Computing A Level Key Stage 5 Curriculum Plan — Year 12

Exam Board: OCR

Qualification: *OCR A Level Computer Science (H446)*

Assessment Information:

Component 01 – Computer Systems
Written exam, 2 hrs 30 mins (40%)
Component 02 – Algorithms & Programming
Written exam, 2 hrs 30 mins (40%)
Component 03 – Programming Project (NEA)
(20%)

Link to official specification

Department Information:

Computing is taught to all KS3 students. Year 7 & 8 have one lesson per week and Year 9 have three lessons over a two-week period.

Computing is chosen as an option for both the OCR GCSE and OCR A Level qualifications.

All lessons are delivered by specialist Computing teachers with relevant industry experience.

ACHIEVE in the curriculum:

Students are expected to be Ambitious during their A Level course. They will have opportunities to Collaborate on tasks with their peers.

Students can demonstrate their Integrity, Endurance and Versatility particularly in challenging topics such as programming, Boolean algebra and algorithmic complexity.

Excellence is a consistent expectation in written exam preparation and in the practical programming project (NEA).

Curriculum Aims & Intent:

The aim is for students to understand and apply the fundamental principles and concepts of Computer Science, including: the structure and function of computer systems, programming techniques, algorithms and data structures, Boolean algebra and networking.

Students will analyse and solve problems through practical experience by designing, writing, testing and debugging programs — developing skills essential for higher education and employment in the computing industry.

Resources:

OCR A Level resources, OCR past papers and mark schemes, PG Online textbooks, Isaac Computer Science, laptops and computers.

Isaac Computer Science – OCR A Level Topics
OCR Official Specification Page

How we keep parents informed:

Year 12 — Progress reports are published 4 times per year, in October, January, March and July, with a face-to-face parents' evening in November.

Parents will also be kept informed of any concerns regarding effort or progress through direct communication from the class teacher.

How parents can help their child:

Parents/carers can help by providing a suitable study space and encouraging good study habits and time management.

Encouraging students to be curious, explore computing topics beyond the classroom, and practise programming regularly will significantly aid their progress.

What we study and when:

Term	Unit / Topic	Knowledge, Understanding & Skills Developed	Resource Links	ACHIEVE / Personal Development	Careers Links
1	**Programming Techniques**	Programming constructs: sequence, selection and iteration. Recursion and comparison to iterative approaches. Global and local variables, scope and lifetime. Modularity, functions and procedures; parameter passing by value and by reference. Use of an IDE to develop, test and	Isaac CS – Subroutines Isaac CS – Recursion Isaac CS – IDEs OCR Learner Activity (DOCX)	Ambitious, Collaborative, Integrity, Endurance	Software Developer / Engineer

