Computing Department Rationale

Computers are widely used in all aspects of business, industry, government, education, leisure and the home. In this increasingly technological age, a study of computer science, and particularly how computers are used in the solution of a variety of problems, is not only valuable to the learners themselves but also essential to the future well-being of the country. Computer science integrates well with subjects across the curriculum.

Computing National Curriculum Aims

Eduqas A level in Computer Science encourages learners to develop:

- an understanding of, and the ability to apply, the fundamental principles and concepts of computer science, including abstraction, decomposition, logic, algorithms and data representation
- the ability to analyse problems in computational terms through practical experience of solving such problems, including writing programs to do so
- the capacity for thinking creatively, innovatively, analytically, logically and critically
- the capacity to see relationships between different aspects of computer science
- mathematical skills
- the ability to articulate the individual (moral), social (ethical), legal and cultural opportunities and risks of digital technology

YEAR 12	Content	Skills	Rationale
Autumn Half Term 1	**Teacher 1:** The representation of number	Data representation and data types Bit patterns and hexadecimal Characters Data types Data storage Arithmetic Floating point form Two's complement Truncation and rounding Logical and arithmetic shifts Overflow and underflow	*Component 2* *Topic 3* With some students starting the course with no GCSE Computer Science experience, this topic is a good leveller. It will suit most mathematicians and start the transition to A level.
	Logical Operations	Boolean expressions and truth tables Logical operations, registers and masking Simplification and de Morgans	*Component 1* *Topic 2* Again, this is a good leveller to build confidence
	Teacher 2: Relational databases and distributed systems	Data consistency, redundancy and independence Normalisation ER modelling SQL DBMS Big data Distributed systems	*Component 2* *Topic 5* The GCSE specification does not cover much in the way of database theory, so it is a good idea to introduce this early on as it is an important concept. It also ties in with the understanding required for the style of project work that we do for the practical component 3.

Autumn Half Term 2	**Teacher 1:** Data Structures Algorithms and programs	3D arrays Records Stacks Queues Linked lists Trees Hash tables Records and arrays Efficiency of algorithms Big O notation Linear search Binary search Shortest path Programming constructs Modular programming Logical operation in programming Traversal of data structures Compression algorithms Efficiency of compression algorithms Testing Pseudocode and flowcharting Variables and constants Identifers Scope of variables Parameters Recursion Mathematical calculations Validation and verification Sorting Bubble sort, insertion sort and quicksort	*Component 1* *Topic 1* This is a difficult and intense topic. It is best left until students are feeling more confident in their understanding of algorithms and are working on some practical programming.
	Teacher 2: Complete discussion and investigation for practical project work.	Describe, to others, the broad aims and limitations of the project. Identify and describe the possible limitations of a solution to the problem. Consider and use feedback from others to refine understanding of the problem and proposed solution. Carry out an investigation of the current system using a variety of appropriate methods. Identify stakeholders of the current system and their requirements for the proposed project. Analyse data collected for input and processing by the current system. Identify and describe all outputs from the current system. Produce a working specification that summarises the purpose of the project. Set objectives, including measurable success criteria for the proposed system.	*Component 3* Programmed Solution to a Problem Non-exam assessment 20% of qualification Students must find a 'real world' client to provide a coded solution for. At this stage in the course, they are keen to get coding. It makes sense to harness this enthusiasm to feel like an A level computer scientist.
Spring Half Term 1	**Teacher 1:** Algorithms and programs	Efficiency of algorithms Big O notation Linear search	*Component 1* *Topic 3*

