7.4 Introduction to Scratch

Number of weeks (between 6&8)	**Content of the unit**	**Assumed prior learning (**tested at the beginning of the unit)
6/7	In this unit pupils will be introduced to the Scratch programming environment and begin by reverse-engineering some existing games. They will then progress to planning and developing their own games, learning to incorporate variables, procedures (using the Broadcast function), lists and operators. They should be able to create a fully working game with lives, scoring and some randomisation of objects. Finally they will learn to test and debug their programs	No previous learning is necessary with this unit but most pupils will have had experience of programming or sequencing instructions from units studied in previous years using software such as Logo, RoboMind, Kodu or GameMaker. They should be familiar with using screen objects.

Assessment points and tasks	**Written feedback points**	**Learning Outcomes (tested at the end and related to subject competences)**
Pupils will submit an Assessment Portfolio of the unit for assessment where they can include details of their planning and testing, and screenshots of the code they used. The final version of the game they create can be accessed separately to form the overall assessment for the unit. The assessment describes grades as Basic, Intermediate, Advanced or Expert. It is expected that teachers will map these onto their own school assessment structure for Computing and ICT. This Schemes of work is aimed towards GCSE Grades F-D	Provide feedback mid-way through the unit. Provide written feedback following the end of unit assessment	***Learning Outcomes for the unit*** **At the end of this Unit all pupils should be able to:** • Relate computational abstractions and simple programming code to on-screen actions • Design simple algorithms to solve problems • Sequence instructions in order to make things happen • Use variables in programming structures • Assemble code in procedural blocks • Use simple Boolean operators in programming code • Identify and use screen objects in their own Scratch game • Carry out simple tests to debug their project **Most pupils will be able to:** • Write their own instructions to create and use a simple list (inventory) • Use the broadcast function in Scratch at a simple level • Make good use of operators • Incorporate a range of sprites which can be controlled in different ways • Improve their project based on peer feedback • Systematically test their own projects to ensure that few errors remain **Some pupils will be able to:** • Use the broadcast function in Scratch effectively

		• Use a range of 'event handlers' effectively to create a complex project • Effectively design, implement and refine their own algorithms • Compare the effectiveness of their algorithms with those of peers • Critically analyse the limitations of their projects

Lesson	**Clear learning intentions**	**Clear success criteria**	**Hook**	**Presentation of content**	**Guided practice**	**Independent practice** (homework)	**Closure**
1	**Learning Objectives:** • Understand that Scratch is a programming environment that allows you to create games, animations and other simulations • Understand what is meant by an algorithm • Create a sprite and write code to make it move and bounce • Load and use an existing Scratch file • Produce design ideas for a Scratch project	Students will be able to identify command blocks and write a simple program	Introduce pupils to the idea of sequencing instructions in a particular order to achieve a desired end result – for example, a recipe. Show pupils the first slide to follow a recipe for Chocolate Rice Krispie squares. Highlight the difference between the ingredients needed and the instructions to make them. This can be compared with the variables and code blocks later.	PowerPoint Guide L1 Worksheet 1 Sequencing Instructions	A simple demonstration will show how to add these instructions in Scratch to make a sprite move left. Pupils can repeat this and add subsequent instructions for Up, Down and Right. A smoother movement can be produced but this involves more complex code and it may be better to leave this as a modification later on when pupils have a better understanding of Scratch. Pupils can add a new sprite and make this move automatically using the **forever** control loop. They may wish to look at the different ways of adding a new sprite: either by using one from the library, by creating one themselves using the costume editor or by importing an image saved on their computer.	What happens when one sprite touches the other? Use **Worksheet 3 Writing an Algorithm** to plan out how this might be implemented.	Show pupils a completed game to give them an idea of what can be achieved.

