

Year Group	Topic	Lesson Content (Order)	What do pupils need to know	Skills utilised/subject disciplines	Cross Curricular Links/transferrable knowledge	Assessment
Year 10	1.1 Systems Architecture	<ul style="list-style-type: none"> • Architecture of the CPU • CPU performance • Embedded systems 	<p>Architecture of the CPU:</p> <ul style="list-style-type: none"> • The purpose of the CPU: <ul style="list-style-type: none"> - The fetch-execute cycle • Common CPU components and their function: <ul style="list-style-type: none"> - ALU (Arithmetic Logic Unit) - CU (Control Unit) - Cache - Registers • Von Neumann Architecture: <ul style="list-style-type: none"> - MAR (Memory Address Register) - MDR (Memory Data Register) - PC (Program Counter) - ACC (Accumulator) <p>CPU performance:</p> <ul style="list-style-type: none"> • How common characteristics of CPUs affect their performance: <ul style="list-style-type: none"> - Clock speed - Cache size - Number of cores <p>Embedded systems:</p> <ul style="list-style-type: none"> • The purpose and characteristics of embedded systems • Examples of embedded systems 	Understanding the components that make up digital systems, and how they communicate with each other and with other systems.	<p>Engineering – embedded systems</p> <p>Creative Media Production – the performance of CPU and graphics cards for editing media</p>	<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>
	1.2 Memory and Storage	<ul style="list-style-type: none"> • Primary storage • Secondary storage • Units • Data storage • Compression 	<p>Primary storage:</p> <ul style="list-style-type: none"> • 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 	Mathematics – working with numbers by converting between number systems and adding binary/hex numbers.	Creative Media Production – data storage requirements for digital media; suitability of secondary storage	Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.

			<ul style="list-style-type: none"> • Virtual memory <p>Secondary storage:</p> <ul style="list-style-type: none"> • The need for secondary storage • Common types of storage: <ul style="list-style-type: none"> - 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: <ul style="list-style-type: none"> - Capacity - Speed - Portability - Durability - Reliability - Cost <p>Units:</p> <ul style="list-style-type: none"> • The units of data storage: <ul style="list-style-type: none"> - Bit - Nibble - Byte - Kilobyte - Megabyte - Gigabyte - Terabyte - Petabyte • How data needs to be converted into binary format to be processed by a computer • Data capacity and calculation of data capacity requirements 	<p>Understanding the components that make up digital systems, and how they communicate with each other and with other systems.</p>	<p>for a given scenario</p> <p>Mathematics – binary number system; hexadecimal number system; conversions between base 2 to base 10 (and vice versa) and between base 16 to base 10 (and vice versa).</p>	<p>End of unit tests</p>
--	--	--	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------

Data storage – numbers:

- 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

Data storage – characters:

- The use of binary code 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

Data storage – images:

- How an image is represented as a series of pixels, represented in binary
- Metadata
- The effect of colour depth and resolution on:

			<ul style="list-style-type: none"> - the quality of the image - the size of an image <p>Data storage – sound:</p> <ul style="list-style-type: none"> • How sound can be sampled and stored in digital form • The effect of sample rate, duration, and bit depth on: <ul style="list-style-type: none"> - the playback quality - the size of a sound file <p>Compression:</p> <ul style="list-style-type: none"> • The need for compression • Types of compression: <ul style="list-style-type: none"> - Lossy - Lossless 			
	<p>1.3 Computer Networks, Connections and Protocols</p>	<ul style="list-style-type: none"> • Networks and topologies • Wired and wireless networks, protocols, and layers 	<p>Networks and topologies:</p> <ul style="list-style-type: none"> • Types of networks: <ul style="list-style-type: none"> - LAN (Local Area Network) - WAN (Wide Area Network) • Factors that affect the performance of networks • The different roles of computers in a client-server and peer-to-peer network • The hardware needed to connect stand-alone computers into a Local Area Network: <ul style="list-style-type: none"> - Wireless access points - Routers - Switches - NIC (Network Interface Controllers/Cards) - Transmission media • The Internet as a worldwide collection of computer networks: 	<p>Understanding the components that make up digital systems, and how they communicate with each other and with other systems.</p>	<p>Online safety Cybersecurity</p>	<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>










