Subject: Computer Science — Year group: 11

	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Content *Declarative Knowledge – 'Know What'*	Programming Arrays, Procedures and functions, Records and files	Logic and languages Logic diagrams, Truth tables, Defensive design, Errors and testing, Translators and facilities of languages,	Data representation Units, Numbers, Characters, Images, Sound, Compression	Exam Prep Revision techniques, past papers for practise and confident building	Exams Prep	Exam
Skills *Procedural Knowledge – 'Know How'*	• write pseudocode solutions to simple problems involving sequence, selection and iteration • use nested selection and iteration statements • use Boolean operations NOT, AND and OR within conditions for iterative and selection structures • use basic string manipulation functions in	• Recognise standard symbols used to represent NOT, AND OR, NAND, NOR and XOR logic gates • Draw truth tables for the above logic gates • Describe some simple validation checks that can be applied to data • Select test data that covers normal (typical), boundary (extreme) and erroneous data • Complete a trace table to trace through a simple algorithm • Give examples of high-level and low-level languages • Give advantages of high-level languages	• Explain why all data needs to be converted to binary before the computer can process it • Convert positive denary whole numbers (0-255) into 8-bit binary numbers and vice versa • Convert between binary and hexadecimal • Explain the use of binary codes to represent characters • Understand the term 'character set' • Explain the relationship between the number of bits per character and the number of characters which can be represented	• List some privacy issues in relation to a given scenario • Choose from a given list, which Act is relevant to a particular scenario • List one attribute and advantage of open source software and proprietary software • Describe some ethical, legal, cultural and/or environmental issues in relation to a given scenario • Describe some privacy issues in	• identify and use variable types integer, real, Boolean, character and string • identify variables and constants in a program • use meaningful identifier names and know why it is important to use them • use arithmetic operations including mod and div • use Boolean operators in pseudocode solutions • show the results of basic string	

	pseudocode solutions • give examples of data structures: arrays and records • use one-dimensional arrays in the design of solutions to simple problems • write simple functions and procedures using parameters • read from and write to a text file • explain what is meant by a data structure and why these are used • use two-dimensional arrays in the design of solutions to simple problems • explain why it is good practice to use local variables	over low-level languages • Explain the differences between a compiler, interpreter and assembler • Recognise a logic gate from its truth table • Draw a logic circuit to solve a given problem • Detect and correct errors in simple algorithms • Use a trace table to find errors or determine the purpose of an algorithm • Be able to justify the choice of test data • Give examples and reasons of when it might be appropriate to use a low-level language • Give examples of when it would be appropriate to use a compiler and interpreter • Draw a logic circuit to implement a • given written logic statement • Write more complex authentication routines • Write robust programs that apply checks to	• Explain the representation of an image as a series of pixels represented in binary • Explain how sound can be sampled and stored in digital form • Perform a binary shift • Explain the need for compression • Add two binary integers and explain overflow errors • Explain why hexadecimal numbers are used to represent binary data • Discuss the effect of colour depth and resolution on the size of an image file • Explain how sampling intervals and other considerations affect the size of a sound file • Explain the effects of a binary shift • Explain the purpose of a check digit • Explain the effect of different types of compression • Explain how instructions are coded as bit patterns	relation to a given scenario • Describe the differences between open source and proprietary software and give advantages of each: • List the clauses of the Data Protection Act and Computer Misuse Act and give examples of situations in which they are relevant • Evaluate the impact of and issues related to the use of computers in society •	manipulation functions • use random number generation • follow through pseudocode solutions to simple problems involving sequence, selection and iteration • explain why functions and procedures are used in creating solutions to problems • use simple functions and procedures that return values to the calling program	

		data entered by the user	• Explain how sampling intervals affect quality of the playback of a sound file • Explain how the computer distinguishes between instructions and data • Calculate a check digit			
Vocabulary	Subroutine, procedure, function, parameter, return value, built-in function, scope, global variable, local variable.	Binary, logic gate, NOT, AND, OR, NAND, NOR, XOR, truth table, logic circuit, logic statement compiler, interpreter, assembler, high level language, low level language, assembly language, source code, object code, bytecode, machine code, machine independence. validation, verification, authentication, syntax errors, logic errors, runtime errors, trace table, dry run, valid data, invalid data, boundary data.	Bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte, denary, overflow, hexadecimal, character set, ASCII, Unicode, character set, check digit, shift, metadata, pixel, colour depth, resolution, sound sampling, playback, lossy, lossless, compression.			
Assessment	End of topic Assessments, Presenting your understanding about a topic to the class, Q & A, Self-evaluation of topics	End of topic Assessments, Presenting your understanding about a topic to the class, Q & A, Self-evaluation of topics	End of topic Assessments, Presenting your understanding about a topic to the class, Q & A, Self-evaluation of topics	End of topic Assessments, Presenting your understanding about a topic to the class, Q & A, Self-evaluation of topics	End of topic Assessments, Presenting your understanding about a topic to the class, Q & A, Self-evaluation of topics	
Exams	<u>Written Paper 1: Computer Systems</u> This component will introduce learners to the Central Processing Unit (CPU), computer memory and storage, wired and wireless networks, network topologies, system security and system software.					

	It is expected that learners will become familiar with the impact of Computer Science in a global context through the study of the ethical, legal, cultural and environmental concerns associated with Computer Science. <u>Written Paper 2: Computational thinking, algorithms and Programming</u> This component incorporates and builds on the knowledge and understanding gained in Component 01, encouraging learners to apply this knowledge and understanding using computational thinking. Learners will be introduced to algorithms and programming, learning about programming techniques, how to produce robust programs, computational logic, translators and facilities of computing languages and data representation.