	• Define a variable • Write algorithms which use variables to hold values such as **Number of Lives Left** or **Score** in a computer game • Understand the purpose of comments in a program • Annotate a program with comments	Learning about variables and understanding how they affect the games Learn to write pseudocode and algorithm	Starter – Pupils discuss what should happen when one sprite touches the other. In pairs, they come up with simple algorithms to achieve their ideas.	PowerPoint Guide L2	Using the Fish Game, demonstrate how to add a code block to the Fish to stop all if it touches the Shark. Highlight the use of a loop (**Forever**) and a selection statement (**If**). Pupils can modify their own simple games from last lesson in the same way and can also use the forever and if blocks to make the movement smoother. Using **Lives** as an example – where would the number of lives they currently have be stored? How many lives would they start with? Introduction to variables. Explain that a variable is a name given to a location in memory, which will store a value – so **Lives**, for example, will store an initial value of, say, 3 to represent the starting value in the game. Show pupils how to use simple pseudocode to work out the steps necessary to set the starting value of **Lives** and reduce it by one when one sprite touches another. When pupils have worked out the steps, demonstrate using the Fish Game how to define a variable called **Lives** and use it to set and deduct lives in a game. Show pupils that variables can be represented in a variety of different ways and that they can hide them or drag them around on the screen to change their position. Right-click on a variable to change its setting. A completed game can then be briefly demonstrated again with more variables representing scoring, for example, to provide more ideas. Pupils should refer to their completed Games Design worksheets to add a **Lives** variable into their own games. Pupils can then think about adding to a **Score** when they 'collect' something. This will involve creating an additional sprite and using the **Show/Hide** commands to make it disappear on contact with their character.	Find a simple game on the Scratch Community site and print the code blocks. Annotate the code to explain what various blocks of code do.	Explain to pupils how they can add Comment boxes in Scratch to help them, and others, understand their code at a later date. If they right-click a block they can add a comment which attaches itself to the block. This is proper practice in the programming world and will help them in their projects. Let pupils spend five minutes adding comments to their code.

3	• Understand the purpose of repeat loops and procedures (“broadcasts”) • Use a broadcast in your own Scratch program	Be able to create a continuous loop within their game And broadcast a message. i.e their name.	Starter – Efficiency of code. Using a **repeat** loop wraps up code into a block that can be repeated. A **procedure** takes advantage of similar efficiencies by wrapping up a section of code and giving it a name.	PowerPoint Guide L3	Explain that the verb (or procedure) ‘Dance’ contains a whole sequence of minor instructions (left leg forward, left leg back and so on) that we don’t need to iterate explicitly (for most people!) Broadcasting ‘Dance’ could get someone else (maybe a pupil?) to carry out these individual instructions. Explain that a procedure uses a similar concept. Programmers can write a sequence of code that they wish to use several times and give that block of code a name (the procedure name). The procedure can then be ‘called’ at a specific time. By using this name in your code, you ask the program to run the sequence of instructions wrapped up inside the procedure. Procedures in Scratch are called **broadcasts**. This is like shouting out a command, and waiting for others to respond. The advantage of a broadcast in Scratch is that other scripts will continue to execute concurrently. This way you can get several things to happen at once, for example, getting a shark to open and close its mouth whilst moving. Demonstrate a broadcast procedure in Scratch to temporarily change the costume of a sprite when touched by another. Pupils can experiment with this in their own games. Demonstrate how to use the broadcast function to change the background of the Stage to create a new level when a certain condition is satisfied – for example, number of points collected reaches a target value. Encourage pupils to use the Scratch community or the accompanying games Moon Mission and Ruin Rescue to find other example uses of broadcasting. Once a level change has occurred, pupils may want to experiment with adding variables to control the speed of sprites. They can change these variables at a level change in order to increase the difficulty.	Encourage pupils to use the Scratch community or the accompanying games Moon Mission and Ruin Rescue to find other example uses of broadcasting.	Allow pupils time at the end of the lesson to document their new code by adding comments where appropriate.