Term	Unit / Topic	Knowledge, Understanding & Skills Developed	Resource Links	ACHIEVE / Personal Development	Careers Links
		debug programs. Object-oriented techniques: classes, objects, attributes, methods, inheritance, encapsulation and polymorphism.	OCR Teacher Instructions (PDF) OCR Delivery Guide		
	Thinking Abstractly	The nature of abstraction and why it is essential in computing. The differences between an abstraction and reality. How to identify relevant details and filter out unnecessary information. Devise and evaluate abstract models for a variety of real-world situations, including entity-relationship models and state-transition diagrams.	Isaac CS – Computational Thinking OCR Delivery Guide	Analytical thinking, Versatility	Systems Analyst / Modeller
	Thinking Ahead	Identifying the inputs and outputs for a given situation. Determining the preconditions that must be satisfied before devising a solution. The nature, benefits and drawbacks of caching: when to cache and when not to. The need for reusable program components and the benefits of decomposing problems into modular routines that can be tested independently.	Isaac CS – Computational Thinking OCR Learner Activity (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Planning, Endurance	Software Architect
	Thinking Procedurally	Identifying the components of a problem and its solution. Determining the correct order of steps needed to solve a problem. Identifying the sub-procedures necessary to solve a problem and how they interact. Structuring a solution as a hierarchy of procedures with clear interfaces, enabling each sub-procedure to be designed and tested in isolation.	Isaac CS – Computational Thinking	Logical reasoning, Excellence	Automation Engineer
	Thinking Logically	Identifying the points in a solution where a decision must be taken. Determining the logical conditions that affect the outcome of a decision. Determining how decisions affect the overall flow of a program, including nested conditions and compound Boolean expressions.	Isaac CS – Computational Thinking OCR Learner Activity (DOCX) OCR Teacher Instructions (PDF)	Critical thinking, Integrity	Cybersecurity Analyst
	Thinking Concurrently	Determining which parts of a problem can be solved simultaneously. Outlining the benefits and trade-offs of concurrent processing for a given situation. Understanding potential synchronisation issues, race conditions and deadlock. Awareness of parallel processing architectures and how concurrent execution differs from sequential execution.	Isaac CS – Computational Thinking OCR Learner Activity 1 (DOCX) OCR Learner Activity 2 (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Collaboration, Efficiency	Parallel Computing Engineer
	Input, Output & Storage	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 and their relative advantages. RAM and ROM: their characteristics, differences and typical uses. Virtual storage, cloud storage and their advantages and limitations in practical scenarios.	Isaac CS – Memory and Storage Isaac CS – Hardware OCR Delivery Guide	Technical knowledge, Versatility	Hardware / Systems Engineer
	Structure & Function of Processor	The role of the ALU, Control Unit and Registers: Program Counter (PC), Accumulator (ACC), Memory Address Register (MAR), Memory Data Register (MDR) and Current Instruction Register (CIR). Data, address and control buses and their widths. The Fetch-Decode-Execute cycle and its effect on each register. Factors affecting CPU performance: clock speed, number of cores and cache memory size.	Isaac CS – Systems Architecture OCR Learner Activity (DOC) OCR Teacher Instructions (PDF) OCR Delivery Guide (PDF)	Deep technical understanding, Excellence	Embedded Systems / CPU Engineer

Term	Unit / Topic	Knowledge, Understanding & Skills Developed	Resource Links	ACHIEVE / Personal Development	Careers Links
		Pipelining to improve processor efficiency. Von Neumann, Harvard and contemporary processor architectures.			
	Types of Processor	The differences between and uses of CISC and RISC processors. GPUs and their uses beyond graphics, including machine learning and scientific simulation. Multicore and parallel systems: how they work, their advantages and limitations in practice.	Isaac CS – Systems Architecture OCR Learner Activity (DOC) OCR Teacher Instructions (PDF) OCR Delivery Guide	Technical curiosity, Ambition	HPC / Systems Engineer
2	**System Software**	The need for, function and purpose of operating systems. Memory management: paging, segmentation and virtual memory — how they work and their trade-offs. Interrupts and Interrupt Service Routines (ISR): their role and interaction with the FDE cycle. Scheduling algorithms: round robin, first come first served, multi-level feedback queues, shortest job first, shortest remaining time. Types of OS: distributed, embedded, multi-tasking, multi-user and real-time. BIOS, device drivers and virtual machines.	Isaac CS – Operating Systems Isaac CS – Translators OCR Learner Activity (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Integrity, Versatility, Endurance	Systems / OS Engineer
	Application Generation	The nature of applications software and justifying suitable applications for specific purposes. Utilities and their role in maintaining a system. Open source vs closed source software: comparative advantages, disadvantages and licensing implications. Translators: interpreters, compilers and assemblers and when each is appropriate. Stages of compilation: lexical analysis, syntax analysis, semantic analysis, code generation and optimisation. Linkers, loaders and the use of libraries.	Isaac CS – Translators Isaac CS – High- and Low-Level Languages OCR Learner Activity (DOCX) OCR Teacher Instructions (PDF)	Problem solving, Versatility	Software / DevOps Engineer
	Computational Methods	Features that make a problem solvable by computational methods. Problem recognition and decomposition into smaller, manageable sub-problems. Use of divide and conquer. Application of abstraction to simplify problems. Strategies: backtracking, data mining, heuristics, performance modelling, pipelining and visualisation applied to a range of real-world problems.	Isaac CS – Computational Thinking OCR Teacher Instructions (PDF) OCR Delivery Guide	Innovation, Endurance, Ambition	Data Scientist / AI Engineer
	Algorithms	Analysis and design of algorithms for a given situation. Suitability of algorithms in terms of execution time and space requirements. Standard algorithms: bubble sort, insertion sort, merge sort, quick sort, binary search, linear search, Dijkstra's shortest path. Big O notation: $O(1)$, $O(n)$, $O(n^2)$, $O(2^n)$, $O(\log n)$. Data structures: stacks, queues, trees, linked lists, graphs. Depth-first (pre-order, in-order, post-order) and breadth-first traversal of trees.	Isaac CS – Searching Algorithms Isaac CS – Sorting Algorithms Isaac CS – Data Structures Isaac CS – Algorithmic Complexity OCR Learner Activity (DOC) OCR Teacher Instructions (PDF)	Logical thinking, Excellence	Algorithm / Software Engineer
3	**Compression, Encryption & Hashing**	Lossy vs lossless compression and appropriate use cases for each. Run-length encoding and dictionary coding as lossless techniques. Symmetric encryption (e.g. AES) and asymmetric encryption (e.g. RSA): operation and comparative strengths. Public key cryptography, digital signatures and certificate authorities. Hashing for password storage (with salting) and checking data integrity.	Isaac CS – Compression Isaac CS – Encryption OCR Teacher Instructions (PDF) OCR Delivery Guide	Security awareness, Integrity	Cybersecurity Engineer