			<ul style="list-style-type: none"> - DNS (Domain Name Server) - Hosting - The Cloud - Web servers and clients • Star and Mesh network topologies <p>Wired and wireless networks, protocols, and layers:</p> <ul style="list-style-type: none"> • Modes of connection: <ul style="list-style-type: none"> - Wired <ul style="list-style-type: none"> ○ Ethernet - Wireless <ul style="list-style-type: none"> ○ Wi-Fi ○ Bluetooth • Encryption • IP addressing and MAC addressing • Standards • Common protocols including: <ul style="list-style-type: none"> - TCP/IP - HTTP - HTTPS - FTP - POP - IMAP - SMTP • The concept of layers 			
	1.4 Network Security	<ul style="list-style-type: none"> • Threats to computer systems and networks • Identifying and preventing vulnerabilities 	<p>Threats to computer systems and networks:</p> <ul style="list-style-type: none"> • Forms of attack: <ul style="list-style-type: none"> - Malware - Social engineering e.g., phishing, people as the “weak point” - Brute-force attacks - Denial of service attacks - Data interception and theft 	Understanding the components that make up digital systems, and how they communicate with each other and with other systems.	Personal Development – online safety and cybersecurity.	Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes. End of unit tests

			<ul style="list-style-type: none"> - The concept of SQL injection <p>Identifying and preventing vulnerabilities:</p> <ul style="list-style-type: none"> • Common prevention methods: <ul style="list-style-type: none"> - Penetration testing - Anti-malware software - Firewalls - User access levels - Passwords - Encryption - Physical security 			
1.5 Systems Software	<ul style="list-style-type: none"> • Operating systems • Utility software 	<p>Operating systems:</p> <ul style="list-style-type: none"> • Their purpose and functionality of operating systems: <ul style="list-style-type: none"> - User interface - Memory management and multitasking - Peripheral management and drivers - User management - File management <p>Utility software:</p> <ul style="list-style-type: none"> • The purpose and functionality of utility software • Utility system software: <ul style="list-style-type: none"> - Encryption software - Defragmentation - Data compression 	Understanding of how software works with hardware. Learning about key features of operating systems and utility software.	Digital literacy skills to support the rest of the curriculum	<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>	
1.6 Ethical, Legal, Cultural and Environmental Impacts of Digital Technology	<ul style="list-style-type: none"> • Ethical, legal, cultural, and environmental impacts 	<p>Ethical, legal, cultural, and environmental impact:</p> <ul style="list-style-type: none"> • Impact of digital technology on wider society including: <ul style="list-style-type: none"> - Ethical issues - Legal issues - Cultural issues - Environmental issues 	Discussion skills. Extended writing. Understanding the impact of digital technology to the individual and to wider society.	Personal Development and Religious Studies – ethical, legal, cultural impacts of technology on our lives.	Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.	

			<ul style="list-style-type: none"> - Privacy issues • Legislation relevant to computer science: <ul style="list-style-type: none"> - The Data Protection Act 2018 - Computer Misuse Act 1990 - Copyrights Designs and Patents Act 1988 - Software licenses: <ul style="list-style-type: none"> ○ Open source ○ Proprietary 			<p>End of unit tests</p> <p>Paper 1 mock paper</p>
Year 11	2.1 Algorithms	<ul style="list-style-type: none"> • Computational thinking • Designing, creating, and refining algorithms • Searching and sorting algorithms 	<p>Computational thinking:</p> <ul style="list-style-type: none"> • Principles of computational thinking: <ul style="list-style-type: none"> - Abstraction - Decomposition - Algorithmic thinking <p>Designing, creating, and refining algorithms:</p> <ul style="list-style-type: none"> • Identify the inputs, processes, and outputs for a problem • Structure diagrams • Create, interpret, correct, and refine algorithms using: <ul style="list-style-type: none"> - Pseudocode - Flowcharts - Reference language - High-level programming language (Python) • Identify common errors • Trace tables <p>Searching and sorting algorithms</p> <ul style="list-style-type: none"> • Standard searching algorithms: <ul style="list-style-type: none"> - Binary search - Linear search 	<p>Understanding and applying the fundamental principles of computer science – including abstraction, decomposition, logic, and data representation. Analyse problems in computational terms through practical experience of solving such problems, including designing, writing and debugging programs. Thinking creatively, innovatively, analytically, logically and critically.</p>	<p>Mathematics – use of algorithmic and logical operators.</p> <p>Art – creativity when designing algorithms.</p> <p>Engineering – controlling robots with text-based, high-level programming language (python).</p>	<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>

			<ul style="list-style-type: none"> Standard sorting algorithms: <ul style="list-style-type: none"> Bubble sort Merge sort Insertion sort 	Apply mathematical skills.		
	2.2 Programming Fundamentals	<ul style="list-style-type: none"> Programming fundamentals Data types Additional programming techniques 	<p>Programming fundamentals:</p> <ul style="list-style-type: none"> 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: <ul style="list-style-type: none"> Sequence Selection Iteration (count- and condition-controlled loops) The common arithmetic operators: <p style="text-align: center;"><u>Comparison operators</u></p> <pre> == Equal to != Not equal to < Less than <= Less than or equal to > Greater than >= Greater than or equal to </pre> <p style="text-align: center;"><u>Arithmetic operators</u></p> <pre> + Addition - Subtraction * Multiplication / Division MOD Modulus DIV Quotient ^ Exponentiation (to the power) </pre> The common Boolean operators AND, OR and NOT 	<p>Understanding and applying the fundamental principles of computer science – including abstraction, decomposition, logic, and data representation. Analyse problems in computational terms through practical experience of solving such problems, including designing, writing and debugging programs. Thinking creatively, innovatively, analytically, logically and critically. Apply mathematical skills.</p>		<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>

