Seamer and Irton CP School – Computing (H.Griffiths)

Topic – Selection in Physical Computing	Year 5 Spring 1	Strand – Programming

Prior Learning

In Year 4 – Summer 1 – Repetition in shape Learners explored the concept of repetition in programming using the Scratch environment. Learners looked at the difference between countcontrolled and infinite loops and used their knowledge to modify existing animations and games using repetition. Their final project was to design and create a game which used repetition, applying stages of programming design throughout. Through previous programming units, learners will have prior experience of programming using block-based construction (eg Scratch) and understand the concepts of 'sequence' and 'repetition', and have some experience of using 'selection'

Key Knowledge I need to understand

I need to understand that:

Programming is when we make and input a set of instructions for computers to follow.

Microcontrollers are devices that can be programmed to control output devices that are connected to them.

We use algorithms which we can plan, model, trial and debug, in order to create accurate command sequences, involving multiple output devices (e.g. LEDs and motors).

learners will use physical computing to explore the concept of selection in programming through the use of the Crumble programming environment. Learners will be introduced to a microcontroller (Crumble controller) and learn how to connect and program it to control components (including output devices — LEDs and motors). Learners will be introduced to conditions as a means of controlling the flow of actions in a program. Learners will make use of their knowledge of repetition and conditions when introduced to the concept of selection (through the 'if...then...' structure) and write algorithms and programs that utilise this concept. To conclude the unit, learners will design and make a working model of a fairground carousel that will demonstrate their understanding of how the microcontroller and its components are connected, and how selection can be used to control the operation of the model. Throughout this unit, learners will apply the stages of programming design.

How I will show what I have learned

To control a simple circuit connected to a computer	- I can create a simple circuit and connect it to a microcontroller - I can program a microcontroller to make an LED switch on - I can explain what an infinite loop does
To write a program that includes count-controlled loops	- I can connect more than one output component to a microcontroller - I can use a count-controlled loop to control outputs - I can design sequences that use count-controlled loops
To explain that a loop can stop when a condition is met	- I can explain that a condition is either true or false - I can design a conditional loop - I can program a microcontroller to respond to an input
To explain that a loop can be used to repeatedly check whether a condition has been met	- I can explain that a condition being met can start an action - I can identify a condition and an action in my project - I can use selection (an 'if…then…' statement) to direct the flow of a program
To design a physical project that includes selection	- I can identify a real-world example of a condition starting an action - I can describe what my project will do - I can create a detailed drawing of my project
To create a program that controls a physical computing project	- I can write an algorithm that describes what my model will do - I can use selection to produce an intended outcome - I can test and debug my project

What vocabulary I need to know

Microcontroller, components, connection, infinite loop, output component, motor, repetition, count-controlled loop, Crumble controller, switch, motor, LED, Sparkle, crocodile clips, connect, battery box, program, condition, Input, output, selection, condition, action, repetition, debug.

The following Glossary may be useful https://icompute-uk.com/ewExternalFiles/iCompute-Glossary.pdf

What's next

In **Year 5 – Selection in Quizzes – Summer 2** pupils develop their knowledge of 'selection' by revisiting how 'conditions' can be used in programming, and then learning how the 'if… then… else...' structure can be used to select different outcomes depending on whether a condition is 'true' or 'false'. They represent this understanding in algorithms, and then by constructing programs using the Scratch programming environment. They learn how to write programs that ask questions and use selection to control the outcomes based on the answers given. They use this knowledge to design a quiz in response to a given task and implement it as a program. To conclude the unit, learners evaluate their program by identifying how it meets the requirements of the task, the ways they have improved it, and further ways it could be improved.

Assessment

National Curriculum Computing links

- Design, write, and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts
- Use sequence, selection, and repetition in programs; work with variables and 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
- 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.

