Subject:	GCSE Computer Science	Year	10	Ability	Mixed

Term / Date(s)	Component 1.2 (Strand 1), 2.1, 2.2 (Strand 2)	Component 1.1 (Strand 1), 2.1, 2.2 (Strand 2)
Topic	**Strand 1** - Memory and Storage **Strand 2** - Programming	**Strand 1** - System Architecture **Strand 2** - Programming
Topic overview **Pupils will learn…**	**Strand 1** – Students will understand how computers store data in binary digits, identify units of measurement for data, calculate file sizes, explore how binary data can be used to represent numbers, text, images, sound, and the need for compression when transmitting data that is used by the main memory to store data currently being used, or long term storage for later use. **Strand 2** - Students will develop and implement algorithms to solve a set of computational problems, debug errors and refine solutions.	**Strand 1** - Students will investigate the relationship between the Central Processing Unit (CPU), how it processes data and instructions and explore component parts, their roles and their impact upon the performance of computer systems. **Strand 2** - Students will develop and implement algorithms to solve a set of computational problems, debug errors and refine solutions.
Components	**Strand 1** • Name, place in the correct order, and convert values between the units of data storage to find the most appropriate unit when calculating a file size or capacity • Understand that data needs to be converted into binary format to be processed by a computer to link all data to electrical states within transistors inside the CPU • Practice calculations of data capacity requirements for text, database, image, and sound files to recognise formulae and calculate the correct file size for a specific file type • Understand the relationship between denary (base 10), binary (base 2) and hexadecimal (base 16) number systems to recognise why each is used to represent specific data • Convert between binary and denary numbers (up to 8 bits) to demonstrate how a computer can recognise and present values recognisable to humans • Add two binary integers together and identify the concept of overflow to understand how computers can manage values greater than can be represented with 8 bits • Convert between binary and hexadecimal to understand the benefits of presenting long sequences of data in hexadecimal • Perform binary shifts and recognise the impact of shifting to understand how computers can perform specific arithmetic operations efficiently • Understand how binary is used to represent characters, represented as a set of pixels (in the form of image) or a set of digital audio samples (to form a sound file), to link data we commonly use to the way it is arranged and managed by a computer system • Identify character sets such as ASCII and Unicode to describe the relationship between the number of bits per character and the number of characters that can be represented. This links to the development of computer systems over time into the globally connected devices we use today • Identify the impacts of colour depth and resolution (on the quality and file size of an image) and the effect of sample rate, duration, and bit depth (on the quality and file size of a sound file) to build links between the amount of data being used to represent the data and its quality • Know the concept of metadata to identify how a computer system can arrange data to form a specific type of file (such as images or audio) • Explore the need for and features of lossy and lossless compression to understand how data can be compressed to be transmitted across devices (such as through the internet) so that we can access it quickly • Identify the role played by RAM and ROM within a computer system to understand that computers must have space to store the instructions they are currently working on • Recognise the role of storage within a computer system, identify the three types of storage technology and compare them according to characteristics to be able to identify the best type of storage for a given scenario as well as the need for long term storage within a modern computer system **Strand 2** • Describe and use abstraction and decomposition when defining, and refining problems / creating solutions to identify how to break problems down into smaller, manageable parts with relevant detail, leading to more effective solutions • Recognise the purpose of shapes used within flowcharts to present algorithms to read solutions and predict outcomes and possible errors • Create algorithms using flowcharts and a high-level programming language (Python) to solve a set problem, correct a solution or improve it	**Strand 1** • Describe the purpose of the CPU to recognise its critical role in helping computers to function • Recognise the steps the CPU performs for the computer to function (the FDE cycle) • Understand that data needs to be converted into binary format to be processed by a computer to link all data to electrical states within transistors inside the CPU • Identify and describe the common components within the CPU to better explain the steps the CPU goes through when processing data • Understand the need for memory within the CPU (using Von Neumann Architecture) to articulate where data and instructions are stored when the CPU carries out a process • Identify and describe the meaning of clock speed, cache size and the number of cores within a CPU, to recognise that the performance of a CPU can be measured, improved and how this impacts the overall performance of a computer system • Recognise the two types of computer system (embedded and general purpose) to identify computers used in real-world scenarios and why each exists (cost, mass production, function) **Strand 2** • Describe and use abstraction and decomposition when defining, and refining problems / creating solutions to identify how to break problems down into smaller, manageable parts with relevant detail, leading to more effective solutions • Recognise the purpose of shapes used within flowcharts to present algorithms to read solutions and predict outcomes and possible errors • Create algorithms using flowcharts and a high-level programming language (Python) to solve a set problem, correct a solution or improve it • Identify and use variables, constants, operators, inputs, outputs, common arithmetic operators, assignment operators and assignments in order determine what data should be input into a solution, what is done with the data and what the outcome should be • Understand the need for sequencing within algorithms to ensure steps take place in the correct order, and recognise / fix solutions when the order is incorrect

