2026–27

Computer Science Department – Curriculum Intent

KS3 Curriculum Intent			
Head of Department: Miss S Cooper/Mr S Verma | Suggested term allocation			
	Year 7	**Year 8**	**Year 9**
Overview	Introduction to the school network and Office 365 App development in AppShed Games programming in Scratch Using computers safely and effectively Introduction to coding through Kodu	Computer crime and cyber security Graphics HTML and website development Introduction to Python Understanding computers	Computational thinking and logic Python next steps Spreadsheet modelling AI and machine learning Networks
Autumn Term	**Introduction to the school network and Office 365 –** Use school accounts securely, organise files, access email and use Office 365 tools to support learning. **Introduction to coding through Kodu –** Apply logical thinking in a different visual programming environment. Create games using rules and events, then test and improve solutions.	**Computer crime and cyber security –** Build on Year 7 online safety by exploring digital threats, their impact and ways to protect devices, accounts and data. **Graphics –** Create and edit digital images for a purpose and audience. Consider image quality, file formats and responsible use of images.	**Computational thinking and logic –** Use decomposition, abstraction and algorithms to solve problems. Apply AND, OR and NOT to logical decisions. **Python next steps –** Extend Year 8 programming through more demanding problems, combining selection and iteration. Develop independence in tracing, testing, debugging and refining solutions.
Spring Term	**Games programming in Scratch –** Develop programs using sequence, selection, repetition and variables. Plan, test and debug games, building confidence in visual programming. **Using computers safely and effectively –** Develop responsible online behaviour, protect personal information, evaluate online content and know how to report concerns.	**HTML and website development –** Use HTML to structure linked web pages with text and images. Apply design skills, test navigation and evaluate the finished website. **Introduction to Python –** Transfer ideas from visual programming to text-based code. Use input, output, variables, data types, selection and repetition to solve problems. Read, test and debug programs.	**Spreadsheet modelling –** Use formulae, functions and cell references to build models. Analyse data, create suitable charts and explore how changing inputs affects results. **AI and machine learning –** Explore how systems learn patterns from data and make predictions. Consider training data, accuracy, bias, privacy and responsible use. Apply logical thinking to evaluate the benefits and limitations of AI.
Summer Term	**App development in AppShed –** Design and create an app for a chosen audience. Organise screens and navigation, add interactive features, and test and evaluate usability.	**Understanding computers –** Explore hardware, software, input, processing, output and storage. Develop an understanding of how computer systems work together.	**Networks –** Explain how devices connect and share data. Explore network hardware, wired and wireless connections, and the internet.

Computer Science Department – Curriculum Intent

Overview of KS4 Curriculum 2026–27		
GCSE Computer Science | Pearson Edexcel 1CP2 | Approximate completion targets		
Period	Year 10	Year 11
Autumn 2026	**Unit 1 – Computational thinking (complete by December 2026)** 2.1.1 Computational thinking 2.1.2 Designing, creating and refining algorithms 2.1.3 Searching and sorting algorithms Develop decomposition, abstraction, logical reasoning and algorithmic problem solving, with theory linked directly to Python practice. **Programming alongside theory** Introduction to Python using Learning to Program in Python. Use Tkinter/Pygame where appropriate and build confidence through regular programming challenges.	**Unit 3 – Computers (finish/recap by end of September 2026)** Complete remaining Year 10 computer systems content and revisit CPU architecture, performance, embedded systems, operating systems and utility software as required. **Paper 2 mini mock** October 2026, in a double lesson before half term. **Unit 4 – Networks (complete for November 2026 mocks)** Networks and topologies Wired and wireless networks Protocols and layers Threats to computer systems and networks Identifying and preventing vulnerabilities Apply network knowledge to realistic security scenarios.
Spring 2027	**Unit 2 – Data (complete by the end of March 2027)** Primary storage Secondary storage Units of data Data storage and representation Compression Develop understanding of how data is represented, stored and handled by computer systems. **Programming** Continue Python problem solving and regular challenges, including use of GUIs where appropriate.	**Unit 5 – Issues and impact (complete by February half term 2027)** Computer systems in the modern world Ethical, cultural and environmental issues Legislation and privacy Evaluate the opportunities, risks and wider impacts of digital technology. **Unit 6 – Programming (target coverage by February 2027)** Programming fundamentals Data types Additional programming techniques Defensive design Testing Boolean logic Programming languages Integrated development environments (IDEs) Cover lists and Turtle in detail and continue Python challenges throughout the year.

