Computing

Whole-School Intent

At our school, the Computing curriculum is designed to equip pupils with the knowledge, skills and understanding needed to thrive in an increasingly digital world. In line with the National Curriculum for Computing, our intent is to ensure that all pupils become computational thinkers, responsible digital citizens and confident users of technology.

Our curriculum is informed by the **Teach Computing** curriculum and enhanced through **Barefoot Computing**, ensuring a research-informed, carefully sequenced and inclusive approach. Pupils develop a deep understanding of computer science, information technology and digital literacy, enabling them to use technology creatively, safely and effectively.

Curriculum Design Intent

Our Computing curriculum:

- Is ambitious and inclusive, enabling all pupils to succeed.
- Is coherently sequenced so that knowledge builds over time.
- Develops pupils' understanding of how computers work, not just how to use them.
- Provides opportunities for real-world application and purposeful outcomes.
- Explicitly teaches online safety as a core and recurring component.

The curriculum is structured around the three strands of the National Curriculum:

1. **Computer Science**
2. **Information Technology**
3. **Digital Literacy**

Knowledge Progression Intent

Pupils progressively develop secure knowledge of:

- Algorithms and programs, including sequencing, selection and repetition.
- How data is represented, stored and used.
- The fundamentals of computer systems and networks, including the internet.
- How digital content is created, organised and presented.
- How technology impacts individuals, communities and society.

Knowledge is revisited and deepened across year groups to support long-term retention, following the spiral model used in Teach Computing and Barefoot Computing.

Skills Progression Intent

Our curriculum ensures pupils learn to:

- Design, write and debug programs with increasing complexity.
- Use logical reasoning to predict and explain program behaviour.
- Create digital artefacts for specific purposes and audiences.
- Select and use a range of software and devices effectively.
- Evaluate and improve their digital work.

By the end of Key Stage 2, pupils are confident programmers who can independently solve problems using computational thinking.

PROGRESSION IN COMPUTING

Computational Thinking Intent

Across all year groups, pupils develop:

- Decomposition – breaking problems into manageable parts.
- Pattern recognition – identifying similarities and trends.
- Abstraction – focusing on important information.
- Algorithmic thinking – creating precise, logical steps.

These skills are embedded across the curriculum and reinforced through unplugged and digital activities, as promoted by Barefoot Computing.

Digital Literacy & Online Safety Intent

We are committed to developing pupils who:

- Use technology safely, respectfully and responsibly.
- Understand how to keep personal information private.
- Can identify and respond appropriately to online risks.
- Understand the impact of their digital footprint.
- Know how to seek help and report concerns.

Online safety is taught explicitly and progressively and revisited regularly to reflect the evolving digital landscape.

Inclusion and Accessibility Intent

Our Computing curriculum:

- Is accessible to all learners, including pupils with SEND.
- Makes effective use of unplugged activities to remove barriers to learning.
- Provides scaffolds, models and adaptive teaching strategies.
- Values collaboration and discussion as key learning tools.
- Ensures all pupils develop confidence and resilience.

The use of Teach Computing and Barefoot resources supports inclusive practice through clear progression, vocabulary development and multiple entry points.

Preparation for the Next Stage

By the end of Key Stage 2, pupils:

- Have a secure understanding of algorithms, programming and data.
- Can explain how computer systems and networks operate.
- Create purposeful digital content independently.
- Use technology safely and responsibly.
- Are well prepared for Key Stage 3 Computing and future digital learning.

Reception – Barefoot Computing
Technology Around Us
• Identify different types of technology • Explain what technology is used for • Begin to use devices appropriately with adult support
Patterns & Logic
• Identify patterns in everyday activities • Predict what comes next in a sequence • Explain their thinking using simple language
All About Algorithms
• Follow a sequence of instructions • Create simple step-by-step instructions for a familiar task • Recognise when instructions are in the wrong order • Begin to debug by spotting mistakes
Online Safety Foundations
• Understand that some things are private • Know to ask an adult for help when unsure • Recognise trusted adults
By the end of Reception
• Follow and give simple instructions • Understand that instructions must be clear and in order • Begin to recognise patterns and repetition • Use technology with support and care • Talk about what technology is used for at home and school

YEAR 1 – Teach Computing
Technology Around Us
• Identify and name common technology in the classroom/home. • Describe what a computer is used for. • Recognise how technology helps us in everyday life. • Explore simple physical components (e.g., mouse/keyboard). • Use technology with increasing confidence.
Digital Painting
• Open and use digital painting tools. • Choose colours and brushes to create digital art. • Change brush size and colour when prompted. • Talk about their digital artwork. • Save and retrieve a simple picture.
Moving a Robot
• Understand what an instruction is. • Give step by step instructions to move a robot. • Predict what the robot will do next. • Test and debug simple sequences. • Talk about successful and unsuccessful instructions.
Grouping Data
• Collect items based on a simple category. • Organise items into groups. • Explain why items are grouped together. • Enter simple data into a computer/table. • Talk about patterns in the data.
Digital Writing
• Open a text editor tool. • Type simple text accurately. • Change the look of text (e.g., bold, colour). • Use basic editing functions (delete/insert). • Save and reopen work.
Programming Animations
• Open a simple animation tool. • Add characters/sprites to a workspace. • Use blocks to animate a sprite. • Predict how sequences create movement. • Debug when the sprite doesn’t behave as expected.

YEAR 2 - Teach Computing
Information Technology Around Us
• Recognise different types of technology around us. • Explain how technology helps us with information. • Identify parts of a computer device. • Use a mouse and keyboard confidently. • Show ways to use technology safely.
Digital Photography
• Use a device to take a clear photo. • Select and crop an image. • Talk about good composition. • Improve images with simple edits. • Use photos for a purpose (e.g., class display).
Robot Algorithms
• Create step by step algorithms on paper. • Predict outcomes of sequences. • Implement algorithms with digital devices. • Debug when sequences don't work. • Explain why a program works.
Pictograms
• Collect data using tally charts. • Enter data to create a pictogram. • Label parts of a chart correctly. • Interpret patterns in data. • Talk about results using simple vocabulary.
Digital Music
• Open a digital music tool. • Select sounds to create a simple rhythm. • Arrange sounds into a pattern. • Change tempo/volume where possible. • Explain choices for their composition
Programming Quizzes
• Open a quiz programming tool. • Design questions and correct answers. • Use sequences and logic for quiz flow. • Test and debug the quiz. • Share and talk about their quiz.

YEAR 3 - Teach Computing
Computing systems and networks – Connecting computers
• Identify components of a network. • Explain how computers connect to share information. • Explore how devices communicate. • Recognise the internet as a network. • Discuss everyday uses of networks.
Creating media – Stop frame animation
• Plan an animation sequence. • Capture frames to show movement. • Edit timing between frames. • Add sound where appropriate. • Evaluate the final animation.
Programming A – Sequencing sounds
• Open a sequencing tool. • Arrange sound clips in order. • Predict sequence outcomes before running. • Debug unexpected sound order. • Describe how sequencing produces music.
Data and information – Branching databases
• Choose attributes to classify objects. • Enter information into a branching database. • Use yes/no questions to sort items. • Explain choices in classification. • Draw conclusions from the database.
Creating media – Desktop publishing
• Open a desktop publishing tool. • Insert text and images. • Format a page for a purpose. • Choose fonts/layout appropriately. • Evaluate design for clarity.
Programming B – Events and actions in programs
• Add event blocks to code. • Understand event triggers. • Predict event outcomes. • Test interactions in a program. • Refine code for smoother events.

YEAR 4 - Teach Computing
Computing systems and networks – The internet
• Explain what the internet is. • Identify services that use the internet. • Describe how information travels. • Explore everyday internet use. • Discuss simple internet safety points.
Creating media – Audio production
• Record audio using a digital tool. • Trim and edit sound clips. • Add effects where appropriate. • Arrange clips to create a sequence. • Review work and refine choices.
Programming A – Repetition in shapes
• Use loops to repeat code. • Predict patterns created by repetition. • Debug looped code. • Change parameters (e.g., angle/size). • Explain how loops simplify tasks.
Data and information – Data logging
• Collect data using sensors. • Import data to software. • Create graphs or tables. • Interpret trends in data. • Evaluate data quality.
Creating media – Photo editing
• Open and edit photos. • Adjust brightness/contrast. • Crop and enhance images. • Use selection tools. • Discuss editing choices.
Programming B – Repetition in games
• Add loops to games. • Predict effects of repetition. • Test game behaviour. • Debug where needed. • Improve game mechanics.

YEAR 5 - Teach Computing
Computing systems and networks – Sharing information
• Describe how information is shared digitally. • Explain internet protocols (e.g., upload/download). • Recognise services that share data. • Discuss online communication tools. • Identify simple data security principles.
Creating media – Video production
• Plan a short video sequence. • Capture clips with purpose. • Edit video timeline. • Add titles/transitions. • Review and improve final edit.
Programming A – Selection in physical computing
• Connect inputs/outputs on a physical device. • Use conditions (if/then) in code. • Predict behaviour of conditions. • Debug conditional programs. • Explain how selection changes output.
Data and information – Flat file databases
• Create a database file. • Add records with multiple fields. • Sort and search data. • Filter information. • Draw conclusions from results.
Creating media – Introduction to vector graphics
• Create shapes using vector tools. • Change stroke/fill properties. • Manipulate objects (rotate/resize). • Group objects. • Evaluate design choices.
Programming B – Selection in quizzes
• Add conditionals to a quiz program. • Predict outcomes for answers. • Test and fix logic errors. • Use variables where needed. • Refine feedback for users.

YEAR 6 - Teach Computing
Computing systems and networks – Internet communication
• Explain how communication happens online. • Describe networks that connect people. • Identify tools for online collaboration. • Discuss reliable vs unreliable sources. • Recognise ways to keep communication secure.
Creating media – Web page creation
• Plan a simple web page structure. • Add text and images using a web editor. • Use links between pages. • Choose appropriate styles. • Test and refine the page.
Programming A – Variables in games
• Use variables in code. • Predict how variables affect game state. • Change variable values through interaction. • Debug errors involving variables. • Discuss how variables improve programs.
Data and information – Introduction to spreadsheets
• Enter data into a spreadsheet. • Use formulas for calculations. • Create charts from data. • Format cells for clarity. • Interpret results to solve problems.
Creating media – 3D modelling
• Navigate a 3D modelling tool. • Create basic 3D shapes. • Adjust size/position of objects. • Combine shapes to form a model. • Evaluate the model against a brief.
Programming B – Sensing movement
• Use sensors to detect movement. • Connect sensor input to code. • Predict sensor responses. • Test and debug sensor programs. • Explain how sensors trigger actions.