Subject:	GCSE Computer Science	Year	11	Ability	Mixed

Term / Date(s)	Components 2.1, 2.2, 2.3	Components 1.1 – 1.6, 2.1 – 2.5 (Revision and Examination)
Topic	Programming and Algorithms	Component 1 – Computer Systems Component 2 – Computational Thinking, Algorithms and Programming
Topic overview **Pupils will learn…**	Students will develop and implement algorithms to solve a set of computational problems, debug errors and refine solutions. Students will also explore a range of searching and sorting techniques.	**Component 1 -** Students will consider how computers function, the components they are made up of, how they communicate, fight security threats and impact the world and society. **Component 2 –** Students will explore how computers create solutions to problems, including the type of data they use, the ways in which data can be ordered, how to test and improved solutions, and the environments these solutions can be developed within.
Components	**Programming and Algorithms** • 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 • 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 the use of searching techniques (binary and linear) and sorting techniques (bubble, merge and insertion). • Identify key uses for and differences between each searching and sorting technique • Be able to carry out searching and sorting techniques with a data set and present the outcomes • Understand how to manipulate string data captured within a program • Explore the use of a programming language to open, read, write and close file handling • Identify the use of SQL to search for and return data • Identify, describe and use 1D and 2D arrays within programs	**Component 1 – All components as previously explored within Part A and B of Year 10, and Part A of Year 11 Computer Science Curriculum Plannings** • 1.1 System Architecture • 1.2 Memory and Storage • 1.3 Networks • 1.4 Network Security • 1.5 System Software • 1.6 Ethical, Environmental, Cultural and Legal Issues **Component 2 - All components as previously explored within Part A and B of Year 10, and Part A of Year 11 Computer Science Curriculum Plannings** • 2.1 Algorithms • 2.2 Programming Fundamentals • 2.3 Producing Robust Programs • 2.4 Boolean Logic • 2.5 Languages and Integrated Development Environments
What pupils should already know (Prior learning components)	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)***	**Component 1** Within GCSE Compute Science, students will have been taught to: • Identify the use of, and components of system architecture • Describe how system performance can be measured and impacted • Understand the differences and use of memory and storage • Identify how data (binary) is used to present numbers, text, images and sound

	• 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 • Identify the tools available within an IDE and how they help to make programming more efficient and user friendly	• Compare and contrast examples of storage technology for specific needs • Identify how networks are constructed and describe key network hardware • Describe how the performance of a network can be impacted by a range of factors • Draw diagrams of specific network types • Identify and describe a range of network threats and protective strategies • Explain the need for system software (operating systems) and their features • Explain the need for utility software (operating systems) and their features • Identify and discuss a range of legal, environmental and ethical impacts of technology on society and individuals • Discuss features of key legislation such as Copyright, Data Protection and Computer Misuse law **Component 2** Within GCSE Compute Science, students will have been taught to • Describe the meaning of decomposition, abstraction and algorithmic thinking, with real life examples • 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 • Create visual solutions to problems in the form of flowcharts / Use flowcharts to design solutions to problems with a programming language • Plan and test programs • Use modules, global and local variables, file handling and sub programs to design more efficient solutions • Identify searching and sorting techniques, perform them with data sets and compare them in terms of advantages and disadvantages • Understand the use of logic gates, truth tables and algebraic logic with examples • Identify the key features and differences between compilers, translators and interpreters • Understand key features within Integrated Development Environments
Transferrable knowledge (skills)	**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 **Programming and Algorithms** • 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 **Transferrable skills for Component 1 and 2 are identified in Part A and Part B of Y10 and Y11 Computer Science Curriculum Planning Documents**
Key vocabulary pupil will know and learn	Abstraction, Decomposition, pattern recognition, algorithm, Variables, constants, operators, inputs, outputs, selection, sequence, iteration, string, integer, float (decimal), Boolean operators, modules, Binary Search, Linear Search, Bubble Sort, Merge Sort, Insertion Sort, Comparisons, Structured Query Language, 1D array, 2D array, String Manipulation	**All key words as identified in Part A and Part B of Y10 and Y11 Computer Science Curriculum Planning Documents**
Assessment activities	• Regular low stakes assessment (MCQ, exam questions with peer and self-assessment) • Use of online assessment (Forms, Kahoot, Blooket and Quizziz) • Long written tasks with flash marking and extended responses	

	• Mock exam papers completed as part of the Year 11 Academy Mock Exam Timetable • Content covered is used to form a Progress In class examination assessment, in the form of an exam style paper with past paper questions from the exam board (October 2022)	
Resources available	Specification (Page 15 to 19) OCR GCSE (9-1) Computer Science Specification - J277 YouTube (Craig n' Dave) J277 videos (full specification) GCSE (J277): OCR Specification Order - YouTube BBC Bitesize Standard algorithms - Searching and sorting algorithms - OCR - GCSE Computer Science Revision - OCR - BBC Bitesize Data types - Data types and programming techniques - OCR - GCSE Computer Science Revision - OCR - BBC Bitesize Defensive design considerations - Producing robust programs - OCR - GCSE Computer Science Revision - OCR - BBC Bitesize KS4 NC information National Curriculum - Computing key stages 3 and 4 (publishing.service.gov.uk)	Specification (Component 1 - Page 6 - 14) (Component 2 – Page 15 – 21) OCR GCSE (9-1) Computer Science Specification - J277 YouTube (Craig n' Dave) J277 videos (full specification) GCSE (J277): OCR Specification Order - YouTube BBC Bitesize – Full OCR Section GCSE Computer Science - 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** - We use networks daily. We use them to read our emails, to message our friends and family and so on. They are a vital part of our use of technology and an increasingly import aspect in our lives. Understanding the structure of networks and how they operate can help use them effectively. This component builds on the role of key components established in **Year 10**, now extending into the ways that millions of devices can connect and send / use data. It links specifically to **1.1 system architecture and 1.2 memory**, whilst the encryption activities use algorithms explored in **Year 10** with **2.1 algorithms and 2.2 programming techniques**. It builds upon the **Year 8 cyber security** unit and leads onto the next Y11 unit; **1.4 network security**. Understanding networks is one of the key areas for growth in the computing industry. **Strand 2** - Students will use Boolean logic within programming and build Python representations that can act like logic gates. 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.	**This last component is revision and based upon:** • Individual student and cohort performance in each component • Revisit activities to key concepts and challenge areas within both component 1 and 2 • Consolidated and based upon previous work completed with the students' team files and evidence folder By revisiting each component in sequence, students are shown the wider links between them and can combine aspects of Component 2 to design solutions to problems of growing complexity.