Computer Science Department – Curriculum Intent

Overview of KS4 Curriculum 2026–27		
GCSE Computer Science | Pearson Edexcel 1CP2 | Approximate completion targets		
Period	Year 10	Year 11
Summer 2027	**Unit 3 – Computers (summer term 2027)** Architecture of the CPU CPU performance Embedded systems Operating systems Utility software Link hardware and software concepts to how complete computer systems operate. **Unit 6 – Programming (summer focus)** Programming fundamentals Data types Additional programming techniques Python practice continues throughout Year 10 in preparation for Year 11 and Paper 2.	**Consolidation and examination preparation** Revision and exam practice across earlier units. Delivery is adapted according to students' strengths and misconceptions identified through teacher assessment. Use past-paper packs, retrieval practice and practical programming tasks to strengthen Paper 1 and Paper 2 performance.
Throughout the year	**Python and independent practice** Work through Learning to Program in Python, ZigZag Python challenges and Paper 2-style practice. Extend into Time2Code, GOCODEIT and Mission Encodable after core work is secure. **Assessment** Use Smart Revise for homework and classwork, with retrieval and practical programming used to identify misconceptions.	**Python and Paper 2** Use Time2Code challenges in GOCODEIT, dual coding and ZigZag Python challenges. Continue practical programming into the summer term, including project-style tasks based on former NEA material where useful. **Assessment** Use Smart Revise for homework and class tests alongside past-paper and practical evidence.
Timing and progression	Dates are approximate completion targets from SOW Plan 2026–27. Year 10 builds on KS3 computational thinking and Python before extending into data and computer systems.	Year 11 completes and revisits Year 10 content, develops networks and wider impacts, and strengthens independent programming and examination technique.

