Curriculum overview: Computer Science

Content studied during Key Stage 2

No prior learning is assumed due to variation of teaching amongst primary schools. Desirable prior learning is listed below:

Digital Literacy

- Evaluate digital content
- Recognise common uses of technology
- Understand how to seek help online, recognising acceptable and unacceptable behaviour and the reasons why age restrictions are in place.

Information technology

- Creating, opening, saving files within a folder structure
- Basic proficiency in the use of office software
- Use the internet to retrieve and search for information
- Be able to use a range of software to present data and information in a suitable format

Computer Science

- Key programming concepts including sequence, selection, iteration, inputs and outputs
- Design, write and debug programs to achieve a specific goal
- Writing of simple algorithms
- The use of constant and variable values
- Understand the concept of networking and communication to provide services such as the world wide web

Key skills/content requirements at GCSE

Theoretical Computing

- Systems architecture,
- The CPU
- Memory devices
- Storage devices
- The internet
- Types of network
- Wireless networking
- Protocols
- Network security
- Types of software and their features
- Ethical, legal, cultural and environmental concerns in computing
- Data representation
- Key computing algorithms
- Developing algorithms to solve given scenarios

Programming

- Write algorithms to solve given problems.
- Visually represent algorithms using flowcharts.
- Write subroutines to make flowcharts more manageable.
- Use Python to program solutions to given problems.
- Order code so that Python runs in sequence.
- Store values in variables so they can be manipulated.
- Take user input and store the values in variables.
- Manipulate values using mathematical operators.
- Use selection to decide what code to run based on a condition.
- Make decisions using comparative operators.
- Use iteration to repeat sections of code.
- Work out when to use condition-controlled (while) or count-controlled (for) iteration.
- Write functions to better organise programs.
- Pass parameters between functions to make code more efficient.
- Use inbuilt functions contained within libraries.

Curriculum Overview

Theoretical Computing – Throughout study students will develop their understanding of the technology used to power the modern day systems which we rely on. Developing understanding of networking, security, inner workings and the storage of data. In addition to this we provide extensive understanding surrounding the origins of computing and a breadth of history.
Programming – Students will develop their ability to recognise a problem, extract important information, design a solution to a problem, implement using code, test and evaluate fully. Students will be introduced to concepts including variables, sequence, arithmetic operators, Boolean operators, selection, iteration, file handling, regular expressions, time handling, random numbers, subroutines and functions.

	Term 1	Term 2	Term 3	Portable knowledge	Key terms
Year 7	**7.1 ICT Skills** • Create and navigate through folders knowing the difference between home and shared area. • Communicate via email. • Use attachments to add files onto an email. • Make use of tools to improve the accuracy of an email. • Relevant theme picked on a yearly basis to reflect areas of need regarding e-safety. • Understand the risks that exist online. • Know what to do if an issue occurs when on the internet. • Know how to prevent issues from happening when using the internet. • Describe the features that make a strong password. **7.2 Office Software** • Style elements within a document to match setting. • Make use of headers/footers to better organise work. • Separate content using bullet points and numbering where appropriate. • Use keyboard shortcuts to optimise working. • Use images and tables to enhance content. • Make use of the slide master to make template slides with key elements copied across each slide.	**7.3 Spreadsheets** • Identify a cell's location via cell referencing. • Identify a group of cells using a range. • Format and structure data as appropriate in a spreadsheet. • Sort and search for data within a spreadsheet. • Use formulae to perform calculations on data. • Use formulae to more easily locate information from other worksheets. • Use conditional formatting to format certain elements based on a condition. • Model possible scenarios using spreadsheets. **7.4 Hardware** • Define the term peripheral, understanding the difference between input, output and storage peripherals. • Describe how some devices can be both input and output. • Identify the roles of different hardware devices that exist within a computer system. • Explain how certain devices can be used by less able users to assist with their day-to-day lives. • Explain why binary numbers are used by computers • Convert between binary and denary representation of numbers • Add together binary numbers	**7.5 Algorithms** • Develop algorithms to solve problems. • Isolate the key elements of a problem that need to be tackled. • Problem decomposition • Write steps for solving a problem in order to be followed sequentially. • Use flowcharts to visually represent an algorithm. • Use decisions and loops to make algorithms more complex. • Create subroutines to make algorithms easier to follow. **7.6 Microbit** • Understand the differences and similarities between a PC and a micro:bit. • Use block-based code to create programs. • Test programs on a virtual machine. • Implement programs onto hardware. • Write programs that acknowledge the limited hardware capacity of a micro:bit. • Use variables to store data while a program is running. • Deciding what code to run based on conditions. • Repeating code to make program more efficient. • Creating programs based on a given problem.	File and folder access and manipulation. Use of Office software to create documents; understanding of how to best format documents for a given purpose. Use of Excel to manipulate data, create formulae, sort and search and format data. Staying safe online. Use of email, including attachments, highly useful for contacting teachers and sending work. Algorithmic thinking which links directly into all programming units. Visually representing algorithms as a flowchart is relevant to designing programs and visually showing progression of problem solving. Basic programming concepts from block-based language (input, variable, output, selection, iteration).	**Computing Basics & E-Safety:** File Folder Email Email address Attachment Carbon Copy Blind Carbon Copy Username Password Private information Public information Inappropriate content Presentation Formatting Text Font Image Content Spreadsheet Cell Cell reference Range Formula Hardware Peripheral Assistive technology Binary Denary Algorithm Flowchart Sequence Selection Condition Iteration Input Process Output Variable Subroutine

