Computing Curriculum – KS3 Overview

Computing	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Y7	**Academy Network Introduction.** Be able to log onto the academy network and Teams. Create a secure password. Be able to access my documents on the network and create folders to store my files. **Introduction to Programming.** Creating programs in Scratch to create interactive animations.	**Spreadsheets.** Organising information in tables using rows and columns. Formatting tables using borders and shading. Using formulae to calculate totals. **Databases.** Using a database to store information in records and fields. Create and format forms to view and edit records and create new records.	**Algorithms.** Reading and following instructions to complete a task. Writing instructions to complete a task. **E-Safety.** Using online services and social media safely. Keeping accounts secure and safe messaging.	**Programming.** Develop an interactive game using Scratch. Writing code to control the main character and interact with other characters. **Spreadsheets.** Creating bar and pie charts from information stored in a table. Using functions to find the minimum, maximum and average values.	**Databases.** Searching a given database for records that match specific criteria. **Web Technologies.** Creating text based webpages using HTML.	**Algorithms.** Following instructions from a flowchart with decisions. Creating flowcharts. **Data Representation.** Know what units of data are commonly used. Know that computers store numbers in binary and be able to convert between binary numbers to decimal numbers.

Computing	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Y8	**Data Representation.** How images are stored as a bitmap in a computer. Black and white images and grey scale images. **Spreadsheets.** Creating charts from data tables. Creating and labelling line charts from a data table.	**Programming.** Scratch programming. Using Pen tools to create an 'Etch-a-Sketch' program and a Paint Program. **Databases.** Creating a data table to store information. Choosing the correct data type for each field and identifying the key field.	**Algorithms.** Following instructions given in a flowchart. Writing algorithms to solve problems using flowcharts using decisions. **Computer Hardware.** Identify key piece of computer hardware and know their purpose. Be able to categorise devices as input, output and storage devices.	**Programming.** Introduction to Python. Creating programs using Python and Turtle to draw shapes. **Spreadsheets.** Organising information into formatted tables. Using functions to summarise data in rows and columns.	**Databases.** Using queries to search a database and mail merge information from a database into a Word document. **Web Technologies.** Creating text based webpages using HTML.	**Programming.** Using data in Python. Inputs and outputs and variable arithmetic. **Data Representation.** Binary arithmetic and data conversion between different data units.
Y9	**Programming.** Using IF commands to make decisions in programs and using loops to repeat sections of code. **Spreadsheets.** Review of bar and pie charts. Creating and formatting Scatter graphs to show data given in a table.	**Data Representation.** Use AND, OR and NOT gates to create logic diagrams. **Databases.** Create data tables by importing information from a given data file.	**Programming.** Encrypting and decrypting messages using Caesar's Cipher. Creating a program in Python to automatically encrypt messages. **Web Technologies.** Creating web pages with images and embedded objects using HTML.	**Algorithms.** Use the Linear Search algorithm to find information in a list. **Spreadsheets.** Use conditional formatting in a table to highlight cell depending on the value stored.	**Databases.** Create databases with more than one table. Use parameter queries to be able to search a database. **Algorithms.** Use the Bubble sort algorithm to sort data into order.	**Programming.** Creating subroutines to draw shapes and create Spiral pattern in Python. **Data Representation.** How images are stored as colour bitmaps by a computer.

Computing / ICT Curriculum – KS4 and KS5 Overview

Computing	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Y10 Computer Science	Problem solving using abstraction and decomposition. Programming with Python. Using variables, data types, iteration and decisions. Creating functions to find area of different shapes. How computer use ASCII and Unicode to store character sets.	Data Representation. Bitmap Images and Audio. Compression. RLE and Huffman Encoding. Data Structures. Using strings and lists in Python.	Search and Sort Algorithms. Databases using SQL. Primary and Secondary Storage	The CPU. Characteristics that affect CPU performance. Mock Exam Revision Programming Project preparation	Programming Project Task. Generating Strong Passwords and Calculating the Strength of a Password	Programming Project Task. Low level languages. Machine Code and Assembly Language.
Y11 Computer Science	How a CPU works using the Fetch-Decode-Execute cycle. Embedded Systems. Networks. Network types and topologies. Protocols and the 4 Layer TCP/IP model.	Network security. Types of security testing and social engineering. Paper 1 Mock Exam and Review.	Legal, Ethic and Environmental Issues. Paper 2 Mock Exam and Review Revision of: Algorithms Data Representation Data Structures	Revision of: Search and Sort algorithms Huffman Encoding CPU Architecture Legal, Ethical and Social issues Revision for final exams	Revision and final preparation for exams.	

Computing/ICT	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Y11 OCR Cambridge National IT	**RO70** Using Augmented Reality to present information **R050** IT in the Digital World. Digital communications	**RO70** Using Augmented Reality to present information **R050** IT in the Digital World. Digital communications individuals and/or organisations	**R050** IT in the Digital World. Human Computer Interface (HCI) in everyday life Cyber-security and legislation: The impacts of a cyber-security attack on individuals and/or organisations	**R050** IT in the digital world Revision and exam technique	**R050** IT in the digital world Revision and exam technique	Revision and final preparation for exams.
Y12 Computer Science	Fundamentals of Programming. Variables and data types. Exception handling. Subroutines, local and global values. Data representation. Number bases and coding systems.	Data representation. Bitmap and vector images. MIDI and audio samples. Compressing and encrypting data. Theory of Computation. Finite State Automations. Networks. Topologies, The Internet and TCP/IP protocol.	Programming, subroutines and scope of variables. Consequences of use of computers. Moral, social, legal and cultural issues and opportunities. Computer Systems, Hardware, software, the OS and translators/compilers.	Boolean logic and logic gates. Internal components of a computer system. The stored program concept. Assembly language. External components of a computer system.	Databases. Databases concepts. Datatables, Relationships, Normalisation techniques. Using SQL queries. Mock exam Paper 1 preparation using skeleton code.	Mock Exams NEA Choice of Project and Analysis. Programming paradigms. Structured/Object Oriented Programming.

Computing/ICT	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Y13 Computer Science	NEA Design and technical solution. Mock exam Paper 1 preparation using skeleton code. Databases SQL queries across multiple tables. Programming abstract data types. Data representation. Rounding errors and floating point. Programming. Recursive techniques. Turing machines and the Halting Problem.	NEA Design and technical solution. Networks. Mock exam Paper 1 preparation using skeleton code. Search and sort algorithms. Time complexity of algorithms. Regular expressions and Backus-Naur Form. Big data.	NEA Testing and evaluation. Mock exam Paper 2 preparation. Work on skeleton code for final exam. Programming paradigms. Structured/Object Oriented Programming.	NEA completion. Revision for exam. Work on skeleton code. Functional Programming. Assembly Language.	Revision.	