Computing Progression Document KS1

	FS1	FS2	Year 1	Year 1	Year 2	Year 2
Programming	**Programming** – All about instructions	**Programming** -Beebots (Adult Supported)	**Programming** Algorithms unplugged	**Programming** Bee-Bots - Virtual	**Programming** Algorithms & Debugging	**Programming** Programming – Scratch Jr
	-To following instructions given by an adult -Giving simple instructions to others -To debug instructions (washing hands)	-To learn the meaning of direction arrows -To follow simple instructions using arrows -To tinker with beebots -To follow simple algorithms and program their beebot -To debug and try again when something goes wrong	**1.What is an Algorithm? - To understand what an algorithm is** -To recognise what we mean by a computer -To understand why we need to log in to a computer -To log in and out of a computer account	**1.Getting to know a virtual device - To explore a new device** -To 'Tinker' with the buttons of an online Bee-Bot -To complete a number of challenges by: --thinking about what they might do first ('predict') --trying it out ('explore) --seeing if I was right ('explain')	**1.Dinosaur Algorithm -** **To decompose a game to predict the algorithms that are used** -To understand the definitions: decomposition and algorithm -To decompose a game to predict algorithms -To plan algorithms for a more complex game	**1. Using ScratchJr** **To explore a new application** -To know that ScratchJr is a coding application -To predict what I think something new will do -To explore something independently -To explain what I found using ScratchJ
			2.Algorithm Pictures - To follow instructions precisely to carry out an action -To explain why an algorithm must be clear and precise -To explain the problems a robot can have following our instruction	**2. Making a virtual Bee-Bot video - To create a demonstration video** - To create a video to explain how to use a Bee-Bot by: -Taking a video recording -Trying it out ('explore') -Seeing if I was right ('explain')	**2.Machine Learning - To understand that computers can use algorithms to make predictions (machine learning)** -To explain what an algorithm is -To explain that computers use algorithms to make predictions -To write a clear and precise algorithm	**2.Creating an animation - To create an animation** -To use the programming blocks I've learned about for a purpose -To recognise a loop in programming -To think about how animals move -To use my programming skills creatively to use code to represent an animal moving
			3. Virtual Assistants - To understand that computers and devices around us use inputs and outputs -To identify some input devices -To identify some output devices -To identify some devices that are both input and output devices	**3. Precise instructions - To plan and follow a set of instructions precisely** - To take on all of the following roles: --'Bee-Bot' (following instructions given by the controller) --'Controller' (giving instructions to the Bee-Bot) --'Judge' (checking that the instructions given by the 'controller' are correct)	**3.Through the Maze - To plan algorithms that will solve problems** -To devise and create algorithms to solve problems -To include loops in my algorithms (count controlled) -To visualise directions from a 2D environment	**3. Making a musical instrument - To use characters as buttons** -To design a musical instrument -To program code to run 'on tap' -To select appropriate blocks for my purpose
			4. Step by Step - To understand and be able to explain what decomposition is -To explain that decomposition is where you break a problem into small manageable chunks -To understand how decomposition allows you to solve a problem more easily -To explain how we use decomposition in our everyday lives	**4. Bee-Bot world - To program a device** -To personalise my Bee-Bot world -To consider how the Bee-Bot can move from one place to another -To plan a Bee-Bot route -To program a Bee-Bot to follow my planned route	**4.Making Maps - To understand what abstraction is** -To explain what abstraction is -To give an example of when abstraction might be useful	**4. Programming a joke - To follow an algorithm** -To use an algorithm to help me with my programming -To sequence the blocks appropriately -To explain what each block in the program does

			5.Debugging Directions - To know how to debug an algorithm -To spot bugs in algorithms -To fix the error (debug it) and explain the problem it caused	**5.Bee-Bot adventures - To create a program** -To know I should not move the Bee-Bot with my mouse -To know how to use programming to give the Bee-Bot clear instructions -To debug my instructions if they go wrong by identifying and correcting the mistake	**5.Unplugged Debugging - To understand what debugging is** -To understand the meaning of the word 'debugging' -To listen to my peer's verbal instructions -To perform a task by following step-by-step instructions	**5. 'The Three Little Pigs' algorithms - To plan and use code to create an algorithm** -To explain what an algorithm is -To choose the code to match my algorithm -To use an algorithm to write a computer program

