Space Adventures Computing Unit 1

Lesson 3 – Moon Buggy

Curriculum Mapping (Computing KS2)

- use selection and repetition in programs; work with various forms of input and output
- use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

Learning Objective
Code a moon buggy simulation, that can be controlled by the cursor keys and stops when it hits a crater.

Prior Learning
Previous lessons in the unit / experience of coding using inputs.

Introduction
Show pupils the ***U1L3 introduction.mp4 video.***
Using the prompts in the video, ask pupils to identify the rules that make the moon buggy move, turn and stop. (This can be done orally or written in rough, and is known as the algorithm).

Main Activity
Pupils use Scratch to create their own simulation of Tazz's moon buggy exploring the surface of the moon.
Show the class the ***U1L3 demonstration.mp4 video*** or how to access it on their own computers.
Hand out the ***U1L3 step by step.pdf*** guide or show pupils how to access it on their computers. (Opening a second tab in the browser will allow pupils to switch between the help guide/video and their own work).

Extension Activity
Show pupils the activities on the ***U1L3 going further_export.pdf*** document. These include experimenting with aspects of the code, adding code to make the moon buggy reverse and drawing what Tazz might see on the moon.

Plenary
Explain that pupils will be using their code in the next lesson. Ask a pupils to demonstrate how to download or save their work, and ensure pupils save/download their work.

Notes
This program is similar to lesson 2, but instead of using **when key pressed** event blocks to steer, all the code in is one longer loop. **If key pressed then** blocks are used to check when different keys are pressed. This shows pupils a different way to code and detect inputs. This method also detects a key being held down more quickly than using event blocks, making a better game.

To check whether a crater has been hit, the code checks if the buggy is touching dark grey. If it is touching it moves the buggy backwards by using a negative value in a move block. Make sure pupils understand how important negative numbers are in coding. Pupils should save their work at the end of this lesson as it will form the basis of the next lesson.