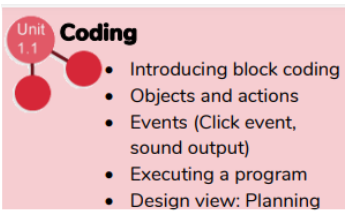
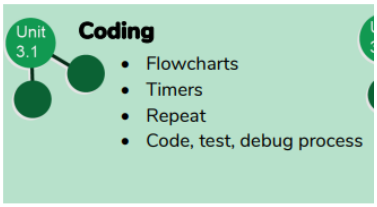
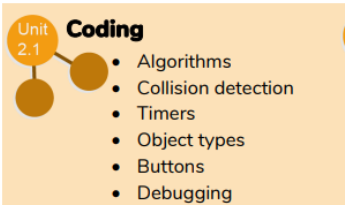
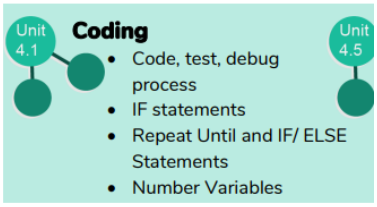
Year 5. Coding

Previous learning:









This unit builds upon the technical knowledge children have acquired in previous years and encourages a deeper level of thinking around the purpose behind a program as well as underlining accuracy. The first lessons encourage the children to look carefully at code and engage in discussions about how to simplify it in order to make it more efficient. They will also look at games and animations from a consumer perspective, and begin to think more about how to improve them, thinking about how to make them more challenging, interesting or varied. They will also study real-life, daily examples of programming and think about how they *should* work, based upon their own personal experiences, then re-creating them accurately in the form of a simulation. Pupils are introduced to new terminology such as decomposition and abstraction, and are shown how these techniques blueprint the coding process. Functions are introduced for the first time, further reinforcing the importance of time-saving efficient coding. Number and string variables are also new this year, and will help to demonstrate to the children some of the other forms variables may take.

Adaptive teaching – Recommended tried and tested practise in these lessons:

Due to the nature of Computing lessons and the common 'firefighting' required in lessons with technical issues or children who are individually confused over what may be considered small stumbling blocks, it is recommended that teachers put time and effort into reinforcing the use of school-wide learning ethos and systems (for example, the use of the 4B's – brain, board, buddy, boss) to encourage children to try to solve their problems in the right way rather than going straight to the teacher, or lessons will quickly become unmanageable. It is also recommended to follow our school's use of the Rosenshine model of mastery teaching strategy and teach lessons in a series of stages, with short, simple tasks being set, and children who finish early can be given encouragement to support and 'lifeguard' other students. This has twofold benefits; the process of teaching others cements and secures pupils' own mastery as well as ensuring the teacher isn't overwhelmed with personal requests for help. If one-to-one support is unavailable for high need children, it is a good idea to make sure these children are sat next to or near a higher ability student who may act as peer support for them, and these children are monitored carefully throughout lessons to ensure they stay on task and understand each objective given. Ensure that any hardware support that is required for individual's SEND requirements are adjusted appropriately and are tied to those students' individual log ins (e.g. Zoom tools or screen tints for students with visual impairments etc). Teachers should have a knowledge of Pupil Premium children, particularly those who have been seen to struggle in Computing lessons, and these children should be monitored alongside those with SEND during lesson time to help ensure they access lessons along with the rest of the class.

Knowledge covered:

NC	Ladders Objectives
(2.1) Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts	Y5 - CS1 - I can use a range of sensing tools (e.g. within programs such as Scratch) to control what happens (e.g. based on the positions of the sprite, mouse position or inputs such as sound level). Y5 - CS2 - I can recognise an error in a program, debug the program and explain the changes I have made.
(2.2) Use sequence, selection, and repetition in programs; work with variables and various forms of input and output	
(2.3) Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	

	Y5 - CS3 (KPI) - I recognise the need to test and retest whilst a program is being developed. Y5 - CS4 - I understand why it may be useful to simplify a program as much as possible (e.g removing unnecessary code or using a REPEAT instruction) Y5 - CS5 - I understand that any system requires input devices (e.g. keyboard) and output devices (e.g. printer). Y5 - CS7 - I understand what decomposition and abstraction are
(2.4) Understand computer networks including the Internet; how they can provide multiple services, such as the world wide web; and the opportunities they offer for communication and collaboration.	**Y5 - CS6 (KPI)** - I understand that computers can be linked together to create a network and that this can help me working with others online (collaboration) Y5 - CS8 - I can explain that computers can be connected together to form systems Y5 - CS9 - I can recognise the role of computer systems in our lives
(2.5) Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content.	
(2.6) Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.	
(2.7) Use technology safely, respectfully and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact.	

Medium Term Lesson Plan:

Lesson 1	L.I I can review my existing coding knowledge. L.I I can simplify code. L.I I can create a playable game. Y5 – CS1 Y5 – CS2 **Y5 - CS3 (KPI)** Y5 – CS4 Y5 – CS5 Y5 - CS7	S.C: I can use simplified code to make my programming more efficient. I can use variables in my code. I can create a simple playable game.	<u>Key Questions</u> *Prior knowledge* Which coding terms can you remember (Coding vocabulary quiz activity) Can you predict what will happen in this program? Can you include co-ordinates and an IF statement?	Vocabulary knowledge review activity. Activity 1: Animal race. Show how terms such as 'any' and the use of variables can make coding much more efficient. Activity 2: Children use collision detection to make events happen within the game. Discuss with the children how this game could be improved or made more challenging. Give them time to develop and improve their programs in their own way.
Lesson 2	L.I I understand what a simulation is. L.I I can program a simulation using 2Code. Y5 – CS1 Y5 – CS2 **Y5 - CS3 (KPI)** Y5 – CS4 Y5 – CS5 Y5 - CS7	S.C I can plan an algorithm modelling the sequence of traffic lights. I can select the right images to reflect the simulation I am making. I can use my plan to program the simulation to work in 2Code.	<u>Key Questions</u> *Prior knowledge* What is an algorithm? Could you write an algorithm for getting dressed? When would a simulation need to be used before creating something physically? What instructions may be needed for a traffic light sequence?	Children are introduced to the vocabulary 'algorithm' and 'simulation'. They should have some understanding of what an algorithm is from previous years' coding experience. Discuss how simulations are used in real life – roller coasters, pilot training, traffic systems etc. Children discuss/note down instructions they think are needed for traffic lights. Watch the video of traffic lights. Do they need to modify their instructions? What did they miss? Plan their algorithms, create and improve in 2code.

Year 5 Coding

Lesson 3	L.I I know what decomposition and abstraction are in Computer Science. L.I can take a real-life situation, decompose it and think about the level of abstraction. L.I I can use decomposition to make a plan of a real-life situation Y5 – CS1 Y5 – CS2 **Y5 - CS3 (KPI)** Y5 – CS4 Y5 – CS5 Y5 - CS7	S.C: I can make good attempts to break down their task into smaller achievable steps. I can recognise the need to start coding at a basic level of abstraction to remove unnecessary details from my program that do not contribute to the aim of the task.	<u>Key Questions</u> *Prior knowledge* How could you decompose an activity like brushing your teeth into smaller parts? When have decomposed a task in our lessons before? Which details have been included in the Football game? What hasn't?	Children are introduced to 'decomposition' and 'abstraction' and the terms are discussed, with examples given. They are then encouraged to think about how to use these ideas when looking at a task. They are then asked to choose a task of their own e.g. a board game to apply these concepts to. Consider partner work/focused support for children who may struggle with this activity.
Lesson 4	L.I I can understand how to use friction in code. L.I I can begin to understand what a function is and how functions work in code. Y3 - CS1 **Y3 - CS2 (KPI)** Y3 - CS3 Y3 - CS5 Y3 - CS6 Y3 - CS7	S.C: I can create a program which represents a physical system. I can create and use functions in my code to make my programming more efficient.	<u>Key Questions</u> *Prior knowledge* What causes a football to stop in real life? What is friction? Can you predict what will happen when the program is run? How does this function help us in our programming?	Children are introduced to a football game where a ball is dragged and let go, and a timer shows how long it moves for until it stops. They use different buttons to affect the time with increasing levels of friction – sand, ice, grass. Friction is discussed and explained. Children are shown the code for this program and are then introduced to functions – a series of commands which can be called and executed with one line of code. Children create their game and reflect on their work.
Lesson 5	L.I I can understand what the different variable types are and how they are used differently. L.I I can understand how to create a string. Y5 – CS1 Y5 – CS2 **Y5 - CS3 (KPI)** Y5 – CS4 Y5 – CS5 Y5 - CS7	S.C: I can create and use strings in programming. I can set/change variable values appropriately. I know some ways that text variables can be used in coding.	<u>Key Questions</u> *Prior knowledge* What are variables? Where have we used them before in our coding? What types of values did these variables contain? Can you see the string in the Alien Blast game? How does it help inform the person playing the game about what's happening?	Children reflect on what they have learned about variables so far. They learn that variables can come in different forms. Today we will be working with a *string* variable. They are shown an 'Alien Blast' game which uses a string variable to inform progress. Children will be shown how this variable works, and how the code in this game is organised by using tabs. They are given time to examine the code and test the game to see how it works. Children then go into the game and edit the design to make it their own, then add a text variable. Challenge: Add a score and a timer.
Lesson 6	L.I I can begin to explore text variables when coding. L.I I can understand what concatenation is and how it works. Y5 – CS1 Y5 – CS2	S.C: I can create a string and use it in my program. I can use strings to produce a range of outputs in their program.	<u>Key Questions</u> *Prior knowledge* In which ways could we use concatenation with string variables? (Remind them of the Alien Blast game and what string variables are)	New vocabulary introduced: Concatenation means joining two strings together. For example, you may wish to concatenate a first name and surname to form a person's full name.

Year 5 Coding

	Y5 - CS3 (KPI) Y5 – CS4 Y5 – CS5 Y5 - CS7			They are given a block of code and are asked to predict how it will run and what it does. Run, then see the effect. They are given another block which links strings (concatenation) and again examine, predict then run the code. Children are challenged to debug the errors Another block of code is given for them to study number variables. Children are then challenged to use random word instructions to create funny sentences.