Curriculum Plan KS5 Computer Science

Year 12

	Autumn	Spring	Summer
Units/Topics	**1.1 The characteristics of contemporary processors, input, output and storage devices** What is the purpose and function of a microprocessor? The Arithmetic and Logic Unit; ALU, Control Unit and Registers (Program Counter; PC, Accumulator; ACC, Factors affecting the performance of the CPU: clock speed, number of cores, cache. The use of pipelining in a processor to improve efficiency. What is the difference between input and output? How different input, output and storage devices can be applied to the solution of different problems. The uses of magnetic, flash and optical storage devices. RAM and ROM and virtual storage. **1.2 Software and software development** Systems Software Applications Generation Software Development Types of Programming Language **1.3 Exchanging data** Compression, Encryption and Hashing **Problem solving and programming Programming techniques.** Programming constructs: sequence, iteration, branching. (b) Recursion, how it can be used and compares to an iterative approach. (c) Global and local variables. (d) Modularity, functions and procedures, parameter passing by value and by reference. (e) Use of an IDE to develop/debug a program.	**1.3 Networks** Characteristics of networks and the importance of protocols and standards. The internet structure: The TCP/IP Stack. DNS Protocol layering. LANs and WANs. Packet and circuit switching. Network security and threats, use of firewalls, proxies and encryption. Network hardware. **Algorithms.** Sorting algorithms Path finding algorithms **1.3 Databases**. Relational database, flat file, primary key, foreign key, secondary key, entity relationship modelling, normalisation and indexing. Methods of capturing, selecting, managing and exchanging data. Normalisation to 3NF. SQL – Interpret and modify. Referential integrity. Transaction processing, ACID (Atomicity, Consistency, Isolation, Durability), record locking and redundancy.	**1.3 Web Technologies** HTML, CSS and JavaScript. Search engine indexing. PageRank algorithm. Server and client-side processing. What is assembly language? Types of Programming languages. Software development. Understand the waterfall lifecycle, agile methodologies, extreme programming, the spiral model and rapid application development. The relative merits and drawbacks of different methodologies and when they might be used. Writing and following algorithms. **Programming project** **Data structures** Arrays (of up to 3 dimensions), records, lists, tuples. The following structures to store data: linked-list, graph (directed and undirected), stack, queue, tree, binary search tree, hash table. How to create, traverse, add data to and remove data from the data structures mentioned above. *(NB this can be **either** using arrays and procedural programming **or** an object-oriented approach).* **Programming techniques.** Use of object oriented techniques.

	Elements of computational thinking. Data Types. Applications Generation. The nature of applications, justifying suitable applications for a specific purpose. Utilities. Open source vs closed source. Translators: Interpreters, compilers and assemblers. Stages of compilation (lexical analysis, syntax analysis, code generation and optimisation). Linkers and loaders and use of libraries. Computational methods.		
Key Assessment	End of unit assessments Timetabled PPE's	End of unit assessments Timetabled PPE's	End of unit assessments Timetabled PPE's
Why is it studied?	The A Level Computer Science qualification consists of three components, which are taught across two academic years. Where Paper 1 'Computer Systems' focusses on the theory of computing, Paper 2 'Algorithms and programming' has a heavier focus on algorithms and programming. The first term will involve introductory programming to ensure all learners are equipped with a knowledge of procedural programming before we move on to Object Oriented programming. Simultaneously learners are introduced to an understanding of the fundamental principles and concepts of computer science, including: abstraction, decomposition, logic, algorithms and data representation and subsequently how to apply them and carry out decomposition of computational problems in their programming project. As well as the everyday application software we use, we introduce learners to the utility software that runs in the background of what we are doing. How does our operating system load? How to we manage all of that wonderful filing? To allow the ability to analyse problems in computational terms through practical experience of solving such problems including writing programs to solve computational problems. The programming project		

Year 13

	Autumn	**Spring / Summer**
Unit/Topics	**1.4 Data types, data structures and algorithms** Represent positive integers in binary. Use of sign and magnitude and two's complement Addition and subtraction of binary integers. Convert between binary hexadecimal and denary. Representation and normalisation of floating point (positive and negative) numbers in binary. Bitwise manipulation and masks: shifts, combining AND, OR, and XOR. Character sets (ASCII and UNICODE) **1.4 Boolean Algebra** Define problems using Boolean logic. Manipulate Boolean expressions Simplify statements in Boolean algebra: De Morgan's Laws, distribution, association, commutation, double	**1.4 Boolean Algebra** Using logic gate diagrams and truth tables. The logic associated with D type flip flops, half and full adders**.** Karnaugh maps to simplify Boolean expressions. **1.5 Legal, moral, cultural and ethical issues** Computing related legislation The Data Protection Act 1998. The Computer Misuse Act 1990. The Copyright Design and Patents Act 1988. The Regulation of Investigatory Powers Act 2000. Moral and ethical issues The individual moral, social, ethical and cultural opportunities and risks of digital technology: Computers in the workforce. Automated decision making. Artificial intelligence. Environmental effects. Censorship and the Internet.

	negation. **Types of Programming languages** Need for and characteristics of a variety of programming paradigms. Procedural languages. Assembly language (including following and writing simple programs with the Little Man Computer instruction set). See appendix 5d. Modes of addressing memory (immediate, direct, indirect and indexed). Object-oriented languages (see appendix 5d for pseudocode style) with an understanding of classes, objects, methods, attributes, inheritance, encapsulation and polymorphism. **Programming Project**	Monitor behaviour. Analyse personal information. Piracy and offensive communications. Layout, colour paradigms and character sets. **Revision** **Programming Project**
Key Assessment	End of unit assessments Timetabled PPE's	End of unit assessments Timetabled PPE's
Why is it studied?	In the final year of this two-year course, we continue with the programming project as it prepares learners for their algorithms paper and allows them to apply their knowledge of data structures and path finding algorithms. We look at the theory of democracy and rule of law in IT. Our study of environmental and ethical issues in computing encompasses our whole school delivery of mutual respect and tolerance for each other and our environment. Learners will develop the ability to articulate the individual (moral), social (ethical), legal and cultural opportunities and risks of digital technology. This term studies include the theory of algorithms and determining whether a system works as it is intended to as we test it for logic and syntax errors. To develop computing mathematical skills with an emphasis on the mathematical skills used to express computational laws and processes, e.g. Boolean algebra/logic and comparison of the complexity of algorithms.	