	• Identify and use variables, constants, operators, inputs, outputs, common arithmetic operators, assignment operators and assignments in order determine what data should be input into a solution, what is done with the data and what the outcome should be • Understand the need for sequencing within algorithms to ensure steps take place in the correct order, and recognise / fix solutions when the order is incorrect • Understand and use selection within algorithms to make decisions and determine the next step within a solution, to design solutions that can adapt to the data inputted • Understand and use iteration (count-controlled and condition-controlled) to repeat patterns within algorithms and design solutions efficiently • Identify the use of integer, real, Boolean, character/string and casting to choose or change to the correct data type needed for a specific scenario • Explore the use of imported modules to perform specific tasks within an algorithm such as random value generation, drawing or use of time	• Understand and use selection within algorithms to make decisions and determine the next step within a solution, to design solutions that can adapt to the data inputted • Understand and use iteration (count-controlled and condition-controlled) to repeat patterns within algorithms and design solutions efficiently • Identify the use of integer, real, Boolean, character/string and casting to choose or change to the correct data type needed for a specific scenario • Explore the use of imported modules to perform specific tasks within an algorithm such as random value generation, drawing or use of time
What pupils should already know (prior learning components)	**Strand 1** Within Key Stage 3 Computing, students will have been taught to • understand how numbers can be represented in binary, and be able to conduct simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal] (***Year 8 - Binary and Computer Logic Unit*****)** • understand how instructions are stored and executed within a computer system; understand how data of distinct types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits (***Year 7 – Getting Started*** **and** ***Year 8 - Binary and Computer Logic Unit***) • Understand the roles played by RAM, ROM and storage within a computer system *(**Year 8 Computer components**)* **Strand 2** Within Key Stage 3 Computing, students will have been taught to • Recognise the use of shapes within flowcharts and apply them to design a solution to a real-world problem *(**Year 9 – Algorithms**)* • Understand the meaning of abstraction and decomposition and apply them to break down or simplify problems *(**Year 9 – Algorithms**)* • Use a programming language to solve several problems, and use a range data types and operators to perform arithmetic and logical operations *(**Year 7 – Programming in Scratch, Year 8/9 – Python programming – Sequence**)* • Use logical operators and selection to determine decisions based on inputted data *(**Year 7 / 8– Introducing Spreadsheets / Advanced Spreadsheets, Year 8/9 – Python programming – Sequence**)* • Recognise the use of specific programming syntax (rules) that must be followed when using a specific language	**Strand 1** Within Key Stage 3 Computing, students will have been taught to • Understand the definition of CPU and the idea that it processes inputs to determine an output *(**Year 8 Computer components**)* • Understand that the CPU manages the performance of a computer, and its performance can be improved • Recognise that the CPU performs all operations in binary (***Year 7 - Binary and Computer Logic Unit***) Within KS4 Computer Science, students will have been taught to • Understand that all data used by computer systems must be converted into binary data (***within Year 10, Component 1.2***) **Strand 2** Within Key Stage 3 Computing, students will have been taught to • Recognise the use of shapes within flowcharts and apply them to design a solution to a real-world problem *(**Year 9 – Algorithms**)* • Understand the meaning of abstraction and decomposition and apply them to break down or simplify problems *(**Year 9 – Algorithms**)* • Use a programming language to solve several problems, and use a range data types and operators to perform arithmetic and logical operations *(**Year 7 – Programming in Scratch, Year 8/9 – Python programming – Sequence**)* • Use logical operators and selection to determine decisions based on inputted data *(**Year 7 / 8– Introducing Spreadsheets / Advanced Spreadsheets, Year 8/9 – Python programming – Sequence**)* • Recognise the use of specific programming syntax (rules) that must be followed when using a specific language Within GCSE Compute Science, students will have been taught to • Use a high-level programming language to design programs to solve a specific problem • Use specific data types to create variables • Use sequencing, selection and iteration in combination to determine outcomes