Computer Science Department – Curriculum Intent

Overview of KS5 Curriculum 2026–27

A Level Computer Science | OCR H446 | Approximate completion targets

Period	Year 12 SKV	Year 12 SAC	Year 13 SKV	Year 13 SAC
Autumn 2026	**Unit 1 – Components of a computer and their uses (October half term)** 1.1.1 Structure and function of the processor: ALU, control unit, registers, buses and the fetch-decode-execute cycle; CPU performance; pipelining; Von Neumann, Harvard and contemporary architectures. 1.1.2 Types of processor: CISC/RISC, GPUs, multicore and parallel systems. 1.1.3 Input, output and storage: selecting suitable devices; magnetic, flash and optical storage; RAM/ROM and virtual storage. **Unit 3 – Software development (December)** Programming paradigms; procedural languages; assembly language and Little Man Computer; memory addressing modes; object-oriented programming including classes, objects, methods, attributes, inheritance, encapsulation and polymorphism.	**Unit 2 – Systems software and applications generation (October half term)** Operating systems: purpose and functions; memory management; interrupts and ISRs; scheduling; distributed, embedded, multitasking, multi-user and real-time systems; BIOS; device drivers; virtual machines. Applications generation: applications and utilities; open- vs closed-source software; translators; compilation stages; linkers, loaders and libraries. **Unit 6 – Data types (December)** Primitive data types; binary representation of positive and negative integers; sign-and-magnitude and two's complement; binary arithmetic; hexadecimal; binary/hex/denary conversion; floating-point representation and normalisation; floating-point arithmetic; bitwise manipulation and masks; ASCII and Unicode. **Programming** Introduction to C# and Unity, using teacher tutorials and the Unity learning hub.	**Unit 3 – Software development (October half term)** Revisit and extend programming paradigms, procedural and assembly languages, addressing modes and object-oriented concepts. Apply knowledge to exam-style questions and programming contexts. **Unit 12 – Algorithms (December)** Analyse and design algorithms for a given situation; judge suitability for a task/data set; compare execution time and space; determine efficiency using Big O notation.	**Unit 7 – Data structures (November/December)** Arrays (up to three dimensions), records, lists and tuples. Linked lists, directed/undirected graphs, stacks, queues, trees, binary search trees and hash tables. Create, traverse, add data to and remove data from these structures.
Spring 2027	**Unit 4 – Exchanging data (February half term)** Databases: relational databases and flat files; primary, foreign and secondary keys; entity-relationship modelling; normalisation and indexing; methods of capturing, selecting, managing and exchanging data; normalisation to 3NF; SQL interpretation/modification; referential integrity; transaction processing, ACID, record locking and redundancy.	**Unit 7 – Data structures (work towards April/May completion)** Arrays, records, lists and tuples; linked lists, graphs, stacks, queues, trees, binary search trees and hash tables. Develop confidence in creating, traversing, adding to and removing from these structures and applying them to problem solving.	**Unit 5 – Networks and web technologies (February half term)** 1.3.3 Networks: characteristics, protocols and standards; TCP/IP stack; DNS; protocol layering; LANs/WANs; packet and circuit switching; network security and threats; firewalls, proxies and encryption; network hardware; client-server and peer-to-peer. 1.3.4 Web technologies: HTML, CSS and JavaScript; search-engine indexing; PageRank; server-side and client-side processing.	**Unit 8 – Boolean algebra (February half term)** Define problems using Boolean logic; manipulate and simplify Boolean expressions including Karnaugh maps; apply De Morgan's laws, distribution, association, commutation and double negation; use logic-gate diagrams and truth tables; understand D-type flip-flops and half/full adders. **Unit 9 – Legal, moral, ethical and cultural issues (April)** Consider moral, ethical and cultural opportunities and risks of digital technology, including computers in the workforce, automated decision making, AI, environmental effects, censorship, monitoring, personal information, piracy and offensive communications.

Computer Science Department – Curriculum Intent

Overview of KS5 Curriculum 2026–27

A Level Computer Science | OCR H446 | Approximate completion targets

Period	Year 12 SKV	Year 12 SAC	Year 13 SKV	Year 13 SAC
Summer 2027	**Unit 12 – Algorithms (May/June)** Analysis and design of algorithms; selection of suitable algorithms; comparison of execution time and space; efficiency and Big O notation. **NEA** Begin documentation in May/June. Complete Analysis before the summer break and begin Design for completion over the summer holidays.	**Unit 7 – Data structures (complete April/May)** Complete the data-structures content and apply structures in programming and exam questions. **NEA** Support the Year 12 analysis and design milestones. Students should brainstorm suitable project ideas and use the project guidance to confirm that the proposed solution is appropriate.	**Revision and examination preparation** Revisit Year 12 and Year 13 theory and programming. Use assessment evidence to target misconceptions and areas of weakness. Complete past papers and focused exam practice.	**Unit 9 – complete in April if required** Finish any remaining legal, moral, ethical and cultural issues content, then consolidate and revise earlier modules using assessment evidence and past-paper practice.
Programming and NEA	**Unit 11 – Programming techniques (throughout Year 12)** Sequence, iteration and branching; recursion; global/local variables; modularity; functions and procedures; parameter passing by value/reference; IDE development/debugging; object-oriented techniques. SKV and SAC support programming, with SKV leading C# and Unity.	**C# and Unity** Students develop practical programming alongside theory. Unity Essentials target: 22 November 2026. Teacher tutorials and the online learning hub support independent progress. **NEA preparation** Practical work builds the programming, testing and documentation habits required for the NEA.	**NEA** Follow the deadlines published in Teams. Apply analysis, design, development, testing and evaluation skills to the individual project. Use teacher assessment and moderation to secure the required evidence.	**NEA** Use the Teams schedule for the current cohort. Development and testing should support clear project documentation, with marking and moderation completed in line with departmental deadlines.
Planning notes	**Planning notes** Year 12 and Year 13 follow the 2026–27 cohort plan. Completion dates are approximate targets and may be adjusted in response to assessment evidence.	**Progression** Theory and programming are deliberately interleaved. C# and Unity provide sustained practical application while OCR specification content is taught in sequenced units.	**Assessment** Revision, past papers and programming tasks are used to diagnose misconceptions and revisit earlier content before the final examinations.	**Independent study** Smart Revise and Isaac Computer Science support homework, class tests and wider retrieval/practice alongside departmental resources.

