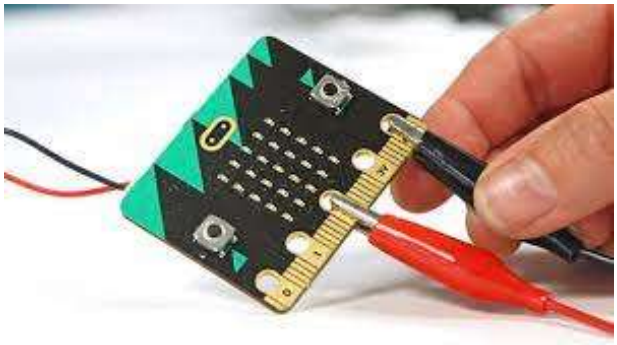
COMPUTING

AT

ALMOND HILL JUNIOR SCHOOL

2023-2024



Computing is not about computers anymore. It is about living – Nicholos Negroponte.

Reviewed July 2023

Intent

At Almond Hill we aim to equip our children with computational skills and knowledge that enable them to become 'lifelong learners'. We teach, encourage and model positive learning behaviours to promote responsible, resilient individuals that are ready for the technological future.

Computing **By the end of KS2 children can:**

- use computational thinking and creativity on a range of programming tasks
- understand how digital systems work
- express themselves with information communication technology
- be ready to build KS3 skills on a solid foundation, preparing them for the digital world

Online Safety **By the end of KS2 children can:**

- Live knowledgeably, responsibly and safely in a digital world.

[Aspects explored: Self-image, online reputation, online bullying, managing online information, health, wellbeing and lifestyles, privacy and security, copyright and ownership]

Implementation

At Almond Hill we implement a 6-lesson unit per term which follows an adapted version of '**Teach Computing**', building on prior knowledge from KS1 where possible. The aims of the Teach Computing Curriculum are as follows:

- Reduce teacher workload
- Show the breadth and depth of the computing curriculum, particularly beyond programming!
- Demonstrate how computing can be taught well, based on research
- Highlight areas for subject knowledge and pedagogy enhancement through training The Teach Computing Curriculum resources.

Knowledge organisation The Teach Computing Curriculum uses the National Centre for Computing Education's computing taxonomy to ensure comprehensive coverage of the subject. This has been developed through a thorough review of the KS1–4 computing programme of study. 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 computing artefacts ■ Effective use of tools — Use software tools to support computing work ■ Impact of technology — Understand how individuals, 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

- The red areas are not covered in the termly units but may be in cross curricular lessons.

Spiral curriculum (sticky learning) 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 that theme. This style of curriculum design reduces the amount of knowledge lost through forgetting, as topics are revisited yearly. It also ensures that connections are made even if different teachers are teaching the units within a theme in consecutive years.

Physical computing The Teach Computing Curriculum acknowledges that physical computing plays an important role in modern pedagogical approaches in computing, both as a tool to engage pupils and as a strategy to develop pupils' understanding in more creative ways. Additionally, physical computing supports and engages a ***diverse range of pupils*** in tangible and challenging tasks.

- Micro:bits are used in Y5 and Y6. Beebots may be used to support SEND in the lower school

Outdoor learning lessons are prepared and taught. Y4 – Re-enacting the Internet and how information is transferred outdoors on the playgrounds. Y5-Gathering light level data with the Micro:bits around the school grounds. Y6-Gathering steps counted data (with class made Micro:bit counters) around the school grounds.

Anti racism and inclusivity in computing, this is being promoted through lessons teaching children to be critical and the questioning of information online to see how it can be presented with western bias. Also, a display promotes diversity in the field (found in the Suite).

Teach Computing Online Safety The unit overviews for each unit show the links between the content of the lessons and the national curriculum and Education for a Connected World framework (ncce.io/efacw). These references have been provided to show where aspects relating to online safety, or digital citizenship, are covered within the Teach Computing Curriculum. Not all of the objectives in the Education for a Connected World framework are covered in the Teach Computing Curriculum, as some are better suited to personal, social, health, and economic (PSHE) education; spiritual, moral, social, and cultural (SMSC) development; and citizenship. However, the coverage required for the computing national curriculum is provided.

All classrooms have interactive whiteboards to enhance children's learning. We have a computing suite, class sets of iPads, physical systems (class sets of Micro:bits), Bee-Bots, robotics kits (cars and robot faces) Log Box data loggers and digital cameras. We use SaaS (Software as a Service) to teach the children to understand and navigate the curriculum. All children have a Microsoft 365 login and password which they use to access a shared student area. All children follow the Almond Hill Online Safety Rules and have agreed to our acceptable use policy. We update our website regularly with online safety information and have had guest speakers in to discuss this area with both children, parents and staff. We have used data from an online safety surveys to create content for our website. Termly Online Safety meetings take place with teaching staff, one governor and students to discuss current online safety issues and promote the safe responsible use of technology.