Computing Progression Document KS2

	Year 3	Year 4	Year 4	Year 5	Year 5	Year 6
Programming	**Programming** Programming: Scratch	**Programming** Further coding with Scratch: Microsoft Office 365	**Programming** Computational Thinking	**Programming** Programming Music: Scratch	**Programming** Lego	**Programming** Intro to Micro:bit
	1.Tinkering with Scratch - To explore a programming application -To know that Scratch is a coding application -To predict what I think different codes will do -To explore an application independently -To explain what I found	**1.Scratch Reminder - To recall the key features of Scratch** -To know what the main parts of Scratch are called -To recognise how to adjust my sprite in Scratch -To add a new sprite to my stage to write a simple script	**1.What is computational thinking? To understand that computational thinking is made up of four key strands** -To understand that problems can be made easier if I use computational thinking -To know that computational thinking is made up of four strands: decomposition, pattern recognition, abstraction and algorithm	**1. Tinkering with Scratch music elements - To tinker with Scratch music elements** -To know that Scratch is a coding application that has music elements -To predict what I think different code blocks will do -To explore Scratch independently -To explain what I found from tinkering	**1. L.O. Code the legs** **L.O. Code the hub numbers** -To open the Lego Spike app -Select the Prime Solution -To create a new project -To rename the project -To investigate the Hub and the motor ports -To use the algorithm given and make the robots legs move -To use the code given and make the hub numbers change -To tinker with the algorithms and change the speed -To tinker with the algorithms and change the lights on the hub -To debug the algorithms if needed	**1: Name badge** **Students create their first programs and transfer them to their micro:bits.** -To explain that the micro:bit is a tiny computer. -To explain that computers need to be given sets of instructions (an algorithm) in code. -To give the micro:bit instructions in code to make a name badge using the LED display output. -Understand the micro:bit is a tiny computer which needs instructions in code to make it work. -Understand that sets of instructions for computers in a sequence are also called algorithms or programs. -Use the MakeCode editor to create instructions in code that the micro:bit can understand and then transfer them to the micro:bit. -Know the micro:bit has an LED display output which it can use to show words (as well as numbers and pictures).
	2.Using Loops -To use repetition (a loop) in a program -To understand and explain what a loop is -To recognise when a loop is used -To choose an appropriate loop	**2. Identifying what Code Does - To understand how a Scratch game works by using decomposition to identify key features** -To recognise that a sprite may contain more than one script -To identify the parts of a Scratch game -To understand what we mean by decomposition	**2.Decomposition - To understand what decomposition is and how to apply it to solve problems** -To decompose a problem -To use decomposition to figure out what Scratch code does -To decompose a problem to figure out which code blocks might have been used	**2. Scratch Soundtracks - To create a program that plays themed music** -To use Scratch's basic sound commands -To include a loop in my program -To debug simple errors in my code	**2. L.O. Code the swinging arms** -To open the existing project -To find which motor hub the arms are connect too -To use the code given to make the arms move -To tinker a code creating an algorithm to make the arms move -To create a dance that involves both the arms and legs	**2. Beating heart** **Create a simple animation to learn about sequence and simple loops.** -To create a micro:bit animation using a sequence of images in a loop. -To explain that the order or sequence of instructions is important. -To explain that loops can make code more compact and easier to read. -Understand that sequence and timing is important when making an animation. -Understand that animations create an illusion of movement by showing a sequence of still images. -Code the micro:bit to show simple animations on its LED display output. -Use loops to make animations run longer using fewer instructions.
	3. Making an Animation **To program an animation** -To decompose a project -To plan what I want to happen -To select the blocks to make that happen	**3. Introduction to Variables -** **To understand what a variable is and how to make one** -To use the 'ask' block in Scratch -To what a variable means -To make a variable	**3. Abstract & pattern recognition - To understand what pattern recognition and abstraction mean** -To know how to recognise patterns -To understand how to abstract key information -To understand how to abstract key information	**3. Planning a Soundtrack - To plan a soundtrack program** -To decompose a story -To plan my program by tinkering -To explain how my program will add to the story	**3. L.O. Code – beats your robot** --To open the existing project -To follow the instructions, to add the Music tab on the Spike App -To use the code given to create music -To tinker a code creating an algorithm to make music move	**3. Emotion badge** **Start learning about inputs and outputs using buttons and icons on the display.** -To make the micro:bit show different pictures on the LED display output depending on which button input is pressed -To explain that inputs are data sent to a computer. -To explain that outputs are data sent from a computer. -Code the micro:bit to make different outputs happen depending on different inputs. (This is a very simple kind of selection. We look at selection in more detail in lesson 5, Nightlight.)

