Complete the RAG by giving a 1,2 or 3 on your confidence of this gained knowledge.

1 - Red - I am not confident of this knowledge.
2 - Amber - I have some knowledge in this area.
3 - Green - I am very confident in this area, I have this knowledge secured.

OCR GCSE Computer Science J277	
	RAG
2.1 – Algorithms	**1, 2, 3**
2.1.1 Computational thinking	
2.1.1 a Principles of computational thinking:	
2.1.1 a i Abstraction	
2.1.1 a ii Decomposition	
2.1.1 a iii Algorithmic thinking	
2.1.2 Designing, creating and refining algorithms	
2.1.2 a Identify the inputs, processes, and outputs for a problem	
2.1.2 b Structure diagrams	
2.1.2 c Create, interpret, correct, complete, and refine algorithms using:	
2.1.2 c i Pseudocode	
2.1.2 c ii Flowcharts	
2.1.2 c iii Reference language/high-level programming language	
2.1.2 d Identify syntax errors and suggest fixes	
2.1.2 e Identify logic errors and suggest fixes	
2.1.2 f Create and use trace tables to follow an algorithm	
2.1.2 g Refining algorithms	
2.1.3 Searching and sorting algorithms	
2.1.3 a Standard searching algorithms:	
2.1.3 a i Binary search	
2.1.3 a ii Linear search	
2.1.3 b Standard sorting algorithms:	
2.1.3 b i Bubble sort	
2.1.3 b ii Merge sort	
2.1.3 b iii Insertion sort	
2.2 – Programming fundamentals	
2.2.1 Programming fundamentals	
2.2.1 a The use of variables	
2.2.1 b The use of constants	
2.2.1 c The use of operators	
2.2.1 d The use of inputs	
2.2.1 e The use of outputs	
2.2.1 f The use of assignments	

2.2.1 g The use of the three basic programming constructs used to control the flow of a program:	
2.2.1 g i Sequence	
2.2.1 g ii Selection	
2.2.1 g iii Iteration	
2.2.1 g iv count controlled i.e. for loop	
2.2.1 g v condition controlled i.e. while loop, repeat until	
2.2.1 h The common arithmetic operators	
2.2.1 j The common Boolean operators AND, OR and NOT	
2.2.1 k Comparison operators Arithmetic operators	
2.2.1 k i == Equal to + Addition	
2.2.1 k ii != Not equal to – Subtraction	
2.2.1 k iii < Less than * Multiplication	
2.2.1 k iv <= Less than or equal to / Division	
2.2.1 k v > Greater than MOD Modulus	
2.2.1 k vi >= Greater than or equal to DIV Quotient	
2.2.1 k vii ^ Exponentiation (to the power)	
2.2.2 Data types	
2.2.2 a The use of data types:	
2.2.2 a i Integer	
2.2.2 a ii Real	
2.2.2 a iii Boolean	
2.2.2 a iv Character and string	
2.2.2 a v Casting	
2.2.3 Additional programming techniques	
2.2.3 a The use of basic string manipulation	
2.2.3 a i Concatenating	
2.2.3 a ii Slicing	
2.2.3 b The use of basic file handling operations:	
2.2.3 b i Open	
2.2.3 b ii Read	
2.2.3 b iii Write	
2.2.3 b iv Close	
2.2.3 c The use of records to store data	
2.2.3 d The use of SQL to search for data	
2.2.3 e The use of arrays as fixed length static structures when solving problems	
2.2.3 f The use of 2D arrays as fixed length static structures when solving problems	
2.2.3 g How to use sub programs (procedures) to produce structured code	
2.2.3 h How to use sub programs (functions) to produce structured code	
2.2.3 j Random number generation	
2.2.3 k SQL commands:	
2.2.3 k i SELECT	
2.2.3 k ii FROM	
2.2.3 k iii WHERE	
2.3 – Producing robust programs	
2.3.1 Defensive design	

2.3.1 a Defensive design considerations:	
2.3.1 a i Anticipating misuse and invalid data	
2.3.1 a ii Authentication to confirm the identity of a user	
2.3.1 b Input validation	
2.3.1 b i Length check	
2.3.1 b ii Range check	
2.3.1 b iii Presence check	
2.3.1 c Practical experience of designing input validation and simple authentication (e.g. username and password)	
2.3.1 d Maintainability:	
2.3.1 d i Use of sub programs	
2.3.1 d ii Naming conventions	
2.3.1 d iii Indentation	
2.3.1 d iv Commenting	
2.3.2 Testing	
2.3.2 a The purpose of testing	
2.3.2 b Types of testing:	
2.3.2 b i Iterative (module/unit tests)	
2.3.2 b ii Final/terminal	
2.3.2 c Syntax errors as errors which break the grammatical rules of the programming language and stop it from being run/translated	
2.3.2 d Logic errors as errors which produce unexpected output	
2.3.2 e Selecting and using suitable test data:	
2.3.2 e i Normal test data as data which should be accepted by a program without causing errors	
2.3.2 e ii Boundary test data as data of the correct type which is on the very edge of being valid	
2.3.2 e iii Invalid test data as data of the correct type but outside accepted validation limit	
2.3.2 e iv Erroneous test data as data of the incorrect type which should be rejected by a computer system	
2.3.2 f Ability to create/complete a test plan	
2.4 – Boolean logic	
2.4.1 Boolean logic	
2.4.1 a Simple logic diagrams using the operators AND (conjunction)	
2.4.1 b Simple logic diagrams using the operators OR (disjuntction)	
2.4.1 c Simple logic diagrams using the operators NOT (negation)	
2.4.1 d Truth tables	
2.4.1 e Combining Boolean operators using AND, OR and NOT	
2.4.1 f Applying logical operators in truth tables to solve problems	
2.5 – Programming languages and Integrated Development Environments	
2.5.1 Languages	
2.5.1 a Characteristics and purpose of different levels of programming language:	
2.5.1 a i High-level languages	
2.5.1 a ii Low-level languages	

2.5.1 b The purpose of translators	
2.5.1 b i The characteristics of a compiler	
2.5.1 b ii The characteristics of an interpreter	
2.5.2 The Integrated Development Environment (IDE)	
2.5.2 a Common tools and facilities available in an Integrated Development Environment (IDE):	
2.5.2 a i Editors	
2.5.2 a ii Error diagnostics	
2.5.2 a iii Run-time environment	