Curriculum Knowledge Map 2024-25

Year 7	Unit 1: Unlocking Computational Thinking: First Steps in Computing		
Rationale:	This unit lays the groundwork for the entire computing curriculum by introducing students to the fundamental concepts of computational thinking. Students will learn how to break down complex problems into smaller, more manageable parts, recognize patterns, develop step-by-step solutions (algorithms), and create simplified models of real-world situations. This unit aims to foster a problem-solving mindset and provide students with the essential tools to tackle challenges logically and systematically.		
Declarative *What should they know?*	**Definitions:** • Computing encompasses Computer Science, Information Technology, and Digital Literacy. • Computational thinking is a problem-solving process. • Decomposition, pattern recognition, abstraction, and algorithms are key components of computational thinking. • Algorithms are step-by-step instructions for solving problems. • Logical thinking is the ability to think in a clear and organised way. • Models are simplified representations of real-world objects or systems. • Computational models are created with algorithms and run on computers. • Programming languages are used to give instructions to computers. • Software is created by writing code in a programming language.	**Facts:** • There are various types of computers (desktops, laptops, tablets, smartphones). • Computers are used in many aspects of life (education, entertainment, etc.). • Algorithms can be used to solve a wide range of problems. • Models help us understand complex systems and predict behaviour. • Different programming languages have different syntax and rules.	
Procedural *What should they be able to do?*	**Skills:** • Break down simple tasks into smaller steps (decomposition) • Spot patterns in everyday things (pattern recognition) • Focus on the important aspects (abstraction) • Write simple instructions for others to follow (algorithms) • Use logical thinking to solve puzzles and challenges		
Disciplinary Literacy *(Tier 3 Vocab)*	• Computing • Computer Science • Information Technology • Digital Literacy • Computational Thinking • Decomposition • Pattern Recognition • Abstraction	• Algorithms • Logical Thinking • Models • Computational Models • Programming Languages • Software • Syntax	
Assessment	**Formative Assessment:** • **Do Now Recaps**: Daily "Do Now" activities will assess prior knowledge and understanding from previous lessons. • **End of Lesson MS Forms**: Formative quizzes administered through Microsoft Forms will gauge student comprehension of concepts covered during the lesson. • **Ongoing Observation:** Throughout the unit, teachers will observe students' application of computational thinking skills during problem-solving activities. This will provide ongoing feedback to inform instructional adjustments. **Summative Assessment:** • **Computational Thinking Application:** Students will be evaluated on their ability to apply computational thinking skills (decomposition, pattern recognition, abstraction, algorithm design) to solve specific problems presented to them throughout the unit. This assessment will measure the practical application of theoretical concepts and demonstrate students' ability.		
Diversity			

Curriculum Knowledge Map 2024-25

Year 7	Unit 2: Bringing Code to Life: How Software and Hardware Work Together	
Rationale:	This unit reveals the inner workings of computer systems, uncovering the intricate relationship between software (the instructions) and hardware (the physical components). Students will learn how software is executed by the central processing unit (CPU), stored in memory (RAM), and interacts with input and output devices. By understanding this interaction, students will gain a deeper appreciation for how technology functions.	
Declarative *What should they know?*	**Definitions:** • **Software**: Programs that give instructions to computers. • **Hardware**: Physical parts of a computer. • **CPU (Central Processing Unit):** The "brain" of a computer. • **RAM (Random Access Memory)**: Temporary storage for data while programs run. • **Input Devices**: Tools to send data to a computer. • **Output Devices**: Tools to receive data from a computer. • **Programming Languages**: Used to write instructions for computers. • **Machine Code**: Binary instructions computers understand.	**Facts:** • Software includes applications, games, operating systems, and websites. • Hardware includes CPU, memory, storage, and input/output devices. • CPU processes instructions/data • More RAM allows for faster processing • Different input/output devices exist for various purposes.
Procedural *What should they be able to do?*	**Skills:** • Identify different types of software and hardware. • Explain the basic roles of the CPU, RAM, input devices, and output devices. • Describe the relationship between software and hardware. • Understand the general process of how a program is executed by a computer. • Distinguish between input and output devices and give examples of each.	
Disciplinary Literacy *(Tier 3 Vocab)*	• Software • Hardware • Central Processing Unit (CPU) • Random Access Memory (RAM) • Storage • Motherboard	• Input devices • Output devices • Programming languages • Machine code Binary
Assessment	**Formative Assessment:** • **Do Now Recaps**: Daily "Do Now" activities will assess prior knowledge and understanding from previous lessons. • **End of Lesson MS Forms**: Formative quizzes administered through Microsoft Forms will gauge student comprehension of concepts covered during the lesson. • **Ongoing Observation:** Throughout the unit, teachers will observe students' understanding of hardware, students will create labelled diagrams of a computer, identifying key components and their functions. **Summative Assessment:** • **Program Execution Understanding:** Students will be evaluated on their ability to trace the steps involved in executing a simple program, demonstrating their understanding of how different hardware components contribute to the process. This assessment will measure the application of theoretical knowledge to a practical scenario, showcasing students' grasp of the interaction between software and hardware.	
Diversity		