Year 8	**8.1 E-safety** • Relevant theme picked on a yearly basis to reflect areas of need regarding e-safety. • Understand the risks that exist online. • Know what to do if an issue occurs when on the internet. • Know how to prevent issues from happening when using the internet. • Describe the features that make a strong password. **8.2 Cyber Security** • Explain the differences between types of malware. • Describe threats other than malware that could compromise a computer system. • Explain the different methods of protecting a computer system from malware and other threats. • Explain the role of encryption. • Use the Caesar Cipher to encrypt data. • Explain the flaws with the Caesar Cipher. • Scramble data using keyword encryption. • Explain the features that make a good keyword for encryption purposes.	**8.3 Data Representation** • Recall how to convert between binary and denary numbers. • Recall how to add two binary numbers together. • Convert between hexadecimal, binary and denary numbers. • Recognise the similarities between how image and sound are stored within a computer. • Understand how compression affects a file, both in terms of quality and size. • Understand the link between resolution and file size in a computer. **8.4 Python** • Create programs to solve problems in a text-based language. • Describe the difference between data types. • Identify the correct data type to be used for a variable. • Use variables and inputs to take user entry and store it for later use in a program. • Use selection (if statements) to decide what code to run based on a condition. • Perform calculations using mathematical and comparative operators in a program. • Use iteration (while) to repeat sections of code. • Recognise errors when running code and correct them. • Solve given problems using Pythons to create programs.	**8.5 Computer networking** • Describe the internet, and the difference between the internet and the World Wide Web. • Explain the different roles of networking hardware. • Identify the most appropriate piece of networking hardware to use in a given situation. • Understand that different types of network exist, and explain the difference between each type. • Explain the difference between each network topologies, knowing when to use each one. • Describe the role of IP and MAC addresses when building a network. **8.6 Web Development** • Create webpages based on a given brief. • Understand what makes a good house style. • Recognise the file types used on a webpage, both for the webpages itself and the content included. • Recall the impact on loading times that file size of content has. • Understand what makes up a domain name. • Link webpages together using hyperlinks and hotspots. • Use CSS to better format a webpage.	Staying safe online. Knowledge of malware and how to protect a computer system. Knowledge of best practice to protect a computer system from threats (eg. password strength). Binary conversion techniques that translate to most other forms of data representation. Representation of image, sound and characters within a computer system. Problem solving skills within text-based programming. Knowledge of how computers are connected together, both locally and over a wider area. How webpages are developed and the difference between search engines and webpages.	Malware Virus Worm Trojan Hacker Phishing Hacking Firewall Anti-malware Encryption Decryption Encryption key Cipher Binary Denary Bit Byte Hexadecimal ASCII Unicode Pixels Resolution Bit depth Compression Lossy Lossless Data type Comparative operator Comparative Operator Arithmetic operator Syntax Network Internet World Wide Web IP address MAC address Packet Packet switching Local Area Network Wide Area Network House style Layout Colour scheme Content Navigation World Wide Web Internet Web browser Uniform Resource Locator (URL) HTML

Year 9	**9.1. Data Representation**
	• The need for binary representation.
	• How numerical data is represented using binary.
	• Adding numbers using binary representation.
	• How colour information is represented as hexadecimal.
	• How images are represented in a computer system.
	• How sound files are represented in a computer system.
	• Describing the differences between lossy and lossless compression techniques.
	• How characters are represented using ASCII and Unicode.
	• Use of check digits in barcodes.
	• Use of parity bits.
	
	9.2. Computer Graphics
	• The differences between bitmap and vector graphics.
	• The use of layering
	• Design features of logos.
	• Uses of colour.
	• Understanding of how DPI affects quality and file size.
	• Use of complex tools to create digital graphics.
	• Knowing what style of graphics to use based on target audience.
	• Understanding which file type to use in a given situation, explaining why.
		9.3. Computer Hardware
		• Describe the roles of CPU, Hard drive, RAM, ROM, Motherboard, PSU, Fan, CMOS battery.
		• Explain the roles of CPU components including registers, ALU, Control unit, buses.
		• Explain the use of the fetch-decode-execute cycle.
		• Describe the differences between input, output and storage peripherals.
		• Explain how optical, magnetic and solid state devices store data.
		• Explain factors which affect the performance of the CPU.
		• Describe uses of embedded systems
		• To use logic gates to perform computational logic.
		