SEND

To ensure that the Computing curriculum is inclusive for children with SEND lessons may need to be adapted to provide appropriate provision. This could be in the form of any of the following:

- Adapted tasks
- Adapted resources / equipment
- Additional support

Topics Across Almond Hill

	Computing Systems and Networks	Programming A	Programming B
Year 3	Connecting Computers	Sequencing Sounds	Events & actions in programs
Year 4	The Internet	Repetition in Shapes	Repetition in Games
Year 5	Systems & Searching	Data Handling with Micro:bits Physical Computing	Selection in Quizzes
Year 6	Communication & Collaboration	Variables in Games	Sensing with Micro:bits

Curriculum Development

Currently, we are focusing on computing systems and networks as well as programming across the school with the spiral approach. Data and information along with creating media may be covered in cross-curricular lessons and clubs. All year groups now have 1 years' experience teaching the computing systems and networks as well as programming units and have improved subject knowledge/CPD to promote good teaching and learning. Further work on using formative and summative assessments as well as end tasks is underway to improve the accuracy of assessment and allow the subject leader to monitor the impact of the curriculum.

Progression of Skills in Computing - Years 3-6

Progression across key stages

All learning objectives have been mapped to the National Centre for Computing Education's taxonomy of ten strands, which ensures that units build on each other from one key stage to the next.

Progression across year groups

Within the Teach Computing Curriculum, every year group learns through units within the same four themes, which combine the ten strands of the National Centre for Computing Education's taxonomy (see table, right).

This approach allows us to use the spiral curriculum approach (see the 'Spiral curriculum' section for more information) to progress skills and concepts from one year group to the next.

Primary themes	Computing systems and networks	Programming	Data and information	Creating media
Taxonomy strands	Computer systems Computer networks	Programming Algorithms Design and development	Data and information	Creating media Design and development
	Effective use of tools			
	Impact of technology			
	Safety and security			

Pedagogy

Computing is a broad discipline, and computing teachers require a range of strategies to deliver effective lessons to their pupils. The National Centre for Computing Education's pedagogical approach consists of 12 key principles underpinned by research: each principle has been shown to contribute to effective teaching and learning in computing.

It is recommended that computing teachers use their professional judgement to review, select, and apply relevant strategies for their pupils.

These 12 principles are embodied by the Teach Computing Curriculum, and examples of their application can be found throughout the units of work at every key stage. Beyond delivering these units, you can learn more about these principles and related strategies in the National Centre for Computing Education pedagogy toolkit (ncce.io/pedagogy).

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.

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.

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) provides pupils with a creative, engaging context to explore and apply computing concepts.

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, called 'semantic waves', can help pupils develop a secure understanding of complex concepts.

Model everything

Model processes or 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.

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 build connections with new knowledge.

Create projects

Use project-based learning activities 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.

Add variety

Provide activities with different levels of direction, scaffolding, and support that promote learning, ranging from highly structured to more exploratory tasks. Adapting your instruction to suit different objectives will help keep all pupils engaged and encourage greater independence.

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.

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.

Structure lessons

Use supportive frameworks 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.

Read and explore code first

When teaching programming, focus first on code 'reading' activities, before 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.

Assessment

Formative assessment

Every lesson includes formative assessment opportunities for teachers to use. These opportunities are listed in the lesson plan and are included to ensure that misconceptions are recognised and addressed if they occur. They vary from teacher observation or questioning, to marked activities.

These assessments are vital to ensure that teachers are adapting their teaching to suit the needs of the pupils that they are working with, and you are encouraged to change parts of the lesson, such as how much time you spend on a specific activity, in response to these assessments.

The learning objective and success criteria are introduced in the slides at the beginning of every lesson. At the end of every lesson, pupils are invited to assess how well they feel they have met the learning objective using thumbs up, thumbs sideways, or thumbs down. This gives pupils a reminder of the content that has been covered, as well as a chance to reflect. It is also a chance for teachers to see how confident the class is feeling so that they can make changes to subsequent lessons accordingly.

Summative assessment

Every unit includes an optional summative assessment framework in the form of either a multiple choice quiz (MCQ) or a rubric. All units are designed to cover both skills and concepts from across the computing national curriculum. Units that focus more on conceptual development include an MCQ. Units that focus more on skills development end with a project and include a rubric. However, within the 'Programming' units, the assessment framework (MCQ or rubric) has been selected on a best-fit basis.

Multiple choice quiz (MCQ)

Each of the MCQ questions has been carefully chosen to represent learning that should have been achieved within the unit. In writing the MCQs, we have followed the diagnostic assessment approach to ensure that the assessment of the unit is useful to determine both how well pupils have understood the content, and what pupils have misunderstood, if they have not achieved as expected.

Each MCQ includes an answer sheet that highlights the misconceptions that pupils may have if they have chosen a wrong answer. This ensures that teachers know which areas to return to in later units.

Rubric

The rubric is a tool to help teachers assess project-based work. Each rubric covers the application of skills that have been directly taught across the unit, and highlights to teachers whether the pupil is approaching (emerging), achieving (expected), or exceeding the expectations for their age group. It allows teachers to assess projects that pupils have created, focussing on the appropriate application of computing skills and concepts.

Pedagogically, we want to ensure that we are assessing pupils' understanding of computing concepts and skills, as opposed to their reading and writing skills. This has been carefully considered both in how MCQs have been written (considerations such as the language used, the cultural experiences referenced, etc) and in the skills expected to be demonstrated in the rubric.