**Key Stage 4 Curriculum Overview Computing**
**OCR GCSE 601/8355/X**

**Curriculum Intent**

To 'Live Life in All Its Fullness' in the world that Holywell students will encounter as adults will require the embracing of technology as a discerning, responsible and competent Digital Citizen. In this light, our Computing Curriculum is designed to empower students with the foundational knowledge, skills, and understanding necessary to thrive in a rapidly evolving digital world. Our curriculum aims to inspire a passion for technology and problem-solving, equipping students with the confidence and competence to navigate, create, and innovate in the digital landscape. By fostering a deep understanding of computing principles, we strive to develop critical thinkers who are not only adept at using technology but also capable of understanding and shaping its future.

Through a blend of theory, practical application, and project-based learning, the KS3 Computing Curriculum seeks to nurture well-rounded individuals who are not only proficient in technology but also capable of using it to enhance their lives and the lives of others. We are dedicated to creating an inclusive and supportive environment that encourages every student to explore, discover, and thrive. Our approach to Computer Science is a side by side one where the students develop both theoretical knowledge and programming skills throughout the two years.

**How students are assessed**

Two exams – worth 50% of the qualification – both exams are 1 hour 30 minutes
Paper 1: Computer Systems
Paper 2: Computational Thinking, Algorithms and Programming

| Term | Year 10 | Year 11 |
|---|---|---|
| **Autumn term** | **Title of unit: Boolean logic (2.4.1)**<br>**By the end of the unit, students will have learned about:**<br>Simple logic diagrams using the operations AND, OR and NOT<br>Truth tables<br>Combining Boolean operators using AND, OR and NOT<br>Applying logical operators in truth tables to solve problems<br><br>**Title of unit: Units (1.2.3)**<br>**By the end of the unit, students will have learned about:**<br>The units of data storage such as Bit and Byte etc:<br>How data needs to be converted into a binary format to be processed by a computer.<br>Data capacity and calculation of data capacity requirements<br><br>**Title of unit: Data storage (1.2.4) & Compression (1.2.5)**<br>**By the end of the unit, students will have learned about:** | **Title of unit: Threats to computer systems and networks (1.4.1)**<br>**By the end of the unit, students will have learned about:**<br>Forms of attack including Malware, Social engineering (phishing) and brute force attacks<br><br>**Title of unit: Defensive design (2.3.1)**<br>**By the end of the unit, students will have learned about:**<br>Defensive design considerations:<br>• Anticipating misuse<br>• Authentication<br>Input validation<br>Maintainability:<br>• Use of sub programs<br>• Naming conventions<br>• Indentation<br>• Commenting |

| | | |
|---|---|---|
| | Numbers<br>How to convert positive denary whole numbers to binary numbers (up to and including 8 bits) and vice versa<br>How to add two binary integers together (up to and including 8 bits) and explain overflow errors which may occur<br>How to convert positive denary whole numbers into 2-digit hexadecimal numbers and vice versa<br>How to convert from binary to hexadecimal equivalents and vice versa<br>Binary shifts<br>**Characters**<br>The use of binary codes to represent characters<br>The term 'character-set'<br>The relationship between the number of bits per character in a character set, and the number of characters which can be represented, e.g.:<br>• ASCII<br>• Unicode<br>Images<br>How an image is represented as a series of pixels, represented in binary<br>Metadata<br>The effect of colour depth and resolution on:<br>The quality of the image<br>The size of an image file<br>Sound<br>How sound can be sampled and stored in digital form<br>The effect of sample rate, duration and bit depth on:<br>The playback quality<br>The size of a sound file<br><br>The need for compression<br>Types of compression<br><br>**Title of unit: Designing, creating and refining algorithms (2.1.2)**<br>**By the end of the unit, students will have learned about:**<br>Identify the inputs, processes, and outputs for a problem<br>Structure diagrams<br>Create, interpret, correct, complete, and refine algorithms using: | **Title of unit: Identifying and preventing vulnerabilities (1.4.2)**<br>**By the end of the unit, students will have learned about:**<br>Common prevention methods such as anti-malware software, firewalls and user access levels.<br><br>**Title of unit: Operative systems (1.5.1)**<br>**By the end of the unit, students will have learned about:**<br>The purpose and functionality of operating systems:<br>• User interface<br>• Memory management and multitasking<br>• Peripheral management and drivers<br>• User management<br>• File management<br><br>**Title of unit: Testing (2.3.2)**<br>**By the end of the unit, students will have learned about:**<br>The purpose of testing<br>Types of testing<br>Identify syntax and logic errors<br>Selecting and using suitable test data<br>Refining algorithms<br><br>**Title of unit: Utility software (1.5.2)**<br>**By the end of the unit, students will have learned about:**<br>The purpose and functionality of utility software<br>Utility system software:<br>• Encryption software<br>• Defragmentation<br>• Data Compression<br><br>**Title of unit: Ethical, legal, cultural and environmental impact (1.6.1)**<br>**By the end of the unit, students will have learned about:**<br>Impacts of digital technology on wider society including:<br>• Ethical issues |