		Binary search Shortest path Programming constructs Modular programming Logical operation in programming Traversal of data structures Compression algorithms Efficiency of compression algorithms Testing	
	Teacher 2: Systems Analysis	Approaches - waterfall and agile Feasibility Investigation Analysis Changeover Alpha, beta and acceptance testing Maintenance Backup and recovery Documentation	*Component 1* *Topic 5*
	Project Design & prototype	Specify, design and document screen layouts, reports and other forms of input and output required to create the user interface. Design and document all data structures that will be required to produce the output for the solution to the problem together with the method of accessing the data in that data structure. **Processing stages:** Design programming routines to be used to handle and process data within the proposed solution to achieve each objective. Document these designs using a structured convention such as pseudo-code. **Prototype:** Produce a range of screens and outputs for the prototype solution. Create a functioning system that carries out all chosen processes. Use realistic data for output and storage.	*Component 3* Throughout the spring term, students continue their project work with guidance from both teachers. A significant amount of time outside the lesson is to be spent on coding and writing up. It makes sense for students to do this at this point in the course, while they still have the time available before the pressures of exam time really hit home.
Spring Half Term 2	**Teacher 1:** The OS	Managing resources Providing an interface Backing store Utilities Multitasking and multi-user systems HCI Interrupts Memory management Scheduling	
Summer Half Term 1	**Teacher 1:** Principles of programming	Programming paradigms OOP Standardisation Backus-Naur	*Component 1* *Topic 4*

	Program constructors **Teacher 2:** Software Engineering **Teacher 2:** Economic, moral and legal issues	High- and low-level languages Compilers, interpreters and assemblers Software tools and IDE Social and economic changes Professional behaviour Effect on employment Legislation	*Component 1* Topic 7 *Component 1* Topic 9
Summer Term	Post-prototype refinement Software development	Obtain feedback from competent third parties. Refine designs in light of the evaluation of the prototype solution and feedback received from others Refine the prototype using the amended design documentation ensuring that the finished system is functional and suitable for audience and purpose. Produce annotated listings for the finished system to facilitate future maintenance.	*Component 3* Practical work continues both inside and outside of lesson time.
Year 13			
Autumn Term 1	**Teacher 1:** Hardware & Communication Data Transmission **Teacher 2:** System Design	Architecture Memory and cabling FE cycle Assembly language programming IO devices Secondary storage Fragmentation Networking communications Protocols Handshaking Wireless technologies Serial and parallel transmission Simplex and duplex transmission Multiplexing and switching TCP/IP and packets Collision detection Data transfer Internet structure HCI Validation Evaluation	*Component 2* *Topic 1* This is a substantial topic, so it makes sense to hit the ground running at the start of the final year. It can then be thoroughly revised in preparation for Block A assessment. *Component 2* Topic 2 *Component 1* Topic 6
Autumn Term 2	**Teacher 1:** Organisation and structure of data	Records and fields Master and transaction files Sequential and random files Hashing algorithms Multi-level indices File generations Backing up and archiving	*Component 2* *Topic 4*

	Teacher 2: Testing & Evaluation	Produce evidence of testing at each stage of development. Produce evidence of all problems encountered and actions taken to overcome these problems. **Design a test plan** to test: - each individual system function with typical, extreme or invalid data - the whole system to ensure that the system produces the correct results for the data input. **Actual test runs** Produce annotated test runs that include commentaries on the outcomes of the testing process. **Evaluation** Produce an evaluation of the programming language used to create the solution. Compare the solution with similar commercially available systems. Identify the successful features of the system and make specific suggestions for improving less successful areas of the system. Describe the strengths and weaknesses of own performance in the design and prototyping of the solution.	*Component 3* Deadline before Christmas. It is essential to stick to this deadline as the danger is the project can expand exponentially, especially when students feel the need for it to be perfect. It is important to be able to concentrate on the examined components as they are after all 80% of the course.
Spring Term 1	**Teacher 1:** Data Security **Teacher 2:** Software	Data integrity Cryptography Biometrics Disaster planning Malicious and accidental damage Malware Penetration testing Open source and off-the-shelf Safety Industrial systems Control systems Expert systems Internet and Intranet	*Component 2* *Topic 8* This is an accessible topic that students feel confident and interested in, so it works to leave it to this point in the course where they are starting to really feel the pressure. However, they still have the focus to take on board the level of detail required. *Component 2* *Topic 7* Similarly, this is an accessible topic which the students won't find too over-facing at this point.
Spring Term2	REVISION	Simplification and de Morgans Big O notation Traversal of data structures Backus-Naur Economic, moral and legal issues	*Component 1* Students are always keen to revise these practical problem solving/mathematical type topics.
Summer Term	REVISION	Assembly language programming Data representation and data types SQL Data security	