Year 9 - Term 1 - Programming

Year group: 9	Subject: Computing
Introduction	The Year 9 Programming curriculum builds directly on the Text based coding proficiency gained in Year 8's Programming - Introduction to python. This unit shifts the focus from simply writing functional code to designing efficient, structured, and modular programs. Students will apply their knowledge of variables, selection, and iteration to graphical problem-solving using the Python Turtle module, before tackling complex, real-world physical computing challenges that demand Advanced algorithms.
Rationale	This MTP is strategically designed to push students to the highest level of KS3 programming competency, preparing them for future study and the application of computer science principles: 1. Mastering Computational Thinking: The term focuses on achieving the pinnacle of KS3 problem-solving: students must learn to decompose complex programming problems into smaller, manageable sub-problems. They will also begin to use abstraction to create more efficient programs. 2. Developing Efficient Code: By introducing Using functions early in the MTP, students learn how to structure their code for clarity and reuse, leading to more advanced programming. 3. Solving Real World Problems: The physical computing element requires students to apply their skills to tangible devices (like Micro:bits, Makeymakeys, Kitronic arcades), demonstrating how code controls physical actions, aligning with the goal of solving real world problems.
Vocabulary:	1. Programming: The overall process of learning how to give instructions to computers using code. 2. Algorithms: Instructions that are followed in a step-by-step manner. 3. Debugging: The crucial skill of acting as a detective to debug any mistakes. This involves using techniques to identify and fix logical and syntax issues in code.

Year group: 9	Subject: Computing
	4. Sequencing: The ability to successfully create a simple program that follows a sequence. This is a fundamental element of flow control. 5. Variables: Elements used in programs to store data. This skill is central to the Secure level in Year 7 and the Emerging level in Year 8. 6. Inputs/Outputs: Basic elements (like buttons) and results (like showing messages) included in simple programs. 7. Selection: The ability to make programs make decisions using 'if/else' statements based on conditions. 8. Iteration / Loops: The process of repeating a block of code by incorporating loops. At the Secure level, students are expected to independently write programs that incorporate loops. 9. Decomposition: The crucial process of breaking down complex programming problems into smaller, manageable sub-problems. This skill is required for mastery in Year 9. 10. Abstraction: A concept used to focus on essential details while hiding complexity, often used to create more efficient programs. This is also a skill required for mastery in Year 9.
Cultural Capital:	The introductory programming unit, focused on fundamentals like Micro:bits and Block based coding, delivers cultural capital through several key areas: • Understanding the Digital World: The curriculum directly addresses the question of "how your favorite games or apps work," emphasizing that "It's all about programming". Students gain insight into how instructions are given to computers using code, which is the core mechanism behind digital media and applications. • Logical Thinking and Creativity: The programming journey is explicitly designed to boost students' logical thinking and creativity, skills that are noted as "helping you in all your subjects". • Investigative Skills: Students immediately learn the crucial skill of debugging, requiring them to "become a detective to debug any mistakes" in their code. This process teaches methodical problem-solving and error identification.

Year group: 9	**Subject: Computing**
	• Foundation for Future Impact: By starting with basic concepts, students begin the journey toward understanding how different apps and devices work together. This foundation is necessary for later progression
Links to National Curriculum	Year 9 builds on Python skills to focus on the efficiency and design of code, including high-level computational thinking skills: • Computational Thinking: Students are taught Decomposition (breaking down complex programming problems into smaller, manageable sub-problems) and Abstraction (focusing on essential details while hiding complexity). • Problem Solving: The curriculum moves towards solving real world problems. • Advanced Programming: Students engage in More advanced programming tasks, incorporating concepts like Advanced algorithms.
Assessment	
What will students be able to do?	**Emerging (E)** Use strong debugging techniques to identify and fix logical and syntax issues in my own and simple programs from others (Mastered Year 8). **Developing (D)** Independently design and write programs that incorporate loops (iteration) and a combination of control structures to solve problems