Subject :	Computer Science		Year Group:	12

	September to October	October - Christmas	January - February	February -Easter	February -Easter
Scheme title	Programming Concepts	Programming Concepts	Abstraction, Finite State Machines	AS Preliminary Material (prep for mock	AS Preliminary Material (prep for mock
Purpose of scheme	SOW being updated	SOW being updated	SOW being updated	SOW being updated	SOW being updated
Knowledge in sequence	4.1.1.1 Data types -Data Type Understand the concept of a data type. Understand and use the following appropri-ately: • integer • real/float • Boolean • character • string • date/time • pointer/reference • records (or equivalent) • arrays (or equivalent).Use, 4.1.1.2 Programming concepts understand and know how the follow-ing statement types can be combined in programs: • variable declaration • constant declaration • assignment • iteration • selection • subroutine (procedure/functionUse definite and indefinite iteration, includ-ing indefinite iteration with the condition(s) at the start or the end of the iterative struc-ture. A theoretical understanding of condi-tion(s) at either end of an iterative structure is required, regardless of whether they are supported by the language being used. Use nested selection and nested it-eration structures. Use meaningful identifier names and know why it is important to use them. 4.1.1.3 Arithmetic operations in a programming languageBe familiar with and be able to use: • addition • subtraction • multiplication • real/float division • integer division, including remainders • exponentiation • rounding • truncation. 4.1.1.4 Relational operations in a programming language Content Additional information Be familiar with and be able to use: • equal to • not equal to • less than • greater than • less than or equal to • greater than or equal to. 4.1.1.5 Boolean operations in a programming languageBe familiar with and be able to use: • NOT • AND • OR • XOR. 4.1.1.6 Constants and variables in a programming language Content Additional information Be able to explain the differences between a variable and a constant. Be able to explain the advantages of using named constants. 4.1.1.7 String-handling operations in a programming language Content Additional information Be familiar with and be able to use: • length • position • substring • concatenation • character → character code • character code → character • string conversion operations. Expected string conversion operations: • string to integer • string to float • integer to string • float to string • date/time to string • string to date/time. 4.1.1.8 Random number generation in a programming languageBe familiar with, and be able to use, random number generation. 4.1.1.9 Exception handlingBe familiar with the concept of ex-ception handling. Know how to use exception handling in a programming language with which stu-dents are familiar. 4.1.1.10 Subroutines (procedures/functions)Be familiar with subroutines and their uses. Know that a subroutine is a named 'out of line' block of code that may be executed (called) by simply writing its name in a pro-gram statement. Be able to explain the advantages of using subroutines in programs. 4.1.1.11 Parameters of subroutinesBe able to describe the use of parame-ters to pass data within programsBe able to use subroutines with interfaces. 4.1.1.12 Returning a value/values from a subroutineBe able to use subroutines that return val-ues to the calling routine. 4.1.1.13 Local variables in subroutinesKnow that subroutines may declare their own variables, called local variables, and that local variables: • exist only while the subrou-tine is executing • are accessible only within the subrou-tine. Be able to use local variables and explain why it is good practice to do so. 4.1.1.14 Global variables in a programming languageBe a+B1ble to contrast local variables with global variables. 4.1.1.15 Role of stack frames in subroutine Be able to explain how a stack frame is used with subroutine calls to store: • return addresses • parameters • local variables. 4.1.1.16 Recursive techniquesBe familiar with the use of recursive tech-niques in programming languages (general and base cases and the mechanism for im-plementation).	SOW being updated	SOW being updated	SOW being updated	SOW being updated
Skills	SOW being updated	SOW being updated	SOW being updated	SOW being updated	SOW being updated
Key Words	Integer, real/float, Boolean, character, string, date/time, pointer/reference, records (or equivalent), arrays (or equivalent). variable declaration, constant declaration, assignment, iteration, selection, subroutine (procedure/function). Exception Handling, Global Variable, Local Variable. Gate, Not, And, Or, XOR, Truth Table, Boolean, De Morgan, Boolean Algebra. Decimal, Binary, Hexadecimal, Floating Point, 2s Complement, Mantissa, Exponent. Bit, Byte, Nybble, Absolute Error, Relative Error, Range, Precision, Normalisation, Underflow, Overflow.	Data Structure, Array, Dimension, Fields, Records, Files. Character, ASCII, Unicode, Parity Bits, Majority Voting, Checksum, Check Digit. Analogue, Digital, Resolution, Colour Depth, Vector Graphics, Sample Size, Sample Resolution, Nyquist's Theorem. MIDI. Lossy, Lossless, RLE, Encryption, Vernam Cipher, Caesar Cipher, Plaintext, Ciphertext.	Abstraction, Processor, main memory, address bus, data bus, control bus, I/O controllers., Von Neumann, Harvard. Arithmetic logic unit, control unit, clock, general-purpose registers, program counter, current instruction register, memory address register, memory buffer register, status register, Fetch Execute Cycle, Instruction Set, Opcode, Operand, Addressing Mode, Interrupt, multiple cores, cache memory, clock speed, word length, address bus width, data bus width.	Hardware, Software, System Software, Application Software. operating systems, utility programs, libraries, translators (compiler, assembler, interpreter). Machine Code, Assembly Language, Imperative High Level Language. Barcode, RFID, Hard Disk, Optical Disk, Solid State Disk	Event Driven Programming, Moral, Ethical, Legal, Cultural. baud rate, bit rate, bandwidth, latency, protocol, Star, Bus, Peer to Peer, Client-Server, WiFi, CSMA/CA, RTS/CTS, SSID.
End Point	SOW being updated	SOW being updated	SOW being updated	SOW being updated	SOW being updated
Assessment method	After each topic, students complete a mini assessment. This may be completed as part of home learning and sometimes completed in class under test conditions. This is then teacher marked and recorded on the central tracking spreadsheet to inform progress and intervention. Students complete full A level assessments where possible in line with the AQA specification at progress points in the year in line with the school calendar. Assessments are cumulative and grade boundaries reflect actual A Level Computer Science grade boundaries.	SOW being updated	SOW being updated	SOW being updated	SOW being updated