4	• Learn what each of the operators in the Scratch Green block menu does • Use the **Pick Random** block to position objects randomly on the screen • Understand the use of the operators **<, =, >, and, or, not**. • Use some of these in a Scratch game	Be able to select and include appropriate operators for their own game.	Explain that the Scratch window has coordinates to define a single point on the screen. A particular point on the screen is indicated below the bottom right of the Scratch window as x, y coordinates. Demonstrate how to position objects on the screen and how to use the **Pick Random** block to change these values.	PowerPoint Guide L4	Use the Fish Game to demonstrate adding a new Crab sprite and randomising its appearance and movement: Ask pupils to look at the AND, OR and NOT operators. Challenge 1 – use built-in help or the Internet to learn how to use operators Challenge 2 – consolidate learning by making objects appear randomly on the screen Pupils should use the remainder of this lesson to develop their game and add comments to all their new blocks of code. Popular suggestions for further development might include jumping and shooting (Covered in lesson 5, see **Link Ruin Rescue** or **Link Get to the Portal**), how to create a scrolling background (**Link Scrolling Example**), or increasing difficulty by creating more monsters, for example (See **Link Moon Mission** for an example on Cloning). **Using Lists** It is also possible to use Lists to create a Backpack function in which they can store items that they collect during the game. Depending on the items in the bag, different things might happen. For example, if Backpack contains Key, then open door. An example of using Lists in this manner is illustrated in Link FruitCraftRPG, and teaching materials to support Lists are featured at the back of this guide in an extension lesson with PowerPoint Guide LE Extension.	Continue with game development. Research and add one new game feature.	Peer testing Students will be asked to provide feedback on each other's game

5	• Learn programming techniques to add shooting at a target into a game • Learn how to adjust x and y coordinates to control the position of a sprite • Learn how to make a sprite jump	Pupils will add shooting as and where appropriate to their own games	Students will be shown examples of games of target shooting and demonstrate how to implement in own game	PowerPoint Guide L5	Pupils will add shooting as and where appropriate to their own games. Encourage them to use existing games and searches on the Scratch community to find examples that they can use in their own games. Demonstrate jumping using the **Ruin Rescue** game. In order for pupils to understand jumping, they need to have understood the use of x, y coordinates and variables. A variable to adjust the y coordinate will be required to control the jump. Show pupils the example on the PowerPoint slide and allow them to look at the **Ruin Rescue** game, the **Get to the Portal** game or others on the Scratch community website. Ask pupils to comment their new code.	Students should annotate and evaluate their scratch game, focusing on any improvements they could make and changes they would like t o make.	Ask pupils to comment their new code.
6	• Learn how to add sound to a Scratch game • (Complete your game using the skills you have learned)	Students should be able to add sound into their games	Sound will be shown how to implement sound into their games	PowerPoint Guide L6	This lesson is designed to give pupils extra time to complete their games using all of the skills learnt so far. If you feel that your pupils have spent sufficient time already on their games, this lesson may not be necessary. Sounds can be added to Scratch in several different ways as indicated on the Sounds tab of a Sprite. You can: • Use a pre-recorded sound from the Scratch library • Record a sound yourself and import it • Upload a sound you have saved on your computer Link Find Sounds or www.findsounds.com may help find a particular sound effect if the Scratch library doesn't have one. Once you have selected a sound to use, the pink Sound blocks can be inserted into the code as appropriate	Revise for upcoming test in next lesson	Go over topics covered in unit so far. Topics to cover for assessment

7	• Carry out final tests • Perform a self-evaluation of level of skills and understanding achieved for the unit • Complete the Assessment Portfolio	Complete the test to the best of their ability, remember on previous learning on the topic	Pupils should be given time, if necessary, to test their own and possibly each other's games.	Assessment sheets	In the Assessment Portfolio, they will answer some questions to test their understanding of what they have learned in the unit. They will complete a self-evaluation to assess the level that they have attained.	none	End of Unit