		9.4. The history of computing
		• Developments in computing led by the following key figures:
		  ○ Ada Lovelace
		  ○ Charles Babbage
		  ○ George Boole
		  ○ John Von Neumann
		  ○ Alan Turing
		  ○ Tim Berners-Lee
		• Key developments in recent computing history from the following figures:
		  ○ Bill Gates
		  ○ Larry Page
		  ○ Mark Zuckerberg
		• Environmental impacts of computing
		• Cultural impacts of computing
		• Social impacts of computing
		• Laws and legislation relating to computing.
			9.5. Designing algorithms
			• Variables and constants
			• Data types
			• Decomposing a problem.
			• Abstracting information from a problem.
			• Creating flowcharts to model an algorithm.
			• Interpreting flowcharts
			• Create subroutines to be reused within a program.
			• Searching algorithms
			• Sorting algorithms
			
			9.6. Python 1
			• Create variables
			• Perform arithmetic operations
			• Use selection in Python
			• Create loops to repeat sections of code
			• Define the difference between count controlled and condition controlled loops.
			• Create lists
			• Manipulating lists
				Conversion between denary, binary and hexadecimal form.
				The use of pixels and resolution of images
				
				Representation of image and sound files in a computer system
				
				Compression techniques used for different file types
				
				Understanding of file types to allow for better optimising of images in later units.
				
				Knowledge of the link between colour and bit depth, and relation to file size.
				
				Creating a product (graphic) for a specific target audience.
				
				The role of hardware and inner workings of the CPU.
				
				The use of logic gates to perform basic calculations
				
				The use of lists and variables to store data
				
				The use of loops to repeat code
				
				The use of if, elif, else to repeat sections of code
				
				Analysing a problem and designing a solution
				
				Implementing a planned solution
				
				Testing and evaluating a program
					Binary
					Denary
					Hexadecimal
					Conversion
					Bit
					Byte
					Kilobyte
					Megabyte
					Gigabyte
					Terabyte
					Resolution
					Pixel
					Bit-depth
					Sampling
					Sample rate
					Compression
					Lossy
					Lossless
					Check digit
					Layers
					Bitmap
					Vector
					Colour
					DPI
					File type
					JPG
					PNG
					GIF
					CPU
					Hard drive
					RAM
					ROM
					Motherboard
					PSU
					Logic gate
					GDPR
					Computer Misuse act
					E-Waste
					Bubble sort
					Insertion Sort
					Merge Sort
					Binary Search
					Linear Search
					Variable
					Constant
					If, Elif, Else
					Condition
					List
					Length

Year 10	**10.1. Creating websites**
	• Recognise what makes an effective or ineffective website.
	• Understand the key HTML tags required to make a webpage.
	• Optimise images for use on a website.
	• Link pages together using hyperlinks and hotspots.
	• Understand how to use classes and IDs in CSS to style a webpage.
	• Write HTML to create the structure of a webpage.
	• Write CSS to create the stylings of a webpage.
	
	10.2. Computer Networking
	• Explain the difference between the internet and world wide web.
	• Explain the need for IP and MAC addresses.
	• State advantages and drawbacks of using computer networks.
	• Describe the differences between LAN, WAN, WLAN and PAN.
	• Compare transmission methods (Ethernet, coaxial, twisted pair, wireless)
	• Describe hardware required to create a network.
	• Describe differences between client-server and peer to peer networks.
	• Name and explain the purpose of networking protocols.
	• Concept of layering
	• Describe the need for encryption.
	• Perform symmetric key encryption.
		10.3. Python 2
		• Writing of data to files
		• Reading of data from files
		• Splitting of data loaded from files
		• Storing data in a structured fashion
		• Explain the need for validation of data
		• Perform validation using regular expressions.
		• Planning a solution to a programming problem
		• Developing effective testing strategies
		
		10.4. Databases
		• The use of tables, fields and records.
		• Differences between flat file and relational databases.
		• Uses of keys
		• Creating relationships
		• Querying a database
		• Creating SQL statements to interrogate a database.
		• Creation of forms
		• Creation of reports
		• Validation methods (Range check, length check, type check, presence check, format check)
			10.5. Python 3
			• Describe the purpose of built in functions
			• Create functions with a specific purpose
			• Describe the use of libraries and their key functions:
			  ○ Random
			  ○ Regular Expression
			  ○ Time
			• NEA specific recap of key concepts.
			
			10.6. NEA
			• Working towards NEA unit (Course requirement)
				Optimisation of images and implications on file size/type.
				