2026–27

Computer Science Department – Curriculum Intent

Curriculum Rationale:

It is the true belief of our Computer Science department that Computer Science empowers children's thinking for logic and problem solving which is one of the major skills needed in the modern job market and at University. With that in mind we have created a Computer Science curriculum that encompasses problem solving and real life programming with languages that are used in industry eg Python and C#. At KS3, we connect theory with practical work in Kodu, Scratch, AppShed and Python, alongside graphics, website development and spreadsheet modelling. The aims of this curriculum are to enable learners to develop:

- An understanding and 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 to think creatively, innovatively, analytically, logically and critically
- The capacity to see relationships between different aspects of computer science
- Mathematical skills.

This curriculum is specified in detail to ensure that knowledge is remembered (not merely encountered) and built upon, enabling cognitive retrieval through sequential mapping of key concepts and synopsis. By grounding computational thinking skills in relevant and enriching knowledge, students become scholarly and confident demonstrating deeper understanding.

At KS3 (Years 7–9), our intent is to develop confident, responsible users and creators of technology. Year 7 begins with the school network and Office 365, followed by app development in AppShed, games programming in Scratch, using computers safely and effectively, and coding through Kodu. Year 8 extends online safety through computer crime and cyber security, develops creative skills in graphics and HTML website development, introduces Python, and explores how computers work. Year 9 develops computational thinking and logic, extends Python programming, then applies digital skills through spreadsheet modelling, AI and machine learning, and networks. Across the key stage, students move from visual programming to text-based code and apply earlier learning to increasingly demanding problems. Safe and responsible use of technology is reinforced throughout. By the end of KS3, students should be able to design, test and improve digital solutions, explain how computer systems and networks support everyday tasks, and evaluate the use of technology responsibly. These foundations support GCSE Computer Science and digital learning across other subjects.

At KS4, students follow Pearson Edexcel GCSE Computer Science (1CP2). Year 10 develops computational thinking, data and computer systems alongside regular Python programming. Year 11 completes and consolidates this learning, extends understanding of networks and the impact of computing, and strengthens practical programming for Paper 2. At KS5, students follow OCR A Level Computer Science (H446). Parallel teaching by SKV and SAC develops understanding of computer systems, software, data and algorithms, while C# and Unity provide opportunities to apply programming techniques. Year 12 introduces NEA analysis and design; Year 13 develops the project in line with the published Teams milestones and consolidates theory for examinations. Across both key stages, students analyse problems, develop and test solutions, and use assessment feedback to improve their understanding and independence.

2026–27

Computer Science Department – Curriculum Intent

Careers:

The ability to program in Python, C# and HTML which is taught at KS3-5 can lead to the following careers if those skills are pursued. Analytical skills such as debugging programs and recognising how to fix them is a skill that is valued amongst employers, as well as the below career fields in Computer Science.

Application analyst

Applications developer

Cyber security analyst

Data analyst

Database administrator

Forensic computer analyst

Game designer

Games developer

IT consultant

Software engineer

Systems analyst

UX designer

Web designer

Web developer