**Computing Curriculum Overview**

| Year Group | Term 1 | Term 2 | Term 3 | Term 4 | Term 5 | Term 6 |
|---|---|---|---|---|---|---|
| **Year 1** | Technology Around Us | Digital Painting | Programming Robots | Grouping Data | Digital writing | Programming Animations |
| **Year 2** | IT Around Us | Digital Photography | Robot Algorithms | Pictograms | Digital Music | Programming Quizzes |
| **Year 3** | Connecting Computers | Stop-Frame Animation | Sequencing Sounds | Branching Databases | Desktop Publishing | Events and Actions in Programs |
| **Year 4** | The Internet | AI (Artificial Intelligence) | Repetition in Shapes | Logging Data | Photo Editing | Repetition in games |
| **Year 5** | Systems and Searching | Video Production | Selection in Physical Computing | Flat-File Databases | Vector Drawing | Selection in Physical Computing (Robots) |
| **Year 6** | Communication and Collaboration | Web Page Creation | Variables in Games | Introduction to Spreadsheets | 3D Modelling | Sensing |

**Year 7 Expectations**

Aims

The national curriculum for languages aims to ensure that all pupils:

- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
- are responsible, competent, confident and creative users of information and communication technology

Key stage 3:

Pupils should be taught to:

- design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems
- understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem
- use 2 or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions
- understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal]
- understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems
- understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits

- undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users
- create, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability
- understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct, and know how to report concerns