				Understanding the difference between the WWW and the internet.
				
				Scripting webpages required similar skills to programming using code.
				
				Different types of network and the technologies used to connect devices within a network.
				
				The use of protocols to transmit data.
				
				Benefits and drawbacks of using networks to share data.
				
				Use of encryption to protect data across networks.
				
				How code can commit data to a file.
				
				How to validate data using regular expressions
				
				How to create bespoke functions for a specified need.
				
				How databases store data in an organised and persistent format.
				
				The use of databases and their purpose within organisations.
				
				Problem analysis and design of solutions to problems.
				
				Implementation of a solution to a programming problem.
				
				Testing a solution and evaluating a final product.
					Internet
					World Wide Web
					HTML
					CSS
					Script
					Styling
					Hyperlink
					Hotspot
					Class
					ID
					IP address
					MAC Address
					LAN
					WAN
					WLAN
					PAN
					WiFi
					Ethernet
					Network Interface
					SSID
					Switch
					Hub
					Router
					WAP
					Modem
					Peer-to-peer
					Client-server
					Protocol (HTTP, HTTPS, FTP, DNS, DHCP, IMAP, POP, SMTP)
					Writing
					Reading
					File mode
					Split
					For
					While
					Subroutine
					Function
					Library
					Time
					Random
					Regular expression
					Table
					Field
					Record
					Relationship
					SQL
					Form
					Report
					Validation

Year 11	**11.1. NEA** • Working towards NEA unit (Course requirement) **11.2. Software** • Open source vs proprietary software • Operating systems • Translator software • Utility software • Features of IDEs • User interfaces • Software development lifecycles	**11.3. Cyber Security** • Types of malware (Virus, worm, Trojan, spyware, key logger, botnet, DDoS) • Non-malware threats (Phishing, human error, blagging, hacking, network policy) • Identification of vulnerabilities (network forensics, penetration testing) • Prevention & protection mechanisms (Firewall, anti-malware, password, user access levels, encryption methods, verification) **11.4. Exam Preparation** • Content specified by subject revision plan.	**11.5. Exam Preparation** • Content specified by subject revision plan.	Problem analysis and design of solutions to problems. Implementation of a solution to a programming problem. Testing a solution and evaluating a final product. How software is developed over time and by a team. The various types of software which are used and how they are defined. The differences between translator software. The purpose of an operating system and role of the Kernel. The need for varied interfaces. Threats to the security of a computer and sources of data. Mechanisms used to protect computer systems and their security.	Open Source Proprietary Source code Operating system Kernel Integrated development environment Utility software Defragmenter Disk clean-up Encryption Zip file Malware Virus Worm Trojan Spyware Botnet DDoS Phishing Human error Blagging Hacking Network policy Network forensics Pen testing Firewall Password Access level Encryption Verification

GCSE external assessment:

Computer Science uses the GCSE 1-9 grading system, where 9 is the best grade. All examinations are terminal (at the end of Year 11). The assessments are comprised of the following components:

- Paper 1: Computer Systems. The paper lasts for 1 and a half hours and is worth 50% of the GCSE grade.
- Paper 2: Programming and Algorithms. The paper lasts for 1 and a half hours and is worth 50% of the GCSE grade.
- NEA: Programming Project. This unit is completed during 20 hours of timetabled lesson time. This involves the development and documentation of a small program for a specified purpose. A minimum of 20 hours must be spent on this project in order for the qualification to be certified. (Please note that this may be subject to change due to an ongoing exam board review)

SMSC in computer science

Spiritual development in computer science

Students are continually reflecting on their own lives and the lives of others as they look at various Computing case studies. Students debate and formulate their own set of values and beliefs through case studies as they share their own experiences. Computing is an area of rapid development and change, this provides students with the opportunity to reflect upon this progress and potential new technologies which will be developed in time.

Moral development in computer science

Within computing, it is important to consider many areas of the human impact technology has. Society is not only becoming more reliant on technology, but the increasing rate in which computers are updated causes substantial waste, as well as increased carbon footprint in line with their increased production.Students will investigate the the use of social-networking and cyber bulling, whilst learning about the legal implications of immoral acts undertaken online. Students will consider where boundaries should lie and the impact of computing on the environment.

Social development in computer science

Computing can also help all students to express themselves clearly and to communicate. As students' progress through their learning they will consider more complex social needs and are encouraged to research and work to find appropriate solutions to issues that may affect particular groups within society.

Cultural development in computer science

With the increased use of social media sites, people are becoming more culturally aware due to the diversity of content posted online for all of the world to see. Computational thinking encourages problem solving and thinking about how to solve an issue from another perspective – a valuable transferable skill that translates to many aspects of life. Students will consider the positive and negative effects of computing upon various groups of people.