St Osmund's Computing Curriculum Framework

The St Osmund's computing curriculum framework at a minimum, fulfils the purpose and aims of the computing national curriculum. The curriculum is designed in unit blocks of varying lengths that run in 50 minutes to an hour sessions. Each unit has a description of the learning outcomes that all pupils are expected to achieve by the end of that sequence of learning, these amount to our age related expectations. The units form the basis of what pupils learn in class, and will be adapted by class teachers who will tailor their own lesson plans based on assessments of pupil learning and appropriate teaching methods.

Curriculum Intent

The intention of the Computing Curriculum is to inspire learners and provide a creative, engaging and inclusive curriculum for all, as well as developing a narrative of what computer science is.

The intention is that every pupil in key stage 2 and 3 has the opportunity to learn material that is recognisably 'Computer Science' and develop the key thinking skills described as 'Computational Thinking', thereby preparing students to confidently access and use all form of technology in the future. St Osmund's believes in providing a rich, deep learning experience that balances all the aspects of computing; computer science, information technology, and digital literacy, providing pupils with a varied curriculum.

The curriculum plans for sequences of learning, that provide opportunities to develop mastery in computational thinking concepts and build on these from KS2 to KS3. It is recognised that computational thinking can be developed in any of the strands of the curriculum.

The School values of Hope, Community, Respect and Love are also explored and promoted throughout the curriculum, especially within the modules recognising the importance of using technology in a safe and positive way and so E-safety is threaded throughout the curriculum. Explicit cross curriculum links are made at key stage 2; at key stage 3 it is recognised that the ways of thinking developed in computing are useful and applicable across the curriculum; similarly how logical and abstract thinking are developed in maths or science will compliment how thinking develops in computing.

Spiralled Curriculum

The curriculum has been designed to build on pupils' prior knowledge. Each topic continues to be developed throughout the four years at St Osmund's, developing pupils' understanding and promoting good progress. Below shows the progression some of the key topics within the curriculum.

Computer Science

		Binary		
		Binary Numbers	**Other Binary Representations**	**Boolean Logic**
Ks2	Year 5			
	Year 6	Converting blocks into binary digits and vice versa		Boolean Operators when using search engines
KS3	Year 7	Converting between binary and denary	Binary Encryption	Introduction to Logic Gates
		Writing Names using ASCII	Introduction to bitmap images using binary digits	
	Year 8	Binary Addition	Binary images	
			Sound Waves	
			2 bit images binary pictures	

Programming

		Programming				
		Logo	**Scratch**	**Micro:Bits**	**HTML**	**Python**
Ks2	Year 5	Iteration, using repeats	Use movement, change costumes, speech	Iteration, loops and repeats		
		Creating Procedures	Iteration - Using repeats and loops			
			Introduction of subroutines			
			Use of variables to create algorithm art			
	Year 6		Use of selection - If and Else	Use of selection - If and Else		
			Use of multiple variables	Use of variables		
			Introduction to lists			

KS3	Year 7			Use of variables to create outputs	Using html tags	Using the Python Editor to look at block code in Python in Microbits
				Representing code in a flow chart	Adding colour, hyperlinks and pictures using html	Changing parts of the Python code to make changes in their block code
	Year 8			Radio Function - sending and receiving messages		Create Inputs and Outputs
				Programming solutions to real life problems		Create variables for use within the code
						Use of selection - If, Elif and Else
						Use of operators
						Random lists

Digital Literacy

		E Safety
Ks2	Year 5	Creating a Positive Digital Footprint
		Identifying Phishing Emails and Scams
		Protecting Information online. Passwords and Sharing information
		Cyberbullying - what it is, how to deal with it and prevention
	Year 6	Consent and Social Media. Terms and Conditions and gathering information
		Communicating safely online
		Promoting a positive digital footprint
		Protecting your information online – social media
KS3	Year 7	Identifying Cyberbullying and ways to deal with it
		Communicating safely online
		Social media and mental health
		Creators Rights – copyright
	Year 8	Digital Footprints and their future implications
		Consequences of a Digital Footprint
		Protecting yourself online/Sexting
		Identifying unsafe behaviour online from others