Term	Unit / Topic	Knowledge, Understanding & Skills Developed	Resource Links	ACHIEVE / Personal Development	Careers Links
	Data Types	Primitive data types: integer, real/floating point, character, string and Boolean. Represent positive integers in binary. Sign-and-magnitude and two's complement for negative numbers. Binary addition and subtraction with overflow detection. Hexadecimal representation and conversion between binary, hex and denary. Normalised floating-point representation in binary including mantissa and exponent. Floating-point arithmetic and rounding errors. Bitwise manipulation and masks: logical shifts, AND, OR and XOR. ASCII and Unicode and why Unicode was introduced.	Isaac CS – Representation of Numbers Isaac CS – Representation of Text OCR Learner Activity 1 (DOCX) OCR Learner Activity 2 (DOCX) OCR Learner Activity 3 (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Precision, Endurance, Excellence	Data / Firmware Engineer
4	**Types of Programming Language**	Procedural programming: program flow, variables and constants, procedures and functions, arithmetic, Boolean and assignment operators, string and file handling operations. Assembly language and the Little Man Computer (LMC) instruction set. Modes of addressing memory: immediate, direct, indirect and indexed. Object-oriented programming: classes, objects, methods, attributes, inheritance, encapsulation and polymorphism. Event-driven programming: event handlers, triggers and the role of the event loop.	Isaac CS – Procedural Programming Isaac CS – Object-Oriented Programming Isaac CS – High- and Low-Level Languages OCR Learner Activity (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Versatility, Collaboration, Endurance	Software Developer / Researcher
	Databases	Relational vs flat-file databases. Entity-relationship modelling and constructing ER diagrams. Primary, foreign and secondary keys. Normalisation to 3NF: first, second and third normal forms with worked examples. Methods of capturing, selecting, managing and exchanging data. SQL: SELECT, FROM, WHERE, GROUP BY, ORDER BY, INSERT INTO, UPDATE, DELETE. Referential integrity. Transaction processing and ACID properties (Atomicity, Consistency, Isolation, Durability), record locking and redundancy.	Isaac CS – Database Concepts Isaac CS – SQL	Accuracy, Integrity	Database Administrator / Engineer
5	**Boolean Algebra**	Defining problems using Boolean logic and constructing Boolean expressions from real-world conditions. Manipulating Boolean expressions using the rules of Boolean algebra. Applying De Morgan's Laws plus the rules of distribution, association, commutation and double negation. Karnaugh maps (K-maps) for simplifying Boolean expressions. Constructing and interpreting logic gate diagrams and truth tables. The logic of D-type flip-flops, half adders and full adders.	Isaac CS – Boolean Logic OCR Learner Activity 1 (DOCX) OCR Learner Activity 2 (DOCX) OCR Learner Activity 3 (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Abstract reasoning, Excellence	Computer / Logic Engineer
	Networks	Characteristics of networks and the importance of protocols and standards. Internet structure and the TCP/IP protocol stack: application, transport, network and link layers. DNS: purpose and how name resolution works. LANs and WANs. Packet switching vs circuit switching. Network security threats: malware, phishing, man-in-the-middle attacks; countermeasures: firewalls, proxies and encryption. Network hardware: routers, switches, hubs, WAPs. Client-server and peer-to-peer models.	Isaac CS – Networking Isaac CS – Network Hardware OCR Teacher Instructions (PDF) OCR Delivery Guide	Connectivity, Collaboration	Network / Security Engineer
	Web Technologies	HTML for structure, CSS for presentation and JavaScript for client-side interactivity. How search engines use crawlers to build and maintain an index. The PageRank algorithm and how link analysis determines	Isaac CS – Web Technologies OCR Learner Activity (DOCX)	Creativity, Versatility	Web / Full-Stack Developer

