SUCCESSFUL CREATIVE HAPPY

Curriculum Knowledge Map 2024-25

Year 9	Autumn 1: Impact of Technology		
Rationale:	This unit aims to foster students' critical thinking and ethical awareness regarding the pervasive influence of technology in society. By examining the documentary "The Social Dilemma" and exploring related topics like data privacy, algorithmic bias, and environmental impact, students will develop an understanding of the complex issues surrounding technology use. They will be challenged to question assumptions, evaluate evidence, and consider the potential consequences of technological advancements. Ultimately, this unit seeks to empower students to become informed and responsible digital citizens who can navigate the digital landscape with discernment and make ethical decisions about their own technology use.		
Declarative *What should they know?*	Definitions: • **Social media:** Platforms for social interaction, content sharing, and communication. • **Algorithm**: A set of rules or instructions that a computer follows to solve a problem or complete a task. • **Ethics**: Moral principles that govern a person's behaviour or the conducting of an activity. • **Digital Divide**: The gap between those who have access to digital technologies and those who do not. • **Personal Data**: Information that relates to an identified or identifiable individual. • **Data Privacy**: The right of individuals to control their personal information. • **E-safety**: Practices and precautions to stay safe online. • **Misinformation**: False or inaccurate information, especially that which is deliberately intended to deceive. • **Filter Bubbles**: Online environments where users are only exposed to information that aligns with their existing views, potentially limiting their exposure to diverse perspectives.	Facts: • Social media platforms generate revenue through user engagement and data collection. • Social media can have negative impacts on mental health and well-being. • Technology raises ethical concerns related to privacy, manipulation, and bias. • Algorithms play a significant role in personalizing online content and can have both positive and negative effects. • Data privacy is a crucial issue in the digital age, and legislation like GDPR aims to protect individuals' rights. • Technology has a significant environmental footprint, from data centre energy consumption to e-waste.	
Procedural *What should they be able to do?*	Skills: • Identify key themes and concerns raised in "The Social Dilemma" related to e-safety, social media use, and potential risks. • Analyse the ethical, social, and psychological impact of technology. • Discuss ethical dilemmas related to technology (data privacy, surveillance, algorithmic bias, digital divide). • Explain how algorithms work and identify different types of algorithms. • Critically evaluate the positive and negative impacts of algorithms. • Explain how personal data is collected, stored, and used by tech companies. • Discuss the importance of data privacy legislation and its impact on users and companies. • Describe the environmental impact of technology, from data centres to e-waste. • Apply critical thinking skills to analyse and discuss complex issues related to technology and society.		
Disciplinary Literacy *(Tier 3 Vocab)*	• Algorithm • Social Media • Ethics • Digital Divide • Data Privacy	• E-Safety • Misinformation • Filter Bubble • Echo Chamber • Persuasive Design	• Addiction • Mental Health • Surveillance • Data Mining • Algorithmic Bias
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. • **Ethical Dilemma Analysis:** Students analyse a real-world ethical dilemma related to technology (e.g., facial recognition, AI in decision-making) and propose ethical solutions. Summative Assessment: • **Research Project:** Students conduct in-depth research on a specific topic (e.g., algorithmic bias, data privacy laws, environmental impact of technology) and present their findings. • **Unit assessment:** Impact of Technology		