The Computational Thinker: Concepts & Approaches

Concepts

- **Logic** — predicting & analysing
- **Algorithms** — making steps & rules
- **Decomposition** — breaking down into parts
- **Patterns** — spotting & using similarities
- **Abstraction** — removing unnecessary detail
- **Evaluation** — making judgement

Approaches

- **Tinkering** — experimenting & playing
- **Creating** — designing & making
- **Debugging** — finding & fixing errors
- **Persevering** — keeping going
- **Collaborating** — working together

Barefoot Computing

COMPUTING AT SCHOOL

www.barefootcas.org.uk

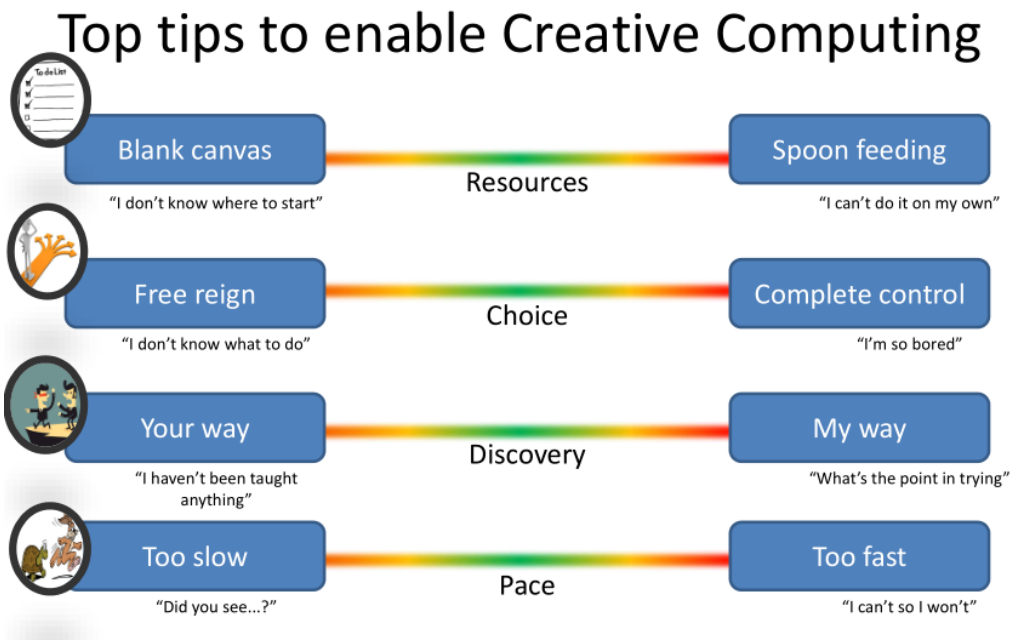
Barefoot would like to acknowledge the work of Julia Briggs and the eLIM team at Somerset County Council for their contribution to this poster.

Section 2: Pedagogical approaches and implementation

There are many different approaches that can be taken to deliver the curriculum content. Below are some of the main teaching methods we use in the delivery of the curriculum and their rational to support teacher lesson planning.

The variety of resources available necessitates a consideration of the pedagogy that each of the resources utilises and how the pedagogy can be applied to your learners to ensure that each pupil achieves regardless of gender or need.

The NC focus is on computational thinking, creativity and problems solving.



Knowledge Organisers

Each module has an associated knowledge organiser that contains key vocabulary and important features of the programs they will be using. These will be easily accessible for pupils in their folders as well as being displayed on the school website. Pupils will be able to reference these throughout the lessons during questioning and 'Do it now' tasks to support their learning.

Do it now tasks

At the start of each lesson, pupils will be given some 'Do it now' questions that review past learning. By repeatedly revisiting concepts, this will help pupils commit learning to their long term memory and aid their retrieval of key information.

Differentiation

In class, all pupils are exposed to challenging content, the teacher will adapt, scaffold or extend the curriculum depending on the need of individual pupils. Questioning also adheres to the school 'No hands Up' policy and so is targeted to each individual by the teacher to consolidate and extend depending on the need of a pupil.