Term	Unit / Topic	Knowledge, Understanding & Skills Developed	Resource Links	ACHIEVE / Personal Development	Careers Links
		relevance. Server-side vs client-side processing, including advantages and trade-offs. Differences between static and dynamic web pages and the role of server-side scripting.	OCR Teacher Instructions (PDF) OCR Delivery Guide		
6	**Computing Related Legislation**	The Data Protection Act 1998 / GDPR and the eight principles of data protection. The Computer Misuse Act 1990: three categories of offence and their penalties. The Copyright, Designs and Patents Act 1988 and its implications for software licensing. The Regulation of Investigatory Powers Act 2000 and its relevance to digital surveillance and communications interception.	Isaac CS – Legislation	Ethics, Integrity, Responsibility	IT Compliance / Legal Officer
	Moral & Ethical Issues	Individual, social, ethical and cultural opportunities and risks of digital technology. Computers in the workforce: automation and job displacement. Automated and AI decision-making: transparency, bias and accountability. Environmental effects of computing: energy use of data centres, e-waste. Censorship and freedom of speech online. Monitoring and surveillance of citizens and employees. Analysing personal data and privacy implications. Piracy, offensive digital communications and cyberbullying.	http://www.empower-yourself-with-color-psychology.com/cultural-color.html Isaac CS – Legislation Isaac CS – A Level Topics (OCR)	Responsibility, Reflection, Integrity	IT Ethics Consultant / Policy Adviser
	Software Development	Software development methodologies: waterfall, agile, extreme programming (XP), rapid application development (RAD) and the spiral model. Testing strategies: black-box testing, white-box testing, integration testing, acceptance testing and regression testing. Designing test data: normal, boundary and erroneous values. Trace tables and dry-run testing. Defensive programming techniques: input validation, authentication, error handling and code maintainability through structured programming.	Isaac CS – Software Development OCR Learner Activity (DOCX) OCR Teacher Instructions (PDF) OCR Delivery Guide	Precision, Collaboration, Excellence	QA Engineer / Software Developer
	Functional Programming	The functional programming paradigm and how it differs from procedural and OOP approaches. First-class and higher-order functions: functions as values that can be passed and returned. Function composition and the benefits of pure functions with no side effects. Partial function application and currying. Key operations: map, filter and fold/reduce applied to lists. Lazy evaluation and its benefits for efficiency. Advantages of functional programming in parallel and concurrent systems.	Isaac CS – Functional Programming OCR Delivery Guide: Types of Programming Language	Abstract reasoning, Innovation, Ambition	Functional / Research Software Engineer