Protection Act, Computer Misuse Act, Copyright etc.	Flowcharts Evaluates the effectiveness of algorithms and models for similar problems. Recognises where information can be filtered out in generalizing problem solutions (abstraction). Uses logical reasoning to explain how an algorithm works. Represents							

			algorithms using structured language.							