| | | |
|---|---|---|
| | • Pseudocode<br>• Flowcharts<br>• Reference language/high-level programming language<br>Identify common errors<br>Trace tables<br><br>**Title of unit: Computational thinking (2.1.1)**<br>**By the end of the unit, students will have learned about:**<br>Principles of computational thinking<br>• Abstraction<br>• Decomposition<br>• Algorithmic Thinking. | • Legal issues<br>• Cultural issues<br>• Environmental issues<br>• Privacy issues<br>Legislation relevant to Computer Science<br><br>**Title of unit: The Integrated Development Environment (IDE) (2.5.2)**<br>**By the end of the unit, students will have learned about:**<br>Common tools and facilities available in an integrated development environment (IDE) |
| **Spring term** | **Title of unit: Architecture of the CPU (1.1.1)**<br>**By the end of the unit, students will have learned about:**<br>The purpose of the CPU<br>The fetch-execute cycle<br>Common CPU components and their function and different type of computer architecture such as Von Neumann architecture<br><br>**Title of unit: Data Types (2.2.2)**<br>**By the end of the unit, students will have learned about:**<br>The use of data types such as Integer, Real and Boolean etc<br><br>**Title of unit: Embedding Systems (1.1.3)**<br>**By the end of the unit, students will have learned about:**<br>The purpose and characteristics of embedded systems<br>Examples of embedded systems<br><br>**Title of unit: Programming fundamentals (2.2.1)**<br>**By the end of the unit, students will have learned about:**<br>The use of variables, constants, operators, inputs, outputs and assignments<br>The use of the three basic programming constructs used to control the flow of a program including Sequence, Selection and Iteration<br>The common arithmetic operators | **Title of unit: Languages (2.5.1)**<br>**By the end of the unit, students will have learned about:**<br>Characteristics and purpose of different levels of programming language:<br>• High-level languages<br>• Low-level languages<br>The purpose of translators<br>The characteristics of a compiler and an interpreter<br><br>**Title of unit: Searching and sorting algorithms (2.1.3)**<br>**By the end of the unit, students will have learned about:**<br>Standard searching algorithms such as Binary and Linear Searches<br>Standard sorting algorithms e.g. Bubble sort, Merge sort and Insertion sort<br><br>**Programming Revision**<br><br>**Searching and Sorting Practical Programming skills**<br><br>**Theory Revision**<br><br>**Practical Programming Skills Revision** |

| | | |
|---|---|---|
| | The common Boolean operators AND, OR, NOT<br><br>**Title of unit: CPU performance (1.1.2)**<br>**By the end of the unit, students will have learned about:**<br>How common characteristics of CPUs affect their performance:<br>• Clock speed<br>• Cache size<br>• Number of Cores<br><br>**Title of unit: Primary Storage (1.2.1)**<br>**By the end of the unit, students will have learned about:**<br>The need for primary storage<br>The difference between RAM and ROM<br>The purpose of ROM in a computer system<br>The purpose of RAM in a computer system<br>Virtual memory | |
| **Summer term** | **Title of unit: Secondary Storage (1.2.2)**<br>**By the end of the unit, students will have learned about:**<br>The need for secondary storage<br>Common types of storage:<br>• Optical<br>• Magnetic<br>• Solid state<br>Suitable storage devices and storage media for a given application<br>The advantages and disadvantages of different storage devices and storage media relating to their characteristics.<br><br>**Title of unit: Additional programming techniques (2.2.3)**<br>**By the end of the unit, students will have learned about:**<br>The use of basic string manipulation<br>The use of basic file handling operations<br>The use of records to store data<br>The use of SQL to search for data<br>The use of arrays (or equivalent) when solving problems, including both one-dimensional (1D) and two-dimensional (2D) arrays | **Exams:**<br>2 exams: 1 hour 30 minutes<br>Each exam is worth 50% of the GCSE<br><br>Components:<br>Computer systems (01)<br>Computational thinking, algorithms and programming (02) |

