The Fisher Way: Curriculum

The Fisher Way aims to educate and inspire with joy, faith and love because we are an inclusive Catholic community.

Successful and resilient learners who aspire to and achieve excellence

Confident individuals who can explore and communicate effectively

Responsible citizens who are active, loving and wise in all their endeavours

Subject	A Level Computer Science
Year Group	Year 13
Intent	**Successful and Resilient Learners:** who can analyse problems and find efficient, creative solutions for both real world situations and artificial systems **Confident Individuals**: who can use digital technology effectively, creatively and safely in their personal lives and future careers **Responsible Citizens**: who understand the social and cultural impact that technology has on their lives and the lives of others.

Narrative	The Year 13 Computer Science curriculum completes the AQA A Level Computer Science course, bringing together the non exam assessment, Paper 1, the on screen programming and problem solving exam, and Paper 2, the written theory exam. In the autumn term, students are introduced to the Paper 1 skeleton program and its preliminary material, learning to read, understand and extend a large program written by someone else. Alongside this, they develop their NEA programming project, completing their analysis and design and building their technical solution for a real client. In Spring 1, students complete and test their NEA project and evaluate it against their objectives. In practical programming, they implement more advanced algorithms. In Spring 2, the focus moves to exam preparation, with intensive work on the skeleton code to prepare for Paper 1 and structured revision of the Paper 2 theory. In Summer 1, students revise data structures, the theory of computation and algorithms, practising exam style questions until the final exams. By the end of Year 13, students have completed an independent software project and are ready to sit both papers, prepared for university study, apprenticeships or careers in computing.					
Half term	Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
Knowledge (topics studied)	Introduction to the Skeleton Program NEA Programming Project		NEA Programming Project Practical Programming: Advanced Algorithms	Skeleton Code Paper 2 Revision	Revision: Data Structures Revision: Theory of Computation Revision: Algorithms	
Key skills	Students read and navigate the AQA skeleton program, tracing how its classes, subroutines and data structures work together, and practise making the kind of changes asked for in Paper 1. In their NEA, they complete their analysis with measurable objectives, produce a documented design including data structures, algorithms and user interfaces, and build their technical solution using appropriate programming techniques.		Students complete their technical solution, test it thoroughly with evidence of normal, boundary and erroneous data, and evaluate it against their objectives with feedback from their client. In practical programming, they implement advanced algorithms such as graph traversals, Dijkstra's shortest	Students analyse the skeleton code in depth, predict likely exam questions, and practise writing, testing and explaining modifications under timed conditions. They revise the Paper 2 theory systematically, including data representation, computer systems and architecture, networking, and the consequences of computing, and	Students revise abstract data types such as stacks, queues, graphs, trees, hash tables and vectors. They revise the theory of computation, including abstraction, finite state machines, regular expressions, Turing machines, Big O notation and the limits of computation. They revise standard	

		path, tree traversals and recursive solutions, and compare their efficiency.	practise extended response questions.	algorithms and practise tracing and comparing them in exam style questions.	
Cultural capital	Working with a large, unfamiliar codebase mirrors the reality of professional software development, where programmers spend more time reading and changing existing code than writing new code. Developing a project for a real client builds independence, communication and project management skills valued at university and in industry.	Students experience the full software development lifecycle, from a client's problem to a tested, evaluated product. Implementing advanced algorithms shows them the techniques behind route planning, search engines and social networks.	Students learn to manage their own revision and to perform under pressure, skills that carry directly into higher education and the workplace.	Students gain a deeper appreciation of the mathematical ideas that underpin computer science and what computers can and cannot do. This prepares them for the step up to university study in computing, mathematics and engineering.	
Assessment	Skeleton Program Tasks NEA Analysis and Design Mock Exams	NEA Final Submission	Mock Exams Past Paper Practice	Paper 1 and Paper 2 Exams	