			<p>Data types</p> <ul style="list-style-type: none"> • The use of data types: <ul style="list-style-type: none"> - Integer - Real - Boolean - Character - String - Casting <p>Additional programming techniques</p> <ul style="list-style-type: none"> • The use of basic string manipulation • The use of basic file handling operations: <ul style="list-style-type: none"> - Open - Read - Write - Close • The use of records to store data • The use of SQL to search for data • The use of arrays when solving problems, including one-dimensional (1D) and two-dimensional (2D) • How to use sub programs (functions and procedures) to produce structured code • Random number generation 			
	2.3 Producing Robust Programs	<ul style="list-style-type: none"> • Defensive design • Testing 	<p>Defensive design:</p> <ul style="list-style-type: none"> • Defensive design considerations: <ul style="list-style-type: none"> - Anticipating misuse - Authentication • Input validation • Maintainability: <ul style="list-style-type: none"> - Use of sub programs - Naming conventions - Indentation - Commenting 	Understanding and applying the fundamental principles of computer science – including abstraction, decomposition, logic, and data representation.		<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>

			<p>Testing:</p> <ul style="list-style-type: none"> • The purpose of testing • Types of testing: <ul style="list-style-type: none"> - Iterative - Final/terminal • Identify syntax and logic errors • Selecting and using suitable test data: <ul style="list-style-type: none"> - Normal - Boundary - Invalid/erroneous • Refining algorithms 	<p>Analyse problems in computational terms through practical experience of solving such problems, including designing, writing and debugging programs. Thinking creatively, innovatively, analytically, logically and critically. Apply mathematical skills.</p>									
2.4 Boolean Logic	<ul style="list-style-type: none"> • Boolean logic 	<p>Boolean logic:</p> <ul style="list-style-type: none"> • Simple logic diagrams using the operators AND, OR, and NOT <table border="1" data-bbox="817 901 1131 1077"> <thead> <tr> <th>Boolean Operators</th> <th>Logic Gate Symbol</th> </tr> </thead> <tbody> <tr> <td>AND (Conjunction)</td> <td></td> </tr> <tr> <td>OR (Disjunction)</td> <td></td> </tr> <tr> <td>NOT (Negation)</td> <td></td> </tr> </tbody> </table> <ul style="list-style-type: none"> • Truth tables • Combining Boolean operators using AND, OR, and NOT • Applying logical operators in truth tables to solve problems 	Boolean Operators	Logic Gate Symbol	AND (Conjunction)		OR (Disjunction)		NOT (Negation)		<p>Problem solving. Creativity. Patience and resilience. Logical thinking.</p>		<p>Ongoing formative assessments in lesson – cold calling; mini whiteboards; quizzes.</p> <p>End of unit tests</p>
Boolean Operators	Logic Gate Symbol												
AND (Conjunction)													
OR (Disjunction)													
NOT (Negation)													
2.5 Programming Languages and Integrated Development Environments	<ul style="list-style-type: none"> • Languages • The integrated development environment (IDE) 	<p>Languages:</p> <ul style="list-style-type: none"> • Characteristics and purpose of difference levels of programming language: <ul style="list-style-type: none"> - High-level languages - Low-level languages 	<p>Problem solving. Creativity. Patience and resilience. Logical thinking.</p>		<p>Ongoing formative assessments in lesson – cold calling; mini</p>								

- The purpose of translators
- The characteristics of a compiler and an interpreter

The integrated development environment (IDE):

- Common tools and facilities available in an integrated development environment (IDE):
 - Editors
 - Error diagnostics
 - Run-time environment
 - Translators

whiteboards;
quizzes.

End of unit tests

Paper 2 mock

GCSE
Computer
Science
follows the
OCR J277
Computer
Science
official

specification.

Latest version available from the examination board here: <https://www.ocr.org.uk/qualifications/gcse/computer-science-j277-from-2020/>

Computer Science allows students to apply for various KS5 courses, especially:

- AS Level Computer Science
- A Level Computer Science
- Cambridge Technicals IT Level 2 and 3

Which then lead to career and progression:

- University
- Employment
- Apprenticeship (Level 2 and 3 / Higher apprenticeship)