Subject	**Computer Science**	**Year Group**	**10**			
	Unit 1 Comp. Thinking	Unit 1 Principles of Comp.	Unit 2 Comp. Thinking	Unit 2 Principles of Comp.	Unit 3 Comp. Thinking	Unit 3 Principles of Comp.
Scheme title	**Intro to programming** **Decomposition and Algorithms**	**Binary**	**Sequence, Selection and Iteration**	**Binary Operations**	**Subprograms**	**Stored Program Computers**
Knowledge in sequence	How do we write Python programs?	How do we convert binary to denary? What are unsigned Integers? Is binary arithmetic different? What is hexadecimal?	How do we write programs that use selection? How do we write for (in range) loop and why are they useful? What are procedures / functions and how do we use them?	How do we represent negative signed integers using two's complement? How can binary shifts be used to multiply and divide? How is text represented in binary using ASCII?	How do we use subprograms to decompose problems? What is the difference between local and global variables and when do we use them?	What is Von Neumann architecture? What is the FDE Cycle and how do we measure computer performance? Why do we need secondary storage and what are the advantages of the different methods?
Purpose of scheme	Introduce and define the term 'program' Construct simple programs using inputs and outputs Recognise primitive data types (int, real, char, string) Define the term 'variable' Create variables of all types Define the term 'decomposition' Define the term 'algorithm'	To explore what is meant by a digital computer. To explore what is meant by the terms binary, bit and byte. To explore how computers store numbers.	To explore how conditions can be used to control the flow of code. To explore how data can be represented by lists. To explore the different means of looping code.	To explore how signed integers are stored in binary. To use left and right shifts to multiply and divide binary values. To explore how text is stored in binary. To understand why hexadecimal is used by pgorammers and be able to convert between hexadecimal and binary.	To understand what is meant by a subprogram. To explore why subprograms are used. To make use of parameters.	To explore what is meant by the stored prgoram concept. To describe the hardware components used during the FDE cycle. To undertand how different components impact performance. To identify the different secondary storage and explore how they work. To identify the advantages and disadvantages of different storage methods.
Skills	Use inputs and outputs in Python Layout code to be readable and maintainable Correct errors in programs Use variables in algorithms and programs	To be able to convert between binary and denary. To be able to determine the number of unique bit patterns that can be represented by a binary pattern of a given length (2^n) To be able to add two binary values.	Use 'if' and 'if else' in code Use 'if elif else' in code Use repetition (condition-controlled loops) in code	To be able to convert a signed integer into binary using two's complement. To be able to perform binary shifts on 8-bit values. To be able to convert text into binary using an ASCII table. To be able to convert hexadecimal in binary and viceversa.	Writing effecient programs. Create functions and procedures.	Knowledge recall Evaluation skills Expression construction
Key words	Program, algorithm, data types, input, output, syntax	Bit, Byte, nibble, binary, denary, integer, overflow	Sequnce, selection, looping Boolean statements Operators	Bit, Byte, nibble, binary, denary, hexadecimal signed integer, unsigned integer, overflow.	Function, procedure, parameter, return value.	
End point	To construct programs with inputs and outputs that use variables with various data types	To be able to give the 8-bit binary equivalent of a denary value and vice versa	Solve problems using code Use repetition in code Use selection in code	Describe the limitations of ASCII. To convert between binary, hex and denary number bases with	To be able to implement appropriate functions and procedures when solving problems	To be able to implement appropriate functions and procedures when solving problems
Assessment Methods	Skills are teacher assessed in lessons. End of unit test.	In class quizzes. End of unit test.	Skills are teacher assessed. End of unit test.	In class quizzes. End of unit test.	Skills are teacher assessed. End of unit test.	End of unit assessment.
Hours	7	7	7	7	7	7

Unit 4 Comp. Thinking	Unit 4 Principles of Comp.	Unit 5 Comp. Thinking	Unit 5 Principles of Comp.	Unit 6 Comp. Thinking	Unit 6 Principles of Comp.
Iterating through lists and strings	**Software**	**Files**	**System Threats**	**Turtle**	**Networks**
How do we apply string/list methods? How does indexing work in 2D lists? How do we iterate through lists using loops? What is a linear search and how might it look in code?	What is an OS? How does an OS manage files? How does an OS manange processes? How does an OS manage peripherals? What is utility software?	How do we apply a merge sort to a dataset? How do we read external files? Why do strings needs processing when reading and writing data? How do we write data to external files?	What is Malware? What are hackers and what are their objectives? What is social engineering? What is data level protection? What does robust software look like?	How do we use the turtle module to draw: -Pens and lines -Coordinates and polygons -Colours and filling -Circles How do we use subprograms in turtle to program effeciently?	What are LAN and WANs? Network Speeds and how is the performance of a network measured? What are the differences between wired and wireless connections? How do we arrange devices on a network (toplogies)?
How to computers use data structures? To explore the characteristics of one-dimensional and two-dimenionsal lists. To use loops to iterate through lists.	To explore the different roles of the operating system including files management, process management, peripheral mangagement and user management. To explore the different types of utility software and to describe their roles.	To explore ways of sorting data. To practise reading and writing to files. To explore how 2D data can be stored in files.	To explore different types of cyber attack and the damage they cause. To explore the different characteristics of different types of malware. To explore ways of keeping a system safe from attacks and malware.	To practise decomposing a problem. To practise using the turtle module to draw shapes and patterns.	To explore the characteristics of wired and wireless networks including hardware, topologies and protocols.
1D and 2D indexing. Applying loops in Python.	Knowledge recall Evaluation skills	Reading and writing files in Python. Using loops to iterate through data. Processing data to ensure it is suitable.	Knowledge recall Evaluation skills	Decomposition and abstraction Writing programs using loops and subprograms.	Knowledge recall Evaluation skills Expression construction
Array, list, index, algorithm	Operating system, perhipheral, directory, algorithm.	Data, algorithm, read, write, append, CSV	Malware, cyber attack, bot, virus, worm, spyware, Trojan horse, keylogger, social engineering, ransomware, hacker.	Decomposition, abstraction, Turtle, color	LAN, WAN, protocol, topology, Wi-Fi. switch
To be able to find values in lists using a loop.	Describe how an OS allocates each active process a share of CPU time Explain the role of a device driver	To read and write data structures to and from files including appropriate processing of data	To identify bad practices and explain ways to minimise risks.	Use the turtle module, programming constructs, and subprograms to create images	State two advantages and two disadvantages of using wireless to connect devices on a LAN rather
Skills are teacher assessed. End of unit test.	End of unit assessment.	Skills are teacher assessed. End of unit test.	End of unit assessment.	Skills are teacher assessed. End of unit test.	Skills are teacher assessed End of unit test.
7	7	7	7	7	7