HIGH LITTLETON CHURCH OF ENGLAND PRIMARY SCHOOL
COMPUTING MEDIUM TERM PLAN TERM 6
2024-2025

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
Hedgehog (Y1)	**Comparing tools** During this lesson learners will become accustomed to the ScratchJr programming environment. They will discover that they can move characters on-screen using commands, and compare ScratchJr to the Bee-Bots used in the previous unit.	**Joining blocks** During this lesson learners will discover that blocks can be joined together in ScratchJr. They will use a Start block to run their programs. They will also learn additional skills such as adding backgrounds and deleting sprites. Learners will follow given algorithms to create simple programs.	**Make a change** During this lesson learners will discover that some blocks in ScratchJr have numbers underneath them. They will learn how to change these values and identify the effect on a block of changing a value.	**Adding sprites** During this lesson learners will be taught how to add and delete sprites in ScratchJr. They will discover that each sprite has its own programming area, and learn how to add programming blocks to give instructions to each of the sprites.	**Project design** During this lesson learners will choose appropriate backgrounds and sprites for a 'Space race' project. They will decide how each sprite will move, and create an algorithm based on the blocks available in ScratchJr that reflects this.	**Following my design** During this lesson learners will use their project designs from the previous lesson to create their projects on-screen in ScratchJr. They will use their project design, including algorithms created in the previous lesson, to make programs for each of their rocket sprites. They will test whether their algorithms are effective when their programs are run.	**POP task**
Fox (Y2)	**Scratch Jr recap** During this lesson, learners will recap what they know already about the ScratchJr app. They will begin to identify the start of sequences in real-world scenarios, and learn that sequences need to be started in ScratchJr. Learners will	**Outcomes** During this lesson, learners will discover that a sequence of commands has an 'outcome'. They will predict the outcomes of real-life scenarios and a range of small programs in ScratchJr. Learners will then	**Using a design** During this lesson, learners will be taught how to use the Start on tap and Go to page (Change background) blocks. They will use a predefined design to create an animation based on	**Changing a design** During this lesson, learners will look at an existing quiz design and think about how this can be realised within the ScratchJr app. They will choose backgrounds and characters for their	**Designing and creating a program** During this lesson, learners will create their own quiz question designs including their own choices of question, artwork, and algorithms. They will increase the number	**Evaluating** During this lesson, learners will compare their projects to their designs. They will think about how they could improve their designs by adding additional features. They will	**POP task**

	create programs and run them in full-screen mode using the Green flag.	match programs that produce the same outcome when run, and use a set of blocks to create programs that produce different outcomes when run.	the seasons. Learners will then be introduced to the task for the next lesson. They will predict what a given algorithm might mean.	own quiz projects. Learners will modify a given design sheet and create their own quiz questions in ScratchJr.	of blocks used within their sequences to create more complex programs.	modify their designs and implement the changes on their devices. Learners will find and correct errors in programs (debug) and discuss whether they debugged errors in their own projects.	
Badger (Y3)	**Moving a sprite** In this lesson, learners will investigate how characters can be moved using 'events'. They will analyse and improve an existing project, and then apply what they have learned to their own projects. They will then extend their learning to control multiple sprites in the same project.	**Maze movement** In this lesson, learners will program a sprite to move in four directions: up, down, left, and right. They will begin by choosing a sprite and sizing it to fit in with a given background. Learners will then create the code to move the sprite in one direction before duplicating and modifying it to move in all four directions. Finally, they will consider how their project could be extended to prove that their sprite has successfully navigated a maze.	**Drawing lines** This lesson will introduce learners to extension blocks in Scratch using the Pen extension. Learners will use the pen down block to draw lines, building on the movement they created for their sprite in Lesson 2. Learners will then decide how to set up their project every time it is run.	**Adding features** In this lesson, learners will be given the opportunity to use additional Pen blocks. They will predict the functions of new blocks and experiment with them, before designing features to add to their own projects. Finally, they will add these features to their projects and test their effectiveness.	**Debugging movement** This lesson explores the process of debugging, specifically looking at how to identify and fix errors in a program. Learners will review an existing project against a given design and identify bugs within it. They will then correct the errors, gaining independence as they do so. Learners will also develop their projects by considering which new setup blocks to use.	**Making a project** In this lesson, learners will design and create their own projects. Using a template (which can be blank or partially completed), learners will complete projects to move a sprite around a maze, with the option to leave a pen trail showing where the sprite has moved. Ideally, projects will include setup blocks to position the sprite at the start of the maze and clear any lines already on the screen.	**POP task**
Otter (Y4)	**Using loops to create shapes** In the first lesson, learners look at real-life examples of repetition, and identify which parts of instructions are	**Different loops** In this lesson, learners look at different types of loops: infinite loops and count-controlled loops. They practise using these within	**Animate your name** In this lesson, learners create designs for an animation of the letters in their names. The	**Modifying a game** In this lesson, learners look at an existing game and match parts of the game with the design. They make	**Designing a game** In this lesson, learners look at a model project that uses repetition. They then design their own games	**Creating our games** In this lesson, learners look at a model project that uses repetition. They then design	**POP task**

