Computing: SPECIFIC LEARNING INTENTIONS

Know (blue), Understand (orange) and Do (green) | Learning Check (Purple)

YEAR GROUP:	Year 9 – 2025-2026					
	Autumn Term 1	Autumn Term 2	Spring Term 1	Spring Term 2	Summer Term 1	Summer Term 2
	Python Programming	Python Programming Advanced	App Development	Practical Algorithms	Cyber Security	Impact on Society
Granular learning intentions/success criteria **(What will pupils know, be able to do and understand)**	**Programming with Inputs and Outputs** • Define what a variable is • Identify common data types • Trace through a program and make predictions • Write a program that asks four questions • User variables to store user input and display a meaningful output **Programming with Selection** • Define the purpose of selection in programming • Identify the operators used for less than and greater than(< >) • Identify the different arithmetic operators (+, -, *, /) • Identify the key syntax for writing a selection statement • Predict the output of selection-based programs by tracing through code. • Explain the difference between IF, ELSEIF, and ELSE statements. • Write programs that use IF...ELSE to make decisions based on user input. • Use IF...ELSEIF...ELSE to handle multiple conditions. • Modify and debug selection statements to ensure correct functionality. **Programming with Iteration** • Define iteration and explain its purpose in programming. • Know that a while loop repeats until a condition is met • Predict the outcome of a block of code that uses iteration. • Create a program that uses a while loop • Calculate the number of times a while loop will repeat **Create a quiz or decision-making program that uses, inputs, outputs, variables and IF...ELSEIF...ELSE.** • Know that a for loop repeats a set number of times	**Programming with Arrays** • Define what an array (list) is • Recognise array indexes start at 0 • Explain why arrays are more efficient than multiple variables • Create an array and store multiple values • Access and display values from an array • Trace through code that uses an array and determine the output **Programming with Modules** • Identify the syntax for importing a module (import module) • Recognise common modules such as random • Import and use the random module to generate random numbers • Combine modules with variables and loops in a program **Summative Assessment 1** **Programming with Sub-routines** • Define what a subroutine is • Identify the syntax for defining and calling a subroutine • Recognise the difference between a function and a procedure • Explain why subroutines improve readability and reusability • Understand how subroutines help break problems into smaller tasks • Write a program with a subroutine that performs a simple task • Add parameters to pass data into a subroutine	**Introduction to App Development** • Define what an app is • identify common features of mobile apps • Recognise examples of successful apps • Explain why apps are designed with a specific audience in mind • Analyse existing apps for features and purpose • Compare different apps and identify strengths/weaknesses **App Development – Mood boards** • Define what a moodboard is • Identify common elements (colours, fonts, icons, images) • Create a moodboard for a chosen app idea • Select appropriate colours, fonts, and imagery to match the target audience **App Development – Wireframes** • Define what a wireframe is • Recognise common design conventions in apps • Explain why planning is important before building an app • Understand how user needs affect design choices • Create a wireframe for a simple app • Include annotations of key design features and interactions **Summarise the concept for an app (name, purpose, audience, features), use a moodboard and wireframe to show your idea.**	**Introduction to Algorithms** • Define an algorithm as step-by-step instructions • Define abstraction and decomposition • Recognise examples of successful apps • Explain how abstraction and decomposition help when creating algorithms • Use abstraction, decomposition and algorithmic thinking to solve problems **Algorithms – Tower of Hanoi** • Identify the rule for tower of Hanoi • Recognise the minimum number of moves needed ($2^n - 1$) • Explain how abstraction and decomposition help when creating algorithms • Solve the Tower of Hanoi with 3 or more disks • Record the steps as an algorithm **Algorithms – Searching** • Identify when binary search can be used (ordered lists only) • Recognise that linear search checks each item in turn • Explain the difference between linear and binary search • Understand why binary search is more efficient than linear search on large datasets • Perform a linear search on a dataset • Perform a binary search on an ordered dataset **Key words check and comparing the efficiency of algorithms**	**Hacking** • Define hacking • Identify types of hackers: black-hat, white-hat, grey-hat • Recognise real-world examples of hacking incidents • Explain why people hack systems • Understand the consequences of hacking • Suggest ways to reduce the risks of hacking • Practice hacking techniques **Common Threats** • Define common threats: phishing, malware, ransomware, spyware • Identify methods used to deliver these threats (emails, links, downloads) • Explain how threats can damage systems and steal data • Understand how personal behaviour can reduce risks • Identify suspicious emails/messages • Create strong passwords and explain why they are effective **Summative Assessment 2**	**Mobile and Wearable Tech** • Identify examples of mobile and wearable technologies • Recognise common uses (communication, fitness, entertainment) • Explain how these technologies change daily life • Understand benefits and drawbacks • Debate and discuss the impact of these technologies on society **Automation and AI** • Define automation and artificial intelligence (AI) • Identify examples of jobs affected by automation • Explain how automation and AI impact employment and industries • Understand ethical issues linked to AI use • Debate and discuss the impact of these technologies on society **Environmental Impact** • Define e-waste and its impact • Identify how technology uses energy and resources • Explain how computing contributes to environmental issues • Understand the importance of recycling and sustainability • Suggest ways to reduce the environmental impact of computing • Create a poster or guide on sustainable tech use **GCSE style 8-mark question on the impact that technology has on society**

	• Explain the benefits of using loops • Compare FOR loops vs. WHILE loops and determine when to use each. • Create a program that uses a for loop • Calculate the number of times a for loop will repeat **Combining Programming Techniques to Create a Program for a Criteria** • Identify key requirements in a problem • Plan a program by breaking down the problem into smaller steps • Select the best type of iteration (FOR or WHILE loops) to repeat tasks efficiently. • Develop the code in parts • Test/run the code during development.		**App Development – Implementation** • Identify the stages of app development (design → build → test) • Understand why testing is an essential stage in development • Recognise the importance of design before development • Build a basic app prototype • Test and refine the app based on feedback	**Algorithms – Sorting** • Identify the steps involved in the merge sort and bubble sort • Understand why bubble sort is inefficient for large datasets • Apply bubble sort to order a small dataset • Apply the merge sort on a small data set		**Surveillance & CCTV** • Define surveillance and identify common examples (CCTV, online tracking) • Recognise the purpose of surveillance in society • Explain the balance between privacy and security • Understand how surveillance impacts behaviour • Write and argument for or against surveillance **Emerging Technology** • Identify examples of emerging technologies (VR, AR, 5G, quantum computing) • Recognise their potential applications • Explain benefits and risks of emerging technologies • Understand how new tech might change society