Transferrable knowledge (skills)	**Strand 1** • Being able to place units of measurement in order and convert between them to present the most appropriate unit when calculating a data capacity (mathematics) • The ability to articulate the link between binary data (0 and 1), transistors and electrical states • The use of a set of provided formulae to correctly perform file size calculations (mathematics) • Being able to convert between number systems and present decimal (denary numbers) in binary and hexadecimal form (and vice versa) as well as perform binary addition and shifts (all requiring mental arithmetic) • The ability to link the number of binary digits used to represent data and the impact on the possible number of integers, characters, colours, or samples that can be presented • Being able to identify factors that affect the file size and quality of image and sound files • Being able to link file size and content and its impact of network performance • Being able to compare RAM and ROM to recognise the role played by both within a computer system	**Strand 1** • Being able to define the term CPU and describe its purpose within a computer system to recognise its importance in any computer system • Understand and articulate the way a CPU functions through use of the FDE cycle to state how a CPU performs basic actions, and by extension, how a computer works • Identify the three performance aspects of a CPU that control its performance and the performance of a computer system in order to articulate how a computer can function more efficiently • Recognise the need for memory within a CPU to store data currently being used

	• Being able to compare storage types, their characteristics and determine which technology is best for a specific scenario **Strand 2** • The use of a text-based programming language and its specific syntax to perform sequencing, selection and iteration • Being able to articulate the need for programming constructs such as sequence, selection and iteration • Being able to recognise the four main data types within programming • Being able to use a combination of mathematical operators to perform arithmetic and logical operations within the text-based programming language which solve a specific problem • Being able to use exemplar code snippets and explain the outcome, as well as modify the code to perform actions needed for a specific purpose **General Skills** • The ability to search for information sources online and curate material based on relevance, factual content and needs of a specific purpose and audience • Being able to use multiple pieces of software (such as a web browser, presentation software, image editing software and a cloud computing system) in quick succession to create and refine design projects • The use of inference and articulation to obtain key knowledge from a topic and apply understanding when presenting findings	• Identify and describe key components within a CPU in order to explain the processes that take place when data is input, processed and output **Strand 2** • The use of a text-based programming language and its specific syntax to perform sequencing, selection and iteration • Being able to articulate the need for programming constructs such as sequence, selection and iteration • Being able to recognise the four main data types within programming • Being able to use a combination of mathematical operators to perform arithmetic and logical operations within the text-based programming language which solve a specific problem • Being able to use exemplar code snippets and explain the outcome, as well as modify the code to perform actions needed for a specific purpose **General Skills** • The ability to search for information sources online and curate material based on relevance, factual content and needs of a specific purpose and audience • Being able to use multiple pieces of software (such as a web browser, presentation software, image editing software and a cloud computing system) in quick succession to create and refine design projects • The use of inference and articulation to obtain key knowledge from a topic and apply understanding when presenting findings
Key vocabulary pupil will know and learn	**Strand 1** - Units of measurement (Bit Nibble, Byte, Kilobyte, Megabyte, Gigabyte, Terabyte, Petabyte), Binary (Base 2), Denary (Base 10), Hexadecimal (Base 16), Overflow Error, Carry Bits, Binary Shift, Character Set, ASCII, Unicode, Metadata, Pixel, Vector, Colour Depth, Resolution, RGB Values, Sample rate, Bit depth, Frequency, Compression, Lossy and Lossless **Strand 2** - Abstraction, Decomposition, pattern recognition, algorithm, Variables, constants, operators, inputs, outputs, selection, sequence, iteration, string, integer, float (decimal), Boolean operators, modules	**Strand 1** - CPU, Transistors, FDE Cycle, Clock speed, number of cores (single, dual and quad), cache memory, program counter, arithmetic logic unit, accumulator, memory address register, memory data register, overclocking **Strand 2** - Abstraction, Decomposition, pattern recognition, algorithm, Variables, constants, operators, inputs, outputs, selection, sequence, iteration, string, integer, float (decimal), Boolean operators, modules
Assessment activities	• Regular low stakes assessment (MCQ, exam questions with peer and self-assessment) • Data from low stakes assessment is used to target topics where misconceptions have been identified and to make corrective actions (MRI) • Use of online assessment (Forms, Kahoot, Blooket and Quizziz) • Content covered is used to form a progress assessment, in the form of an exam style paper with past paper questions from the exam board	
