YEAR 1

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
1	**1.1.3** a) How different input output and storage devices can be applied to the solution of different problems b) The uses of magnetic, flash and optical storage devices c) RAM and ROM d) Virtual storage	**2.1.2** Thinking ahead (introduced)	Simple problem solving tasks
2			
3		**2.2.1** Introduction to the e) Use of an IDE to develop/debug a program	Procedural/Imperative language IDE of Centre's choice
4	**1.2.3** b) Writing and following algorithms	**2.2.1** b) Programming constructs: sequence, iteration, branching	Programming exercises involving sequence
5	**1.1.1** a) The Arithmetic and Logic Unit; ALU, Control Unit and Registers (Program Counter; PC, Accumulator; ACC, Memory Address Register; MAR, Memory Data Register; MDR, Current Instruction Register; CIR): How this relates to assembly language programs b) The Fetch-Decode-Execute Cycle	**2.1.4** Thinking logically (introduced) **2.2.1** a) Programming constructs: sequence, iteration, **branching**	Programming exercises involving branching (IF, nested IF, SELECT/CASE)
6			
7	**1.2.1** a) The function and purpose of operating systems e) Distributed, Embedded, Multi-tasking, Multi-user and Real Time operating systems f) BIOS g) Device drivers	**2.2.1** a) Programming constructs: sequence, **iteration**, branching	Programming exercises involving iteration (FOR, WHILE, REPEAT)
8			
9			

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
10	**1.2.2** a) The nature of applications b) Utilities c) Open source vs Closed source		
11	**1.2.4** a) Procedural languages	**2.2.1** b) Recursion, how it can be used and compares to an iterative approach	Programming exercises demonstrating recursion (eg factorial)
12 13 14 15	**1.4.1** a) Represent positive integers in binary b) Use of Sign and Magnitude and Two's Complement to represent negative numbers in binary c) Addition and subtraction of binary integers d) Represent positive integers in hexadecimal e) Representation and normalisation of floating point numbers in binary f) Floating point arithmetic, positive and negative numbers, addition and subtraction g) Bitwise manipulation and masks: shifts, combining with AND, OR, and XOR h) How character sets (ASCII and UNICODE) are used to represent text	**2.1.3** Thinking Procedurally (introduced) **2.2.1** c) Global and local variables. d) Modularity, functions and procedures, parameter passing by value and by reference **2.2.2** a) Features that make a problem solvable by computational methods b) Problem Recognition c) Problem Decomposition d) Use of divide and conquer	Programming exercises involving functions, procedures and parameters

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
16 17	**1.4.2** a) Arrays (of up to 2 dimensions)	**1.4.2** a) Arrays (of up to 2 dimensions)	Programming exercises involving arrays
18 19 20	**1.4.2** b) The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table c) How to create, traverse, add data to and remove data from the data structures mentioned above	**1.2.3** a) Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development b) The relative merits and drawbacks of different methodologies and when they might be used	Programming exercises including the algorithms for the main data structures
21 22 23	**1.4.3** a) Define problems using Boolean logic b) Use the following rules to derive or simplify statements in Boolean algebra: De Morgan's Laws, distribution, association, commutation, double negation	**2.3.1** a) Analysis and design of algorithms for a given situation b) The suitability of different algorithms for a given task and data set, in terms of execution time and space c) Algorithms for the main data structures, (Stacks, queues, trees, linked lists, depth-first (post-order) and breadth-first traversal of trees) d) Standard algorithms (Bubble sort, insertion sort, merge sort, quick sort, Dijkstra's shortest path algorithm, A* algorithm, binary search and linear search)	

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
24	**1.2.4** b) Assembly language (including following and writing simple programs with the Little Man Computer instruction set) c) Modes of addressing memory (immediate, direct, indirect and indexed)	**1.2.4** b) Assembly language (including following and writing simple programs with the Little Man Computer instruction set)	Assembly language programming exercises
25			
26	**1.3.1** a) Lossy vs Lossless compression b) Run Length Encoding and dictionary coding for lossless compression c) Symmetric and asymmetric encryption d) Different uses of hashing	**1.3.2** a) Relational database, flat file, primary key, foreign key, secondary key, normalisation and indexing b) Normalisation to 3NF c) SQL - Interpret and modify (list of key words) d) Referential Integrity e) Transaction processing, ACID (Atomicity, Consistency, Isolation, Durability), record locking and redundancy	Practical exercises using appropriate DBMS
27			

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
28	**1.3.3**		
29	a) The TCP/IP Stack b) Protocol layering c) LANs and WANs d) Packet and circuit switching e) Protocols f) Client-server and Peer to peer		
30	**1.3.4**	**1.3.4**	Practical HTML, CSS, JavaScript exercises
31	a) HTML, CSS and JavaScript	a) HTML, CSS and JavaScript	
32	b) Search engine indexing c) PageRank Algorithm d) Server and client side processing	b) Search engine indexing c) PageRank Algorithm d) Server and client side processing	

YEAR 2

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
1	**1.5.1** a) Data Protection Act b) Computer Misuse Act c) Copyright and Patents Act d) Regulation of Investigatory Powers Act	**1.5.2** These include but are not limited to: a) Computers in the workforce b) Automated decision making c) Artificial intelligence d) Environmental effects e) Censorship and the Internet	
2			
3	**1.2.4** d) Object-oriented languages (using Java/C++ style pseudocode) with an understanding of classes, objects, methods, attributes, inheritance, encapsulation and polymorphism	**2.1.1** Thinking abstractly (introduced) **1.2.4** d) Object-oriented languages (using Java/C++ style pseudocode) with an understanding of classes, objects, methods, attributes, inheritance, encapsulation and polymorphism	Practical OO pseudocode exercises
4			
5			
6			

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
7	**1.1.2** a) The differences between and uses of CISC and RISC processors b) GPUs and their uses (including those not related to graphics) c) Multicore and Parallel systems **1.1.1** c) The use of pipelining in a processor to improve efficiency	**2.2.2** e) Use of abstraction f) Candidates should apply their knowledge of • backtracking • data mining • heuristics • performance modelling • pipelining • visualisation to solving problems **2.1.5** Thinking Concurrently (introduction)	Programming exercises complex enough to demonstrate and utilise computational methods
8			
9			
10	**1.2.1** b) Memory Management (paging, segmentation and virtual memory)		
11			

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
12	c) Interrupts d) Scheduling: Round Robin, First come first served, Multi-level feedback queues, shortest job first and shortest remaining time h) Virtual Machines		
13 14 15	**1.2.2** d) Translators: Interpreters, compilers and assemblers e) Stages of compilation (Lexical Analysis, Syntax Analysis, Code Generation and Optimisation) f) Linkers and loaders		
16 17 18	**3.1** Analysis **3.1.1** Problem identification **3.1.2** Stakeholders **3.1.3** Research the problem **3.1.4** Specify the proposed solution		

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
19			
20	**3.2** Design **3.2.1** Decompose the problem **3.2.2** Describe the solution **3.3.2** Describe the approach to testing		
21			
22			
23			
24	**3.3** Developing (and testing) the solution **3.3.1** Iterative development process **3.3.2** Development Testing **3.3.3** Post development testing		
25			
26			
27			
28			

Week Number	50% of available delivery time	50% of available delivery time	Suggested problem solving/programming tasks to support Unit 2 delivery
29	**3.4** Evaluation **3.4.1** Success of solution **3.4.2** Describe the final product **3.4.3** Maintenance and development.		
30