Previously high attaining pupils and potential high attaining pupils

Within each activity, high attaining pupils are encouraged to showcase their creativity and independent thinking. To obtain a band 5, pupils need to reference what new skills and individual content they have included in their project on their assessment grid. This content could be achieved by using hint sheets to explore new material and concepts, adapt what they have been taught to give a different outcome or include their own independent learning.

SEND and Pupil Premium

All teachers are aware of the SEND and Pupil Premium pupils within their classes. Teachers should have read any learning passports and become acquainted with how best to support that individual. Teachers will have also spent time devising a seating plan that best supports the learner's needs and identify key pairings for discussion and paired work.

Each teacher will scaffold and adapt the lesson content to best suit the individuals' needs so that every pupils can achieve their full potential and access the Computing Curriculum.

Unplugged activities

Unplugged activities are useful to focus the learning on just the intended understanding by reducing cognitive load. Several unplugged activities for developing algorithmic thinking are included in key stage 2.

Learning to Program

Introductory programming can be seen as equivalent to a language course or like learning a musical instrument; it involves practice and making lots of mistakes. Perseverance is required to develop fluency. By the end of year 8, we would like students to be able to independently code e.g. given a challenge and write the code without any input. That means that they have to have developed skills and knowledge that they are confident enough to transfer to a new problem.

Students do need to memorise the syntax of some key programming constructs e.g. inputs/outputs, if statements, and for loops. This decreases cognitive load and enables students to practices programming more fluently. Typing activities and debugging challenges (syntax and logic errors) support the development of this fluency.

Pair programming

Pair programming is quite simple: pairs of programming students work collaboratively on a shared project and take turns acting in two roles: the Driver controls the computing device and writes the code, while the Navigator provides direction, spots errors, and thinks ahead to the next part of the project. Throughout a programming session, the pair regularly swaps roles, so that each individual performs each role several times while working on the same programming task.

Research has conclusively shown the efficacy of pair programming, so it is a recommended practice for computing teachers in schools.

Although the principle is simple, pair programming, when implemented well, has positive effects on the performance and attitude of learners, as well as added benefits for the teacher:

- One of the benefits demonstrated by most studies is learners' increased engagement in and enjoyment of the programming task.
- There is also evidence that, compared with solo programming, pair programming boosts learners' self-confidence and interest in the subject material. This may explain the increased student retention through introductory programming courses and into follow-up courses.

- Studies have found that pair programming can lead to learning gains, most readily evidenced through improved code quality and programs with fewer common mistakes.
- Pair programming has been shown to reduce cognitive load by effectively sharing any additional load between the pair. This "distributed cognition" means that each role takes on different types of cognitive tasks.
- Opportunities to collaborate are valuable in and of themselves: students get to practice a transferable skill that is important for their future careers and especially sought after in the software development sector.
- There are also added benefits for the teacher: firstly, each learner has a partner that they can direct low-level and practical questions to, freeing up the teacher to focus on the paired work going on.

https://www.youtube.com/watch?v=vgkahOzFH2Q

PRIMM: A structured approach to teaching programming

PRIMM is one approach that has been researched as a way to help teachers to structure lessons in programming.

The following text is taken from:

https://blogs.kcl.ac.uk/cser/2017/09/01/primm-a-structured-approach-to-teaching-programming/

It is based on research into the learning of programming but combines different areas. Our overall interpretation of the research is that teaching programming requires a blended approach using a range of strategies.

PRIMM stands for the following:

- Predict
- Run
- Investigate
- Modify
- Make

You may not be able to go through all stages in one lesson and may even focus on one stage more than another but remembering PRIMM gives you a way of labelling what you are doing when you are teaching programming.

Being PRIMM-like in your lessons

The following table gives some examples of activities that can be used within a PRIMM-like approach. We would recommend that all these activities are done in pairs.

Stage	Activity	Why
Predict	In pairs look at a piece of code printed out or on the whiteboard and ask students what they think it will do. They can write down their prediction or discuss in a group class discussion. Live coding is quite useful here or the code can be already prepared.	This activity encourages students to look for clues in the program that suggest what its function is.