	repeated. Learners then use Scratch, a block-based programming environment, to create shapes using count-controlled loops. They consider what the different values in each loop signify, then use existing code to modify and create new code, and work on reading code and predicting what the output will be once the code is run.	Scratch and think about which might be more suitable for different purposes.	animation uses repetition to change the costume (appearance) of the sprite. The letter sprites will all animate together when the event block (green flag) is clicked. When they have designed their animations, the learners will program them in Scratch. After programming, learners then evaluate their work, considering how effectively they used repetition in their code.	changes to a sprite in the existing game to match the design. They then look at a completed design, and implement the remaining changes in the Scratch game. They add a sprite, re-use and modify code blocks within loops, and explain the changes made.	based on the model project, producing designs and algorithms for sprites in the game. They share these designs with a partner and have time to make any changes to their design as required.	their own games based on the model project, producing designs and algorithms for sprites in the game. They share these designs with a partner and have time to make any changes to their design as required.	
Robin (Y5)	**Exploring conditions** In this lesson, learners revisit previous learning on 'selection' and identify how 'conditions' are used to control the flow of actions in a program. They are introduced to the blocks for using conditions in programs using the Scratch programming environment. They modify the conditions in an existing program and identify the impact this has.	**Selecting outcomes** In this lesson, learners will develop their understanding of selection by using the 'if... then... else...' structure in algorithms and programs. They will revisit the need to use repetition in selection to ensure that conditions are repeatedly checked. They identify the two outcomes in given programs and how the condition informs which outcome will be selected. Learners use this knowledge to write their own programs that use selection with two outcomes.	**Asking questions** In this lesson, learners consider how the 'if... then... else...' structure can be used to identify two responses to a binary question (one with a 'yes or no' answer). They identify that the answer to the question is the 'condition', and use algorithms with a branching structure to represent the actions that will be carried out if the condition is true or false. They learn how questions can be asked in Scratch,	**Designing a quiz** In this lesson, learners will be provided with a task: to use selection to control the outcomes in an interactive quiz. They will outline the requirements of the task and use an algorithm to show how they will use selection in the quiz to control the outcomes based on the answer given. Learners will complete their designs by using design templates to identify the questions that will be asked, and the outcomes for	**Testing a quiz** In this lesson, learners will use the Scratch programming environment to implement the first section of their algorithm as a program. They will run the first section of their program to test whether they have correctly used selection to control the outcomes, and debug their program if required. They will then continue implementing their algorithm as a program. Once completed, they will	**Evaluating a quiz** In this lesson, learners will return to their completed programs and identify ways in which the program can be improved. They will focus on issues where answers similar to those in the condition are given as inputs, and identify ways to avoid such problems. Learners will also consider how the outcomes may change the program for subsequent users, and identify how	**POP task**

			and how the answer, supplied by the user, is used in the condition to control the outcomes. They use an algorithm to design a program that uses selection to direct the flow of the program based on the answer provided. They implement their algorithm as a program and test whether both outcomes can be achieved.	both correct and incorrect answers. To demonstrate their understanding of how they are using selection to control the flow of the program, learners will identify which outcomes will be selected based on given responses.	consider the value of sharing their program with others so that they can receive feedback. Learners conclude the lesson by using another learner's quiz and providing feedback on it.	they can make use of 'setup' to provide all users with the same experience. They will implement their identified improvements by returning to the Scratch programming environment and adding to their programs. They conclude the unit by identifying how they met the requirements of the given task, and identifying the aspects of the program that worked well, those they improved, and areas that could improve further.	
Deer (Y6)	**The micro:bit** Pupils will be introduced to the micro:bit as an input, process, output device that can be programmed. Pupils will familiarise themselves with the device itself and the programming environment, before creating their own programs. They will then run their programs on the device.	**Go with the flow** Pupils will explore how if, then, else statements are used to direct the flow of a program. They will initially relate if, then, else statements to real-world situations, before creating programs in MakeCode. They will apply their knowledge of if, then, else statements to create a program that features selection influenced by a random number to	**Sensing inputs** Pupils will initially use the buttons to change the value of a variable using selection. They will then develop their programs to update the variable by moving their micro:bit using the accelerometer to sense motion. Finally, they will learn that a variable's value remains the same after it has been	**Finding your way** Pupils will apply their understanding of the importance of order in programs. They will then use operands in selection to determine the flow of a program. Pupils will then modify a program which will enable the micro:bit to be used as a navigational device. To code this, they will adapt the code they completed to make a basic compass.	**Designing a step counter** Pupils will be working at the design level. They will pick out features of a step counter, a piece of technology with which they are likely to be familiar. They will then relate those features to the sensors on a micro:bit. In the main activity, pupils will design the algorithm and program flow for their step counter project.	**Making a step counter** Pupils will use the design that they have created in Lesson 5 to make a micro:bit-based step counter. First they will review their plans, followed by creating their code. Pupils will test and debug their code, using the emulator and then the physical device. To successfully complete this	**POP task**

		create a micro:bit fortune teller project.	checked by the program.			project, Pupils will need to demonstrate their understanding of all the programming lessons they've had so far.	