Curriculum Knowledge Map 2024-25

Year 9	Autumn 2: Computational Thinking & Algorithms	
Rationale:	Building upon their understanding of technology's societal impact, this unit delves into the core of computational problem-solving: algorithms. Students will explore how algorithms shape our digital world, from everyday applications to advanced artificial intelligence like AlphaGo. They will learn to design, analyse, and optimise algorithms, fostering computational thinking skills that are essential for understanding and creating technology. Through hands-on activities and real-world examples, students will gain a deeper appreciation for the power of algorithms, their ethical implications, and their influence on society and culture. This unit will not only prepare students for future programming endeavours but also equip them with the critical thinking skills necessary to navigate an increasingly algorithmic world.	
Declarative *What should they know?*	Definitions: • **Algorithm**: A step-by-step set of instructions to solve a problem or complete a task. • **Algorithm Complexity**: How much time and memory an algorithm uses. • **Flowchart**: A visual diagram using symbols to represent the steps of an algorithm. • **Pseudocode**: A plain language description of algorithm steps, resembling code but not specific to any language. • **Search Algorithm**: A method for finding a specific item in a collection of data. • **Sort Algorithm**: A method for arranging items in a collection in a specific order. • **Machine Learning**: A type of artificial intelligence where computers learn from data. • **Artificial Intelligence (AI)**: The ability of machines to mimic human intelligence.	Facts: • Algorithms are used in many everyday activities. • Algorithms vary in complexity depending on the problem they solve. • Flowcharts and pseudocode are tools for designing and representing algorithms. • Different search and sort algorithms have varying efficiency. • AlphaGo is a significant example of AI in the field of game playing. • AI has potential applications in many fields but raises ethical concerns.
Procedural *What should they be able to do?*	Skills: • Identify real-world examples of algorithms. • Explain the basic concept of algorithm complexity. • Understand and use flowchart symbols. • Break down a problem into smaller parts. • Design simple algorithms using flowcharts and pseudocode. • Trace the logic of an algorithm using a flowchart. • Understand the relationship between flowcharts and programming constructs.	Skills: • Understand the purpose of search and sort algorithms. • Explain the concept of Big O notation for algorithm analysis. • Describe the development and significance of AlphaGo. • Discuss the potential benefits and risks of AI in various fields. • Consider the ethical implications of AI. • Research and present information on AI-related topics.
Disciplinary Literacy *(Tier 3 Vocab)*	• Algorithm • Complexity (Time & Space) • Flowchart • Pseudocode • Search Algorithm • Sort Algorithm	• Machine Learning • Artificial Intelligence (AI) • Sequence • Selection (or Conditional Statement, If-Then-Else) • Iteration (or Loop)
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. • **Algorithm Design:** Students design an algorithm (using flowcharts or pseudocode) to solve a more complex problem (e.g., sorting a list of names, finding the shortest path between points). • **AI/Machine Learning Research Project:** Research and present on AlphaGo or a similar AI application. Assess understanding of how it works, its significance, and its potential impact. Summative Assessment: • **Research Project:** Students conduct in-depth research on a specific topic (e.g., algorithmic bias, data privacy laws, environmental impact of technology) and present their findings. • **Unit assessment:** Computational Thinking & Algorithms	

Curriculum Knowledge Map 2024-25

Year 9	Spring 1: Theoretical Foundations for Python Mastery
Rationale:	Building on their exploration of computational thinking, algorithm design, and the real-world impact of AI in the previous unit, this unit focuses on equipping students with the essential Python programming knowledge and skills needed to bring their algorithmic ideas to life. By mastering core Python concepts, students will be able to translate their understanding of algorithms into functional code, effectively bridging the gap between theoretical problem-solving and practical implementation.
Declarative *What should they know?*	Definitions: • **IDE (Integrated Development Environment):** A software application that provides tools for writing, testing, and debugging code. • IPO (Input-Process-Output) Model: A framework for describing the steps in a computer program. • Data Types: Categories of values that a programming language can work with (e.g., strings, integers, floats, Booleans). • Variable: A named storage location in a program that holds a value. • Type Casting: The process of converting a value from one data type to another. • List: A data structure that can store an ordered collection of items. • Sequential Execution: Instructions in a program are executed one after another in the order they appear. • Indentation: The use of spaces or tabs to indicate the structure of code in Python. • Statement: A line of code that performs an action. • Expression: A combination of values, variables, and operators that produces a result. • Code Block: A group of statements that are executed together as a unit. • Boolean Logic: A system of logic that uses the values True and False. • Comparison Operators: Symbols used to compare values in a program (e.g., >, <, ==, !=). • Conditional Statements: Structures (if, elif, else) that allow a program to make decisions based on conditions. • Nested Conditional Statements: Conditional statements placed within other conditional statements. • Logical Expressions: Expressions that use Boolean operators (and, or, not) to create more complex conditions. • Loop: A programming construct (for, while) that repeats a block of code multiple times. • Iteration: The process of repeatedly executing a block of code within a loop. Facts: • Python has various IDEs available for programmers. • Python supports different data types, each with specific uses. • Type casting allows for flexibility in data manipulation. • Lists are versatile and can store different data types. • Indentation is crucial for readability and functionality in Python. • Boolean logic and comparison operators are essential for making decisions in code. • Loops are powerful tools for automating repetitive tasks and processing data.
Procedural *What should they be able to do?*	Skills: • Use a Python IDE to write, test, and debug code. • Analyse problems using the IPO model. • Declare and use variables of different data types. • Perform type casting to convert between data types. • Create, manipulate, and access elements within lists. • Use correct indentation to structure Python code. • Write and evaluate Boolean expressions. • Use comparison operators to compare values. • Write conditional statements to control program flow. • Write loops (for, while) to repeat code blocks. • Break out of or continue loop execution as needed.
Disciplinary Literacy *(Tier 3 Vocab)*	• Integrated Development Environment (IDE) • Input-Process-Output (IPO) Model • Data Types (String, Integer, Float, Boolean) • Variable • Type Casting • List • Sequential Execution • Indentation • Statement • Expression • Code Block • Boolean Logic • Comparison Operators (>, <, ==, !=, etc.) • Conditional Statements (if, elif, else) • Nested Conditional Statements • Logical Expressions • Loop (for, while) • Iteration
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. Summative Assessment: • **Unit assessment:** Programming Fundamentals

Curriculum Knowledge Map 2024-25

Year 9

Spring 2: Crafting Interactive Stories: A Text-Based Adventure Game Project

Rationale:

Building on proficiency in Python fundamentals, this unit empowers students to apply their knowledge and skills in a creative and engaging context: the development of a text-based adventure game. Students will leverage modular programming techniques to design, implement, and refine an interactive story, incorporating elements of computational thinking and algorithmic logic. Through this project-based approach, students will solidify their understanding of core Python concepts, hone their problem-solving skills, and gain valuable experience in software development, debugging, and the creation of a digital portfolio to showcase their work.

