Computer Science – Year Group Overview

	Term 1	Term 2	Term 3	Term 4	Term 5	Term 6	
	1.1 Systems Architecture	1.2 Memory and Storage	1.2 Memory and Storage	1.3 Computer Networks, Connections and Protocols	1.4 Network Security 1.5 Systems Software	1.6 Ethical, legal, cultural, environmental impacts of digital technology	Careers
Year 10	**Architecture of the CPU** • The purpose of the CPU. • Common CPU components and their function. • Von Neumann architecture. **CPU performance** • How common characteristics of CPUs affect their performance. **Embedded Systems** • The purpose and characteristics of embedded systems. • Examples of embedded systems.	**Units of Storage** • The units of data storage. • How data needs to be converted into a binary format to be processed by a computer. • Data capacity and calculation of data capacity requirements. **Primary Storage (Memory)** • 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. **Secondary Storage** • The need for secondary storage. • Common types of storage. • Suitable storage devices and storage	**Data Storage (Image)** • 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 an image & the size of an image. **Data Storage (Sound)** • How sound can be sampled and stored in digital form. • The effect of sample rate, duration and bit depth on: playback quality & size of a sound file. **Compression** • The need for compression. • Types of compression.	**Networks and Topologies** • Types of 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. • The Internet as a worldwide collection of computer networks. • Star and Mesh network topologies. **Wired and Wireless Net** • Modes of connection Encryption. • IP addressing and MAC addressing Standards. **Protocols and Layers** • Common protocols: ○ TCP/IP ○ HTTP ○ HTTPS	**Threats** Forms of attack: • Malware. • Social engineering. • Brute-force attacks. • Denial of service attacks. • Data interception and theft. • The concept of SQL injection. **Vulnerabilities** Common prevention methods: • Penetration testing. • Anti-malware software. • Firewalls. • User success levels. • Passwords. • Encryption. • Physical security. **Operating Systems** The purpose and functionality of operating systems. **Utility Software** The purpose and functionality of utility software.	**Ethics** Impacts of digital technology on wider society including: • Ethical. • Legal. • Cultural. • Environmental. • Privacy. **Legislation** 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).	**Term 4: Network Manager** While studying the networks and topologies students will have the chance to discover the careers opportunities within network management. **Term 5: White Hat Hackers** While studying about hacking and the different types of threat online students will have the opportunity to explore the role of a white hat hacker with our careers partner. **Term 5: Software Developer** While studying operating system and utility software students will have the chance to discover the careers opportunities within software development.

		media for a given application. • The advantages and disadvantages of different storage. **Data Storage (Numbers)** • How to add two binary integers together. • How to convert positive denary whole numbers to binary numbers. • 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 (Character) ASCII and Unicode** • 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.		○ FTP ○ POP ○ IMAP ○ SMTP • The concept of layers.			

	2.2 Programming fundamentals	2.2 Programming fundamentals	2.2 Programming fundamentals	2.3 Producing Robust Programmes	2.2 Programming fundamentals Practical Programming	Practical Programming	

	Programming Fundamentals
	
	• Introduction to Python.
	• Use of variables, constants, operators, inputs, outputs and assignments.
	• IDLE.
	• PRINT and INPUT.
	• Data Types.
	• Arithmetic operators.
	• Comparison operators.
	• Sequence.
	• Selection:
	  ○ IF
	  ○ ELIF
	  ○ ELSE
		Iteration
		
		• Count and controlled loops.
		• WHILE
		• FOR CSV
		• OPEN
		• CLOSE
		• READ
		• WRITE
		
		String Manipulation
		
		• Concatenation.
		• Slicing.
		• Searching.
			Additional programming techniques
			
			• The use of basic string manipulation.
			• The use of basic file handling operations.
			• The use of records to store data.
			• The use of arrays (or equivalent) when solving problems including 1D and 2D arrays.
			• Random number generations.
			
			Functions
			
			• Subprograms.
			• Functions.
			• Parameters.
				Defensive Design
				Defensive design considerations:
				
				• Anticipating misuse.
				• Authentication.
				
				Input validation
				Maintainability:
				
				• Use of sub programs.
				• Naming conventions.
				• Indentation.
				• Commenting.
					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.
					
					Noel's Music Quiz – Extended Programming Project
					
					• The use of basic string manipulation.
					• The use of records to store data.
					• The use of arrays (or equivalent) when solving problems, including both one-dimensional and two-dimensional arrays.
					• How to use subprograms (functions and procedures) to produce structured code.
					• Random number generation.
						Noel's Music Quiz – Extended Programming Project
						
						• The use of basic string manipulation.
						• The use of records to store data.
						• The use of arrays (or equivalent) when solving problems, including both one-dimensional and two-dimensional arrays.
						• How to use subprograms (functions and procedures) to produce structured code.
						• Random number generation.	

Computer Science – Year Group Overview

	Term 1	Term 2	Term 3	Term 4	Term 5	Term 6
Year 11	**2.1 Algorithms** **2.2 Programming fundamentals**	**2.1 Algorithms** **2.4 Boolean Logic**	**2.5 Programming Languages and Integrated** **Development Environments**	**Pre-Exam Revision**		
	Computational Thinking Principles of computational thinking: • Abstraction. • Decomposition. • Algorithmic Thinking. **Designing, creating and refining algorithms** • Identify the inputs, processes, and outputs for a problem. • Structure diagrams. • Create, interpret, correct, complete, and refine algorithms. • Identify common errors. • Trace tables. **Additional Programming Techniques** Use of SQL to search for data: • SELECT • FROM • UPDATE • INSET • INTO	**Searching and Sorting** Standard searching algorithms: • Binary search. • Linear search. Standard sorting algorithms: • Bubble sort. • Merge sort. • Insertion sort. **Boolean Logic** • Simple logic diagrams using the operators AND, OR and NOT. • Truth tables. • Combining Boolean operators using AND, OR and NOT. • Applying logical operators in truth tables to solve problems.	**Languages** • Characteristics and purpose of different levels of programming language. • The purpose of translators. • The characteristics of a compiler and an interpreter. **The Integrated Development Environment** Common tools and facilities available in an Integrated Development Environment (IDE): • Editors. • Error diagnostics. • Run-time environment. • Translators.	Recap Units 1 & 2 based on the QLA of assessments.		

	Programming Evidence Functions & Procedures	**Application of algorithms**	**Application of algorithms**			
	Noel's Music Quiz – Extended Programming Project • The use of basic string manipulation. • The use of records to store data. • The use of arrays (or equivalent) when solving problems, including both one-dimensional and two-dimensional arrays. • How to use subprograms (functions and procedures) to produce structured code. • Random number generation.	Create, interpret, correct, complete and refine algorithms using; • Pseudocode. • Flowcharts.	Create, interpret, correct, complete and refine algorithms using; • Reference language/high-level programming language.			