UNIT: PROBLEM SOLVING

Mark on how confident you are with each topic and what you need to do, use this as a checklist for your revision notes and then to keep track of the topics you have revised. Tick and date once you have completed the tasks.

Topic	How confident am I?			What do I need to do to improve my understanding – create a mind map, write revision note cards, use online tests, attempt some practice question papers, read and highlight my notes…. OR Who do I need to seek support from to help me progress	I have made my revision notes on this topic	I have revised this topic	I am confident on this topic	Evidence
	 I could teach	 I would need to look at my notes	 I can not remember					
Overview from syllabus 1.1 Algorithms								
Content from syllabus				Specific skills or knowledge to be assess				
1.1.1 understand what an algorithm is, what algorithms are used for and be able to interpret algorithms (flowcharts, pseudocode, written descriptions, program code)								
1.1.2 understand how to create an algorithm to solve a particular problem, making use of programming constructs (sequence, selection, iteration) and using appropriate conventions (flowchart, pseudo-code, written description, draft program code)								
1.1.3 understand the purpose								

of a given algorithm and how an algorithm works								
1.1.4 understand how to determine the correct output of an algorithm for a given set of data								
1.1.5 understand how to identify and correct errors in algorithms								
1.1.6 understand how to code an algorithm in a high-level language								
1.1.7 understand how the choice of algorithm is influenced by the data structures and data values that need to be manipulated								
1.1.8 understand how standard algorithms (bubble sort, merge sort, linear search, binary search) work								
1.1.9 be able to evaluate the fitness for purpose of algorithms in meeting specified requirements efficiently using logical reasoning and test data								
1.2 Decomposition and abstraction								
1.2.1 be able to analyse a problem, investigate requirements (inputs, outputs, processing, initialisation) and								

design solutions								
1.2.2 be able to decompose a problem into smaller sub-problems								
1.2.3 understand how abstraction can be used effectively to model aspects of the real world								
1.2.4 be able to program abstractions of real-world examples								

Review of previous examinations: