Oak Hill Church of England

Primary School

'Jesus said, "I have come that you may have life in all its fullness'

(John 10:10)

Faith- Values- Action

Computing Curriculum

Intent

Computing at Oakhill Primary School intends to develop 'thinkers of the future' through a modern, ambitious and relevant education in computing. We want to equip pupils to use computational thinking and creativity that will enable them to become active participants in the digital world. It is important to us that the children understand how to use the ever-changing technology to express themselves, as tools for learning and as a means to drive their generation forward into the future.

Whilst ensuring they understand the advantages and disadvantages associated with online experiences, we want children to develop as respectful, responsible and confident users of technology, aware of measures that can be taken to keep themselves and others safe online.

Our aim is to provide a computing curriculum that is designed to balance acquiring a broad and deep knowledge alongside opportunities to apply skills in various digital contexts. Beyond teaching computing discreetly, we will give pupils the opportunity to apply and develop what they have learnt across wider learning in the curriculum.

Implementation

At Oak Hill we use the 'Teach Computing' Curriculum. It uses the National Centre for Computing Education's computing taxonomy to ensure comprehensive coverage of the subject. All learning outcomes can be described through a high-level taxonomy of ten strands, ordered alphabetically as follows:

- Algorithms – be able to comprehend, design, create, and evaluate algorithms.
- Computer Networks – understand how networks can be used to retrieve and share information, and how they come with associated risks.
- Computer Systems – Understand what a computer is, and how its constituent parts function together as a whole
- Creating Media – Select and create a range of media including text, images, sounds and video
- Data and Information – Understand how data is stored, organised and used to represent real-world artefacts and scenarios.
- Design and Development – Understand the activities involved in planning, creating and evaluating computer artefacts
- Effective Use of Tools – use software tools to support computing work
- Impact of Technology – Understand how individual, systems and society as a whole interact with computer systems.
- Programming – Create software to allow computers to solve problems

- Safety and Security – Understand risks when using technology, and how to protect individuals and systems.

For the teaching of Computing we follow 12 pedagogical principles;

- **Lead with concepts** – Support pupils in the acquisition of knowledge, through the use of key concepts, terms and vocabulary, providing opportunities to build a shared and consistent understanding. Glossaries, concept maps and displays along with regular recall and revision can support this approach.
- **Unplug, Unpack, Repack** – teach new concepts by first unpacking complex terms and ideas, exploring these ideas in unplugged and familiar contexts, then repacking this new understanding into the original concept. This approach (semantic waves) can help pupils develop a secure understanding of complex concepts.
- **Create Projects** – Use project based learning opportunities to provide pupils with the opportunity to apply and consolidate their knowledge and understanding. Design is an important, often overlooked aspect of computing. Pupils can consider how to develop an artefact for a particular user or function, and evaluate it against a set of criteria.
- **Challenge Misconceptions** – Use formative questioning to uncover misconceptions and adapt teaching to address them as they occur. Awareness of common misconceptions alongside discussion, concept mapping, peer instruction or simple quizzes can help identify areas of confusion.
- **Structure Lessons** – Use supportive framework when planning lessons such as PRIMM (Predict, Run, Investigate, Modify, Make) and Use-Modify-Create. These frameworks are based on research and ensure that differentiation can be built in at various stages of the lesson.
- **Work Together** – Encourage collaboration, specifically using pair programming and peer instruction, and also structured group tasks. Working together stimulates classroom dialogue, articulation of concepts and development of shared understanding.
- **Model Everything** – Model processes and practices – everything from debugging code to binary number conversions – using techniques such as worked examples and live coding. Modelling is particularly beneficial to novices, providing scaffolding that can be gradually taken away.
- **Add Variety** – Provide activities with different levels of direction, scaffolding and support that promote active learning, ranging from highly structured to more exploratory tasks. Adapting your instructions to suit different objectives will keep all pupils engaged and encourage greater independence.
- **Make Concrete** – Bring abstract concepts to life with real-world, contextual examples and a focus on interdependencies with other curriculum subjects. This can be achieved through the use of unplugged activities, proposing analogies, storytelling around concepts, and finding examples of the concepts in pupils' lives.
- **Read and Explore Code First** – When teaching programming, focus first on code reading activities, bore code writing. With both block-based and text-based programming encourage pupils to review and interpret blocks of code. Research has shown that being able to read, trace and explain code augments pupils' ability to write code.
- **Get Hands on** – Use physical computing and making activities that offer tactile and sensory experiences to enhance learning. Combining electronics and programming with arts and crafts (especially through exploratory projects) provide pupils with a creative, engaging context to explore and apply computing concepts.
- **Foster Program Comprehension –** Use a variety of activities to consolidate knowledge and understanding of the function and structure of programs, including debugging, tracing and Parson's Problems. Regular comprehension activities will help secure understanding and connection with new knowledge.

The units for Key Stages 1 and 2 are based on a spiral curriculum. This means that each of the themes is revisited regularly (at least once in each year group) and pupils revisit each theme through a new unit that consolidates and builds on prior learning within a theme.

In Years 1-6 we operate a two-year cycle.

<u>EYFS</u>

We have based our computing approach for the EYFS on play-based, use of computer devices and non-technology devices in activities that focus on building children's listening skills, curiosity and creativity, communication and problem solving.

Technology in the Early Years at Oak Hill will include opportunities for pupils to:

- take a photograph with a camera or tablet
- search for information on the internet
- play games on the interactive whiteboard
- explore an old typewriter or other mechanical toys
- use a Beebot
- watch a video clip
- listen to music

Allowing children the opportunity to explore technology in this carefree and often child-led way, means that not only will they develop a familiarity with equipment and vocabulary but they will have a strong start in Key Stage 1 Computing and all that it demands.

Impact

The implementation of this curriculum ensures that when children leave Oakhill Primary School, they are competent and safe users of ICT with an understanding of how technology works. They will have developed skills to express themselves and be creative in using digital media and be equipped to apply their skills in Computing to different challenges going forward.