		-To store an answer to a question as a variable				-Understand that inputs and outputs involve the flow of data in and out of computers. -Apply this knowledge using the micro:bit's button inputs and display output.
	4. Storytelling - To program a story -To choose appropriate blocks -To continue someone else's program -To debug my own program	**4. Making a Variable - To understand how to make a variable in Scratch** -To create a variable and use it to store information -To 'call' a variable within my program -To identify that variables can be words or numbers	**4. Algorithm Design -To understand how to create an algorithm and what it can be used for** -To create an algorithm for drawing a square -To use my algorithm to write a script using Scratch -To use pattern recognition to modify my script to draw different shapes	**4. Programming a Soundtrack - To program a soundtrack** -To work from a plan -To use a range of programming commands -To explain how my program enhances the scene	**4. L.O. Robotics within present day society** -To understand what is meant by robotics -To understand what is meant by present day society -To know that technology is always forever changing and improving -To know this includes, houses, factories, shops, mechanics, farming, films etc -To research and list robotics used today in everyday life	**4. Step counter** **Introduce variables to track your step count and begin to use the accelerometer input.** -To turn my micro:bit into a step counter using the accelerometer and variables -To explain that the accelerometer is a sensor, an input that senses movement. -To explain that variables are containers for storing data which can be accessed and updated. -Understand how sensor inputs from the accelerometer can be used to detect movement, such as when a step is taken. -Understand that variables are used to keep track of the current step count. -Understand that the order of instructions is important: display the variable's value after updating it, not before. -Apply this learning to build a practical, real-world
	5. Programming a Game - To program a game -To explain the purpose of an algorithm -To decompose a problem -To use an algorithm to code a program	**5. Times tables Project - To use knowledge of how variables work to create a quiz** -To create a range of questions and use an 'if/else' block to check whether the answer is correct -To use a variable called 'score' to calculate the total number of correct answers for those completing my quiz -To make sure my quiz is engaging and exciting for the people playing it	**5. Applying Computational Thinking - To combine computational thinking skills to solve a problem** -To apply decomposition, pattern recognition, abstraction and algorithm design to problems -To work with a partner and discuss how to solve a problem	**5. Battle of the Bands - To program music for a specific purpose** -To combine known commands -To code music with a purpose -To use repetition in a program -To use various forms of output [sound]	**5. L.O. Understanding the advantages and disadvantages of robotics in society** -To understand the advantages of robotic technology -To understand the disadvantages of robotic technology	**5. Lesson 5: Nightlight** **Make an automatic nightlight and discover how logic, conditionals and inputs and outputs combine to make a simple control system.** -To code a micro:bit to make a light that switches on when it gets dark using sensors and logic. -To explain that sensors are inputs that sense things in the real world, such as movement and light. -To explain that logic is how computers make decisions in code based on whether things are true or false. -Understand how inputs, outputs, and computer code work together to make control systems. -Understand how logic (conditional 'if… then… else' instructions) is used to make different outputs happen depending on changes in data from a sensor. -Use 'forever' infinite loops to keep control systems responding to changes in the environment. -Practise testing and improving a project to make the nightlight work better in specific local lighting conditions.