Curriculum Knowledge Map 2024-25

Year 7	Unit 3: Applying Computational Thinking & Computer Systems to Programming		
Rationale:	This unit bridges the gap between theory and practice by applying the knowledge gained in Units 1 and 2 to the creation of a simple game. Using the game (Ms) Pac-Man as a context, students will explore how computational abstractions are used to model game elements and how programming constructs (sequence, selection, iteration) bring these elements to life. This unit reinforces their understanding of key concepts, introduces them to the world of programming, and sparks their creativity.		
Declarative *What should they know?*	**Definitions:** • **Gameplay**: The way a game is played, including its objective, rules, and character interactions. • **Computational Abstraction**: Simplifying complex game elements into manageable representations for programming. • **Game Engine**: The underlying software that powers a game, managing graphics, input, logic, etc. • **Sequence**: The ordered execution of instructions in a program. • **Selection** (Conditional Statements): Programming structures that allow decisions based on conditions (if-else statements). • **Iteration** (Loops): Programming structures that repeat a set of instructions multiple times (for, while loops). • **Debugging**: The process of finding and fixing errors (bugs) in a program.	**Facts:** • Pac-Man has specific gameplay elements (characters, maze, power-ups, scoring system). • These game elements can be modelled using computational abstractions. • Software and hardware interact to run a game. • Sequence, selection, and iteration are fundamental programming constructs used in games like Pac-Man. • Game development involves using tools and programming languages. • Testing and debugging are essential steps in game development.	

Procedural
What should they be able to do?
	Skills:
	• Gameplay:
	  ○ Explain how to play Pac-Man and what the goal of the game is.
	  ○ Identify the different characters in Pac-Man and their roles.
	  ○ Describe the maze and how it affects the game.
	• Computational Thinking:
	  ○ Break down the game into smaller parts (e.g., Pac-Man, ghosts, pellets).
	  ○ Notice patterns in how the characters move or how points are scored.
	  ○ Recognize that game elements like characters are simplified versions of real-world things.
		• Programming Concepts:
		  ○ Understand that instructions in a program happen in a specific order (sequence).
		  ○ Recognize when the game makes a choice (selection), like which way Pac-Man moves.
		  ○ Identify when actions are repeated (iteration), like Pac-Man eating pellets.
		• Game Development:
		  ○ Use a simple tool to create basic game elements (e.g., a maze or a character).
		  ○ Make a character move or respond to simple controls.
		  ○ Combine different game parts together to make a basic game.
		  ○ Check for and try to fix any problems in the game.
		• Software & Hardware:
		  ○ Identify what parts of Pac-Man are software (the game itself, the score) and what parts are hardware (the screen, controller).
		  ○ Understand that the software tells the hardware what to do in the game.	

Disciplinary Literacy *(Tier 3 Vocab)*	• Computational Abstraction • Game Engine • Sequence	• Debugging • Algorithm (though this might be familiar from previous units) • Game Logic	• Selection (or Conditional Statements) • Iteration (or Loops)
Assessment	**Formative Assessment:** • **Do Now Recaps**: Daily "Do Now" activities will assess prior knowledge and understanding from previous lessons. • **End of Lesson MS Forms**: Formative quizzes administered through Microsoft Forms will gauge student comprehension of concepts covered during the lesson. • **Maze Design:** Evaluation of a functional, well-designed maze incorporating decomposition and pattern recognition. • **Game Logic:** Assessment of implemented game logic using sequence, selection, and iteration in character movements and interactions. **Summative Assessment:** • **Digital Portfolio:** Review of the student's documented progress, demonstrating understanding of concepts and application of skills throughout game development.		
Diversity			

Curriculum Knowledge Map 2024-25