Declarative

What should they know?

Definitions:

- **Modular Programming**: Breaking a program into smaller, reusable modules or functions.
- **Text-Based Adventure Game**: A game where players interact through text commands, exploring a world and solving puzzles.
- **Function**: A self-contained block of code that performs a specific task.
- **Input Parameter**: Data passed into a function to be used in its calculations.
- **Return Value**: The result a function sends back after completing its task.
- **Game Logic**: The rules and procedures that determine how a game functions.
- **Game State**: The status of all elements in a game (player location, inventory, etc.).
- **Debugging**: The process of finding and fixing errors in code.
- **Digital Portfolio**: A collection of work demonstrating skills and knowledge.

Facts:

- Modular programming makes code easier to manage, reuse, and maintain.
- Text-based adventure games have a specific structure and rely on player choices.
- Python is a programming language suitable for creating text-based games.
- Game logic is often implemented using conditional statements and loops.
- Testing and debugging are crucial for functional code.
- Digital portfolios showcase technical skills and learning progress.

Procedural

What should they be able to do?

Skills:

- Explain the benefits of modular programming and how it's used in game development.
- Break down (decompose) a text-based game into its core components.
- Create a visual representation (map or flowchart) of a game's structure.
- Translate a visual design into Python code structure.

- Define and use functions with input parameters and return values.
- Use variables to track game state elements.
- Implement game logic using conditional statements (if/else) and loops.
- Handle user input and provide appropriate feedback within the game.

- Identify and fix common errors in Python code (debugging).
- Explain the importance of testing and debugging.
- Create a digital portfolio that showcases their game project effectively.
- Reflect on their learning process and demonstrate growth through the portfolio.

Disciplinary Literacy

(Tier 3 Vocab)

- Modular Programming
- Function
- Input Parameter
- Return Value
- Game Logic
- Game State

- Debugging
- Digital Portfolio
- Conditional Statements (if/else)
- Loops (for, while)
- Variable (may be familiar from previous units)

- String (may be familiar from previous units)
- Boolean (may be familiar from previous units)
- User Input
- Syntax Error
- Logic Error

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.
- **Text-Based Adventure Game:** Students create their own text-based adventure game, demonstrating their understanding of modular programming, game logic, and Python coding skills.

Summative Assessment:

- **Code Review & Reflection**: Students submit their game code and a written reflection on their development process. The reflection should address challenges faced, solutions implemented, and how they applied computational thinking concepts.
- **Unit assessment:** Python Programming

Curriculum Knowledge Map 2024-25

Year 9 — Summer 1: Unveiling the Machine: How your Adventure Game comes to life

Rationale:

This unit aims to deepen students' understanding of the fundamental relationship between their text-based adventure game and the underlying computer hardware. By exploring the inner workings of the CPU, the representation of data in binary, and the role of memory in game execution, students will gain a comprehensive understanding of how their Python code is translated into instructions that the computer can process. This unit will bridge the gap between high-level programming concepts and low-level machine operations, fostering a deeper appreciation for the intricate processes that occur within a computer system when running their game.

Declarative

What should they know?

Definitions:

- CPU (Central Processing Unit): The brain of the computer that executes instructions.
- Memory: Temporary storage for data and instructions while the computer is running.
- Storage: Permanent storage for data and programs (e.g., hard drive, SSD).
- Input/Output Devices: Tools for interacting with a computer (e.g., keyboard, mouse, monitor).
- Von Neumann Architecture: A design model for computers where instructions and data are stored in the same memory.
- Control Unit (CU): The part of the CPU that manages the flow of instructions.
- ALU (Arithmetic Logic Unit): The part of the CPU that performs calculations and logical operations.
- Registers: Small, fast storage areas within the CPU.
- Fetch-Decode-Execute Cycle: The basic operation cycle of a CPU.
- Binary Code: A system of representing data using only two digits (0 and 1).
- ASCII/Unicode: Character encoding standards that represent text in binary form.
- Variable: A named storage location in memory for data.
- Memory Address: A unique identifier for each location in memory

Facts:

- Different hardware components have specific roles in running software.
- The CPU is responsible for processing instructions and data.
- The Von Neumann architecture is the foundation of most modern computers.
- Computers use binary code to represent all data and instructions.
- High-level code (like Python) is translated into binary code for the CPU.
- Game data is stored in binary form within the computer's memory.
- Variables are used to store and track game state (e.g., player location).