| | | |
|---|---|---|
| | How to use sub programs (functions and procedures) to produce structured code<br>Random number generation<br><br>**Title of unit: Networks and topologies (1.3.1)**<br>**By the end of the unit, students will have learned about:**<br>Types of networks:<br>• LAN (Local Area Network)<br>• WAN (Wide Area Network)<br>Factors that affect the performance of networks<br>The different roles of computers in a client-server and a peer-to-peer network<br>The hardware needed to connect stand-alone computers into a Local Area Network<br>The Internet as a worldwide collection of computer networks:<br>• DNS (Domain Name Server)<br>• Hosting<br>• The Cloud<br>• Webservers and Clients<br>Star and Mesh network topologies<br>**Title of unit: Wired and wireless networks, protocols and layers (1.3.2)**<br>**By the end of the unit, students will have learned about:**<br>Modes of connection including, Wired, Ethernet, Wireless, Wi-Fi and Bluetooth<br>Encryption<br>IP addressing and MAC addressing<br>Standards<br>Common communication protocols<br>The concept of layers | |
| Evidence of learning | Learning will be evidenced through classwork, set assignments in theory and programming and through block tests as the student progress through the course. | |
| Links to prior learning | The OCR GCSE Computer Science curriculum builds upon the structured progression of knowledge and skills developed at Key Stages 2 and 3, | |

| | |
|---|---|
| | supporting a smooth transition and deepening understanding. At Key Stage 2, pupils are introduced to the fundamentals of programming, logical reasoning, and basic computer systems, laying the groundwork for computational thinking. These core principles are extended at Key Stage 3, where students engage with more formal programming languages (such as Python), understand key computational concepts like algorithms and data representation, and begin to explore how computer systems and networks function.<br><br>OCR's GCSE Computer Science course advances this foundation by challenging students to apply computational thinking in more complex contexts. The curriculum introduces advanced programming techniques, structured problem solving, and algorithm development. It also includes in-depth theoretical components such as system architecture, memory and storage, cybersecurity, and ethical, legal, and environmental issues in computing. These topics are clearly mapped to the KS3 National Curriculum, allowing learners to consolidate and extend their prior learning in a rigorous and engaging way. This ensures that students leave the course not only with practical skills but also with a robust theoretical understanding that prepares them for further study or careers in the digital sector. |
| Links to future learning | Computer Science A level – Bedford Sixth form<br>Advanced Computing – Bedford College<br>Level 3 ICT – Kimberley College |

**Reading in the curriculum (Literacy & Vocabulary)**

New vocabulary is introduced to students through key terms in each lesson. Throughout the curriculum we use a range of different reading resources to add depth and knowledge to students understand.

**Careers**

Software Developer, Cybersecurity Analyst, Web Developer, Game Developer, Data Analyst, IT Support Technician, Network Engineer, AI/Machine Learning Engineer, Systems Analyst, Data Scientist, Business Analyst, UX/UI Designer, IT Consultant, Ethical Hacker, SEO specialist, Database Administrator

**Safeguarding including safety in the curriculum**

Every year group has an e-safety unit each year to explicitly explore safety and safeguarding in relation to computing/technology

Holywell School Curriculum Overview

| |
|---|
| **Values across the curriculum**<br>Our curriculum supports the understanding of the school's core values throughout all of our units of work. |
| **Spirituality in the curriculum**<br>Our curriculum supports the spiritual development of students by creating an environment of curiosity, exploring interconnectedness, and fostering open-mindedness. By developing these key attributes, we hope to develop a sense of connection to something bigger than ourselves, to help students 'Live life in all its fulness', living our values; being the best we can be, in community. |
| **How we track your progress**<br>Linking to the progress descriptors all students' progress is tracked through the work they produce and contribute to in class, homework, end of unit assessments and in class assessments/quizzes. |
| **Parents/Carers can support their child by:**<br>These are the programs we use in school for coding: Ozaria, Scratch, Flowol, Microbits. Scratch and Ozaria are available to download free. |