OCR GCSE Component 02 Personalised Learning Checklist

Component 02			R	A	G
2.1 Algorithms	**2.1.1 Computational Thinking**	I can explain and apply the following terms to solve problems: Abstraction, decomposition and algorithmic thinking			
	2.1.2 Designing, Creating and Refining algorithms	I can identify the inputs, processes, and outputs for a problem			
		I can complete and create structure diagrams to show a problem structure and subsections of the problem			
		I can create, interpret, correct, complete, and refine algorithms using: Pseudocode, Flowcharts, Reference Language/High-Level Language			
		I can identify common errors and suggest fixes			
		I can create and use trace tables to follow an algorithm			
	2.1.3 Searching and Sorting algorithms	I can explain the main steps of standard searching algorithms: Binary Search and Linear Search			
		I can explain the main steps of standard sorting algorithms: Bubble Sort, Merge Sort, Insertion Sort			
		I can apply the searching or sorting algorithm to a data set and show the steps involved			
		I can identify the algorithm if given the code or pseudocode for it			
2.2 Programming Fundamentals	**2.2.1 Programming Fundamentals**	I can declare variables and constants with meaningful identifier names			
		I can use the three basic programming constructs to control the flow of a program: Sequence, Selection, Iteration			
		I can use common arithmetic, comparison and Boolean operators			
	2.2.2 Data Types	I can identify when to use the following data types: Integer, Real, Boolean, Character and String			
		I can use casting to change the data types so that the program can handle data effectively			
	2.2.3 Additional Programming Techniques	I can use Boolean operators in selection statements			
		I can use nested selection and nested iteration			
		I can use definite (count-controlled) and indefinite (condition-controlled) iteration			
		I can use basic string manipulation including concatenation, slicing and formatting (upper, lower etc.)			
		I can use basic file handling operations including: Open, Read, Write, Close			
		I can create and use an array/list in a programming language			
		I can create and use 2D arrays to represent database tables of fields and records to store data			
		I can create a sub programs (functions and procedures) to produce structured code			
		I can use local and global variables/constants within subprograms appropriately			
		I can use parameters to pass data, and arrays to pass and return data within a program			
		I can use random number generation in a programming language			
		I can use SQL to search for data using the SQL commands: SELECT, FROM, WHERE			

2.3 Producing Robust Programs	2.3.1 Defensive Design	I can understand the issues a programmer should consider ensuring that a program caters for all likely input values			
		I can construct a working authentication system			
		I can anticipate misuse of a program and explain/construct methods to prevent the misuse			
		I can construct my programs to deal with invalid input from a user by including input validation/sanitisation			
		I can explain the methods of maintainability of a program through the use of sub programs, naming conventions, indentation and commenting			
	2.3.2 Testing	I can explain the importance of testing, including identifying and describing the types of errors that can be found during the process			
		I can explain the difference between testing modules of a program during development (iterative) and testing the program at the end of production (final/terminal)			
		I can identify, select and use suitable test data including: Normal/Valid, Boundary/Extreme, Abnormal/Invalid/Erroneous			
		I can create/complete a test plan			
		I can refine an algorithm based on the testing			
2.4 Boolean Logic	2.4 Boolean Logic	I can create, complete or edit simple logic diagrams using the operators: AND, OR, NOT			
		I can create, complete or edit logic diagrams of more than one gate, using the operators: AND, OR, NOT			
		I can create, complete or edit truth tables using the operators: AND, OR, NOT			
		I can apply logical operators in truth tables to solve problems			
2.5 Programming Languages and Integrated Development Environments	2.5.1 Languages	I can explain the characteristics and purpose of different levels of programming language: high-level and low-level			
		I can explain the purpose of and need for translators			
		I can describe the characteristics of, differences between, benefits and drawbacks of: a compiler and interpreter			
	2.5.2 The Integrated Development Environment (IDE)	I can describe the common tools and facilities available in an IDE: editors, error diagnostics, run-time environment, translators			
		I can describe how the tools and facilities can be used to develop a program			