Resources available	Specification (Page 10, 11, 15 to 18) OCR GCSE (9-1) Computer Science Specification - J277 YouTube (Craig n' Dave) J277 videos (full specification) GCSE (J277): OCR Specification Order - YouTube BBC Bitesize (Data Representation) Units - Data representation - OCR - GCSE Computer Science Revision - OCR - BBC Bitesize What is an algorithm? - Algorithms - KS3 Computer Science Revision - BBC Bitesize BBC Bitesize (Programming techniques) The three basic programming constructs - Programming constructs - OCR - GCSE Computer Science Revision - OCR - BBC Bitesize KS4 NC information National Curriculum - Computing key stages 3 and 4 (publishing.service.gov.uk)	Specification (Page 6, 15 to 18) OCR GCSE (9-1) Computer Science Specification - J277 YouTube (Craig n' Dave) J277 videos (full specification) GCSE (J277): OCR Specification Order - YouTube BBC Bitesize (System Architecture) General purpose computers - Systems architecture - OCR - GCSE Computer Science Revision - OCR - BBC Bitesize KS4 NC information National Curriculum - Computing key stages 3 and 4 (publishing.service.gov.uk)
Notes **Why this topic is important…**	**Strand 1** - This topic builds on knowledge developed throughout key stage 3, such as units of measurement, binary and denary conversion and the concept of character sets and images and adds greater depth by exploring specific representations and how the amount of data the computer uses to represent data impacts on the quality as well as the overall size of files. This topic is important because it re-establishes the basic idea that computer systems use binary data to represent numbers, text, images, and sound to function in the ways we see every day. Understanding these concepts is fundamental in connecting data to the CPU (**Component 1.1**), performing mathematical calculations (binary, denary, hex, shifts, data capacity and file calculations) and develop an understanding of the need for memory and storage within a computer system (**Component 1.2**). By recognising capacity, students will then be able to determine appropriate storage technologies in the next topic. The logical sequencing students will develop for conversions will also help students to understand the need for	S**trand 1** - This units explores the central component; the CPU, and as how a computer's performance can be measured. It builds on the idea of Moore's Law, first introduced in **Unit 3 of Year 7**, and expands of the technical specifics explored in **Unit 3 of Year 8**, by delving deeper into each part of the CPU, the role they play and the process known as the FDE cycle, which is performed billions of times per second. By understanding this, students can grasp how a computer actually processes an instruction, and many of them very quickly.

	performing processes in specific order to achieve the correct result, itself a key factor in being able to access computational thinking and to create algorithms.Finally, the concepts of metadata and compression will tie into ethical considerations (**Component 1.6**) and network performance issues (**Component 1.4**) and as such form a base understanding of concepts in future topics. **Strand 2** - The term 'algorithm' has come to refer to any set of rules that precisely define a sequence of operations, such as making a cup of tea or cleaning your teeth. In the world of computing, an algorithm is a set of instructions that can be implemented as code to program a computer. Computational thinking is a logical, strategic approach to problem solving involving four cornerstones: decomposition, abstraction, pattern recognition and algorithm design to formulate an efficient and effective algorithm. These principles are key to being able to explore a problem and design an effective algorithm to solve it and are the cornerstones of the programming 'side' of Computer Science. Following on from **Programming in Scratch (Year 7 unit 5**), Algorithms and **Python Programming (In Year 8 and 9**), students will then apply a deepening knowledge to create several different algorithms, which will support them to develop logical thinking skills and support them accessing programming and algorithm challenges. Within programming and algorithms, students will often be faced with a specific problem that will demand they apply a range of key techniques independently. These skills will be tested within the programming project of GCSE Computer Science (**completed within Year 11**) and will support them in developing solutions for unseen problems they would be presented with if they were to work in the industry in the future. The unit also builds upon the use flow diagrams, first introduced in the Year 7 unit 5; students will learn how to use the functions of some standard flow diagram symbols and how to link them together to represent an algorithm. They will apply this learning to practical activities that will enable them to practise flow-diagram design and consolidate their comprehension of flow diagrams and how they relate to algorithms. It also links to unit 3 of Year 8 by identifying which components of the computer process data and the ways in which computers process data in **Year 10 Components 1.2 and 1.1**.	This unit builds upon the **Unit 3 of Year 7**, where students were introduced to the key components inside a standard computer. They will extend prior learning how by identifying each component's basic functionality and its impact upon performance. The unit also explores how technology has advanced by revisiting Moore's law and comparing the performance of computers over time. This content is core to knowing how a computer functions, itself a key aspect of Component 1 of GCSE Computer Science. By understanding how a CPU behaves and how it can be improved builds on topics of **Component 1.2 within Year 10** and supports students when moving into **Component 1.3 later in Year 11**. **Strand 2** - Students will continue to develop independent programming skills and build solutions in Python to a range of problems. This is critical to being able to program effectively. Programming skills are constantly developed and refined throughout all half terms or Year 10 and 11, sometimes linking directly to strand 1.