Run	Download a piece of code from a shared area and check against your prediction (don't copy in the code).	Having code that is provided has many benefits – it moves the weight of ownership of any errors from the student to the teacher, increasing confidence, and it also means that time is not spent copying in code, which can be a challenging exercise to students who struggle with literacy at any level.
Investigate	There are lots of different activities you can do at this stage: trace through the code, comment the code, answer questions about it, label particular concepts, highlight it, draw the flow of control, etc. Again pair work helps to encourage discussion about the nitty gritty of the program	It takes many activities of this type, repeated in different forms in different lessons, for students to start to understand the underlying concepts in a secure way. We may tend to think that writing one selection statement correctly means that students have a good understanding of selection but really "getting" this takes some time.
Modify	Given a working piece of code, students are challenged to add a variety of modifications, starting very simply and having a series of exercises increasing in difficulty with larger modifications.	The transfer of ownership moves from the code being "not mine" to "partly mine" as students gain confidence by extending the function of the code. This activity obviously provides the scaffolding that students need to add small snippets of code and see their effect within a bigger program
Make	Once students are confident in modifying the program that you have created, they can create their own program from scratch, which has similarities with the previous program but that they can design themselves	Design of a new program is an important skill, and should start with planning and trying to construct a suitable algorithm. This is difficult, but does give students an opportunity to be creative and have the satisfaction of making their own program

Scaffolding learning

The reason we designed the PRIMM approach is because we recognised, from our own teaching and others' research that learners need much more support to understand programming concepts. This means that strategies are needed that scaffold the learning and also promote discussion about what is going on in the programs. Working in pairs can also provide students with mutual support which is a form of scaffolding, as well as promoting discussion and an articulation of the problems, around the code.

Spectrum of techniques

Importantly, copying code in from a worksheet does not give us any indication that a student understands what they are typing in, and the processes involved in copying are high on cognitive load. At the other extreme, tinkering or exploring without guidance can be fun, but without an understanding of the concepts can lead to frustration. Other strategies in between these extremes include running and testing code, predicting outcomes of code, tracing code, annotation of code, modifying code, etc. and PRIMM is one approach that combines these.

Physical Computing

Physical computing is an educational context for teaching computer science by using hardware and software to create tangible constructs. Using simple inputs, physical computing takes data from the physical world and manipulates it digitally to create outputs. This is included to increase student engagement by getting pupils interacting with hardware and experience different computer systems.

Homework

Pupils will be set one to two pieces of homework each term that is linked to the topic they are currently studying. This will then be marked in class, offering immediate feedback.

Marking and Feedback

Within computing lessons, pupils can achieve very different but relevant outcomes. Therefore, pupils' predominant source of feedback is immediate, individually-tailored verbal feedback provided by the Teacher. However, alongside this, pupils may be offered alternative forms of feedback to enhance their own learning and progress. These are:

- Written feedback – pupils are given notes on their own piece of work and their next steps in order to improve their work and make progress such as, when programming in Python adding # within their code to add notes specifically for them.

```
name = input What is your name?
print("Hello " + name)

#you are missing 2 key things from line 1.
```

- Self-assessment – pupils are given time to reflect on their work and enhance or debug their work, becoming effective independent learners. Charting their starting point
- Peer assessment – Pupils receive feedback from fellow pupils to improve their work.
    - Pupils also use 'Pair Programming' where one pupil writes exactly what the other instructs, thus prompting valuable discussion.
- Whole class feedback – if a key teaching point needs addressing for the whole class.

Curriculum Impact

Assessment

In line with the DASP Middle Schools we use a 5 band system which are reported to parents three times a year.

Teachers record bands for each different module where the success criteria for ARE has been decided on as a department. Work is then moderated as a department to ensure consistency. These are then used to inform the overall band at the end of each term.

Online quizzes are also used to assess pupils' vocabulary and retention skills which are then banded. This also adds to a pupils' final assessment band.

Monitoring

Monitoring is undertaken regularly to ensure consistency across the department. This is done through:

- Lesson drop ins.
- Looking at pupil work
- Pupil conferencing
- Looking at data
- Discussion with teachers within the department

Feedback will be given both individually and departmentally so that we can continue to progress as a department.