

Curriculum Map Year 8: Computer Science

	Topic	Key Knowledge <i>What will students know by the end of this topic?</i>	Key Skills <i>What skills will students have developed by the end of this topic?</i>	Assessment Opportunities <i>How will student progress be measured? What are the key assessment pieces?</i>
Half Term 1	1. Small Basic	- Screen text: How to display messages on the screen. - Custom looks: How to change window background and text colours. - Inputs & variables: How to capture user answers and save them in neatly named variables. - Making choices: How to use code logic to choose between two, three, or more different outcomes. - Random numbers: How to generate random numbers up to a chosen limit. - Loops: How to make code repeat until a specific condition is met. - Self-teaching: How to use the internet and guides to solve coding tasks on your own.	- Showing messages: Writing commands to print text on the screen. - Styling code: Changing background and font colours in the program window. - Saving data: Setting up inputs and storing data in suitable variables. - Coding paths: Writing if-statements to handle multiple choices and answers. - Adding chance: Generating and applying random numbers in a script. - Repeating tasks: Setting up loops that run until a target is hit. - Finding answers: Using online help files to learn new skills and fix errors.	Self-assessment Quizzes End of unit test with student reflection and actions
Half Term 2	Bebras Challenge	- Bebras rules: The layout and rules of the Bebras competition tasks. - Thinking patterns: Basic ideas of breaking down problems, spotting patterns, and designing steps. - Problem solving: Different ways to think through tricky logic puzzles. - Coding links: How puzzle solving connects to writing computer code.	- Logic analysis: Breaking big, unfamiliar problems into smaller pieces. - Pattern spotting: Finding similarities in puzzles and ignoring unneeded details. - Step-by-step thinking: Creating clear steps or rules to solve challenges fast. - Resilience: Staying confident and trying again when tackling hard puzzles.	Challenge scored remotely Certificates
Half Term 3	2. Spreadsheet modelling	- Sheet layout: Naming key parts of a spreadsheet grid. - Data types: Knowing how to enter data and pick the right format for numbers or text. - Formulas vs Functions: Understanding that formulas use basic symbols ([*], [-], [+]) while functions use named words (SUM, COUNTIF). Cell locking: Why absolute cell references ([*\$]) are used to fix a cell position. - Logic & styling: How IF statements make choices and how conditional formatting changes cell looks automatically.	- Navigating grids: Identifying and using rows, columns, and cell addresses. - Formatting sheets: Setting up data types and styling cells to look clean and professional. - Writing formulas: Creating maths links using cell references and signs ([*], [-], [+]). Locking cells: Applying absolute references correctly when copying code. - Using tools: Writing IF statements, setting up conditional formatting rules, and using SUM, COUNTIF, or SUMPRODUCT.	Self-assessment Quizzes End of unit test with student reflection and actions

	Topic	Key Knowledge <i>What will students know by the end of this topic?</i>	Key Skills <i>What skills will students have developed by the end of this topic?</i>	Assessment Opportunities <i>How will student progress be measured? What are the key assessment pieces?</i>
Half Term 4		Goal Seek: Knowing how this tool adjusts numbers to hit a specific money target.	Target tracking: Using Goal Seek to solve business problems and hit financial targets.	
	3. AI	- Rules vs Data: Knowing how code written by people (rules-based) differs from code that learns from datasets (data-driven). - Machine Learning (ML): Understanding how ML fits into AI and knowing the three main ways machines learn. - AI Pros & Cons: Identifying everyday AI tools alongside their benefits, flaws, and the fact that AI can make mistakes. - FAST Ethics: Knowing how to use AI responsibly by following the FAST principles (Fairness, Accountability, Safety, and Transparency)	- Sorting tech: Spotting and naming different everyday AI applications. - Spotting patterns: Describing how computer models use supervised learning to group or classify data. - Coding AI: Setting up and training basic AI models to accurately recognise text and images. - Evaluating tech: Discussing real-world benefits and issues caused by AI tools. - Responsible use: Applying the FAST rules to check if an AI tool is acting fairly and safely	Self-assessment Quizzes End of unit test with student reflection and actions
	4. Flowol	- Flowcharts & Mimics: Knowing how standard flowchart symbols build an algorithm to control on-screen hardware models (mimics). - Sequences & Loops: Understanding how sub-second delays time flashing lights (Lighthouse) and how loops make systems run forever. - Sensors & Choices: Knowing how systems use inputs (like pedestrian buttons) and selection rules to change traffic signals safely. - Real-world Systems: Understanding how sequence logic maps to real infrastructure like bridges and crossings.	- Mapping Logic: Drawing accurate flowcharts using standard start, process, and decision blocks. - Programming Mimics: Coding Flowol scripts to automate flash patterns for lighthouses and bridge lights. - Handling Inputs: Building interactive systems that alter traffic lights when a crosswalk sensor is triggered. - Debugging Systems: Testing code to find errors, adjusting light sequences, and fixing timing issues.	Self-assessment Quizzes End of unit test with student reflection and actions
	5. Theory	- The CPU: Knowing the job of the processor, how the Fetch-Decode-Execute cycle works, and what factors change its speed. - Storage & Binary: Why computers need permanent storage and why they use 1s and 0s (binary) to handle data. - Data Sizes: Knowing the order of data units from smallest to largest. - Digital Media & Text: Understanding how computers use binary numbers to store picture pixels and text characters.	- Tracking the CPU: Explaining how a processor fetches and runs instructions. - Converting numbers: Changing whole numbers (decimal) into binary code and back again. - Ordering data: Arranging different digital storage sizes accurately by scale. - Translating images: Converting simple grid pictures to and from binary data lines. - Cracking text codes: Translating hidden messages between normal characters and binary text code.	Self-assessment Quizzes End of year test with student reflection and actions
Half Term 5				

	Topic	Key Knowledge <i>What will students know by the end of this topic?</i>	Key Skills <i>What skills will students have developed by the end of this topic?</i>	Assessment Opportunities <i>How will student progress be measured? What are the key assessment pieces?</i>
Half Term 6	6. Quadcopter Challenge	- Online Research: How to find existing technology ideas online and summarise the information in your own words. - Design Logic: Knowing what physical parts, weights, and shapes a drone needs to stay safe and fly stable. - UK Drone Code: Knowing basic safety laws for flying drones in the UK, like height limits (120m) and keeping away from crowds. - Pitching Ideas: Understanding how to structure a team slide presentation to share a project idea clearly.	- Sifting Information: Researching drone examples online and answering design questions using your own language. - Creative Planning: Sketching and drafting an original, safe quadcopter design while tracking size constraints. - Team Presentation: Working as a group to build a digital slideshow and confidently pitch your design to an audience.	Verbal feedback Competition feedback
	7. Standard algorithms	- Search Methods: Knowing how a linear search checks items one by one, while a binary search splits sorted data in half. - Sorting Methods: Knowing how a bubble sort swaps pairs of items, and how a merge sort splits lists down before gluing them back in order. - Efficiency: Understanding why some algorithms are faster than others depending on if data is already sorted.	- Finding Data: Acting out or tracing linear and binary searches to locate specific items in a dataset. - Organising Lists: Manually running bubble sort passes and merge sort trees to put unsorted data into order. - Comparing Code: Looking at data lists to evaluate which search or sort method is the best one to use.	Self-assessment Lesson quizzes / Exit tickets End of topic reflection