Cross Curricular links

Science – Electricity (Year 4)

- Construct a simple series electrical circuit, identifying and naming its basic parts, including cells, wires, bulbs, switches, and buzzers

Design and Technology (Key stage 2)

Design

- Generate, develop, model, and communicate their ideas through discussion, annotated sketches, cross-sectional and exploded diagrams, prototypes, pattern pieces, and computer-aided design

Make

- Select from and use a wider range of tools and equipment to perform practical tasks [for example, cutting, shaping, joining, and finishing], accurately
- Select from and use a wider range of materials and components, including construction materials, textiles, and ingredients, according to their functional properties and aesthetic qualities

Evaluate

- Evaluate their ideas and products against their own design criteria and consider the views of others to improve their work

Technical knowledge

- Understand and use electrical systems in their products [for example, series circuits incorporating switches, bulbs, buzzers, and motors]
- Apply their understanding of computing to program, monitor, and control their products

Assessment

Formative assessment opportunities are provided throughout each of the lesson plan documents. The learning objectives and success criteria are introduced in the slide decks at the beginning of each lesson and then reviewed at the end.

Summative assessment – the assessment rubric document should be used to assess student's work from lesson 6. The rubric should be completed digitally and stored in individual pupil folders and then used alongside teacher judgement to complete ScholarPack.
https://education.lego.com/en-gb/product-resources/wedo-2/teacher-resources/teacher-guides

Teacher Subject Knowledge
You will need experience of constructing programs using the Crumble programming software. It uses the same drag-and-drop style as Scratch. You will need to write programs that turn LEDs (Sparkles) on and off, change LED colours, spin motors, use push switches as inputs, and combine a number of these components. Additionally, you will connect the Crumble controller to battery packs, Sparkles, motors, and push switches. For further support on using Crumbles, see the Crumble 'Getting Started' guide at redfernelectronics.co.uk/crumble-getting-started.

Levels of abstraction
When programming, there are four levels that can help describe a project (known as 'levels of abstraction'). Research suggests that this structure can support learners in understanding how to create a physical computing project or standalone program and how it works:

- Task — this is what is needed
- Design — this is what it should do
- Build — this is how it is done
- Running the code — this is what it does

Spending time at the 'Task' and 'Design' levels before engaging in writing code aids learners in assessing the 'do-ability' of their programs and reduces a learner's cognitive load during programming. Learners will move between the different levels throughout the unit, and this is highlighted within each lesson plan.

Repetition
You will need to know that repetition is used in programming to give the same instruction or set of instructions several times. Repetition uses loops as the means to give these instructions. This unit makes use of two types of loops: infinite and count-controlled. These have been defined below.

Infinite loop
An infinite loop is a loop that commands the instruction/set of instructions to repeat forever. When an infinite loop is used in a program, there is no way of ending the program, as the command(s) within the loop will be repeated endlessly. For this reason, infinite loops should only be used when writing a program that is intended to run forever. The exception to this is when using selection in physical computing, as you will see throughout this unit.

Count-controlled loop
A count-controlled loop is a form of repetition in which a set of commands are carried out a specific number of times. Count-controlled loops should only be used when it is known how many times a set of commands needs to be repeated.

Condition-controlled loop
A condition-controlled loop is a form of repetition in which a set of commands stop being carried out when a condition is met. The condition could be anything from when the 'score' in a game reaches a certain value to when a key on a keyboard has been pressed.

Conditions
Conditions are statements that need to be met for a set of actions to be carried out. They can be used in algorithms and programs to control the flow of actions. When a condition is met, it is referred to as 'true' and when it is not met, it is referred to as 'false'. You will need to be able to identify and use conditions in algorithms in the form of statements to both start and stop sets of action. Additionally, you will need to understand that conditions can be used in loops, and when they are, that the set of actions in the loop will be carried out repeatedly until the condition is true, for example, 'until button A is pressed'.

Selection
Selection is "part of a program where, if a condition is met, then a set of commands are run".

Selection is implemented in programming using if...then... statements. Selection is used to control the flow of actions in algorithms and programs by checking if a condition (see above) has been met. If it has been met, the identified actions will be carried out. When selection is used in programs, loops (see above) often have to be used to instruct the device to check the condition repeatedly. Without using loops, the condition would only be checked once. It's important to understand that each loop cycle will complete before the condition is checked again. In the Crumble programming software, selection is implemented through the if...then... command block.

In addition to the above, you will also need to understand that programs are an implementation of an algorithm, and that when the program does not produce the required output, the algorithm should be debugged. This should then be implemented in the program