Procedural

What should they be able to do?

Skills:

- Explain the role of each hardware component (CPU, memory, storage, I/O devices) in executing a program.
- Describe how these components interact with each other to run a text-based adventure game.
- Explain the concept of the Von Neumann architecture and its significance.
- Describe the internal structure of the CPU and its main components.
- Explain the steps of the fetch-decode-execute cycle.
- Understand how binary code represents data and instructions.
- Describe the relationship between high-level code, assembly language, and machine code.
- Explain how text, numbers, and other data types are represented in binary.
- Understand how variables are used to store and access data in memory.
- Explain how memory addresses are used to locate data in memory.
- Describe how game state is stored and updated in memory.

Disciplinary Literacy

(Tier 3 Vocab)

- Central Processing Unit (CPU)
- Memory (RAM)
- Storage (Hard Drive, SSD)
- Input/Output (I/O) Devices
- Binary Code
- Fetch-Decode-Execute Cycle
- Memory Address
- Von Neumann Architecture
- Control Unit (CU)
- Arithmetic Logic Unit (ALU)
- Registers
- Assembly Language
- ASCII
- Unicode

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.
- **Algorithm Design:** Students design an algorithm (using flowcharts or pseudocode) to solve a more complex problem (e.g., sorting a list of names, finding the shortest path between points).
- **AI/Machine Learning Research Project:** Research and present on AlphaGo or a similar AI application. Assess understanding of how it works, its significance, and its potential impact.

Summative Assessment:

- **Research Project:** Students conduct in-depth research on a specific topic (e.g., algorithmic bias, data privacy laws, environmental impact of technology) and present their findings.
- **Unit assessment:** Computational Thinking & Algorithms

Curriculum Knowledge Map 2024-25

Year 9	Summer 2: Cyber Security & Software Development		
Rationale:	This final unit empowers Year 9 students to become informed and responsible digital citizens by deepening their understanding of cybersecurity threats and defensive strategies. Building upon prior knowledge of computational thinking and programming, students will explore the intricacies of secure systems, learn to protect their personal information, and create a practical login system using Python. This culminates in students developing the critical thinking and practical skills needed to navigate the digital world safely and confidently.		
Declarative *What should they know?*	Definitions: • **Cybersecurity**: The practice of protecting systems, networks, and data from digital attacks. • **Confidentiality**: Ensuring that only authorized individuals can access sensitive information. • **Integrity**: Maintaining the accuracy and consistency of data. • **Availability**: Ensuring that systems and data are accessible to authorized users when needed. • **Malware**: Malicious software designed to harm computer systems. • **Phishing**: Fraudulent attempts to obtain sensitive information (e.g., usernames, passwords, credit card details) by disguising as a trustworthy entity. • **Social Engineering**: Manipulating individuals to divulge confidential information or perform actions. • **DDoS Attack**: A Distributed Denial of Service attack, where a network is flooded with traffic to disrupt its normal operation. • **Authentication**: Verifying the identity of a user or device. • **Encryption**: Converting data into a secret code to prevent unauthorized access. • **Plaintext**: The original, unencrypted data. • **Ciphertext**: The encrypted data. • **Encryption Keys**: Used to encrypt and decrypt data.		Facts: • Cyber threats are constantly evolving and can have significant consequences. • Strong passwords are essential for protecting online accounts. • Encryption is used to secure sensitive data like financial transactions and communications. • Social engineering attacks prey on human psychology and trust. • Python can be used to create secure authentication systems.
Procedural *What should they be able to do?*	Skills: • Identify and classify different types of cyber threats. • Create and manage strong passwords. • Evaluate the pros and cons of different authentication methods.	• Explain the concept of encryption and its different types. • Identify real-world examples of encryption in use. • Recognize common social engineering tactics and apply strategies to avoid falling victim. • Create a simple login system in Python using authentication techniques	• Implement basic error handling in Python for login security. • Follow best practices for personal cybersecurity. • Research and stay informed about current cybersecurity threats and trends.
Disciplinary Literacy *(Tier 3 Vocab)*	• Cybersecurity • Confidentiality • Integrity • Availability • Malware	• Phishing • Social Engineering • DDoS Attack (Distributed Denial of Service) • Authentication • Two-Factor Authentication (2FA)	• Encryption • Plaintext • Ciphertext • Encryption Key • Brute force Attack
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. • **Authentication System:** Students will be assessed on their ability to create a functional Python login system that securely verifies user credentials and uses error handling. Summative Assessment: • **Unit assessment:** Cyber Security		