**Year 1**

**Use Bee-Bots for practical programming**

**Knowledge objectives (from CAS: A curriculum for schools)**

* Algorithms are sets of instructions for achieving goals
* Algorithms can describe everyday activities and can be followed by humans and by computers.
* Computers need more precise instructions than humans do.
* Computers are controlled by a sequence of instructions.
* A computer program is like the narrative part of a story, and the computer’s job is to do what the narrator says. Computers have no intelligence, and so follow the narrator’s instructions blindly.

**Knowledge objective notes/explanations -** *The knowledge objectives above should be covered across the two programming units in Year 1.* Pupils should become familiar with the term **algorithm** and understand that it means **a series of instructions**, which can be introduced in Unit 1: Bee Bot and recapped in unit 2: Daisy the Dino. In either or both units they should practise following algorithms themselves (*making sandwiches perhaps or moving around a room*) and in doing so recognise the need for algorithms to be precise. Across both units they should develop an understanding that computers operate by following instructions precisely as they have no intelligence - this can be shown since both programs precisely follow the sequence of instructions entered by the user.

**Unit 1: Bee Bots (app)**

* Give and follow instructions, which include straight and turning commands, one at a time.
* Explore outcomes when instructions are given in sequence.
* Give a simple sequence of instructions.

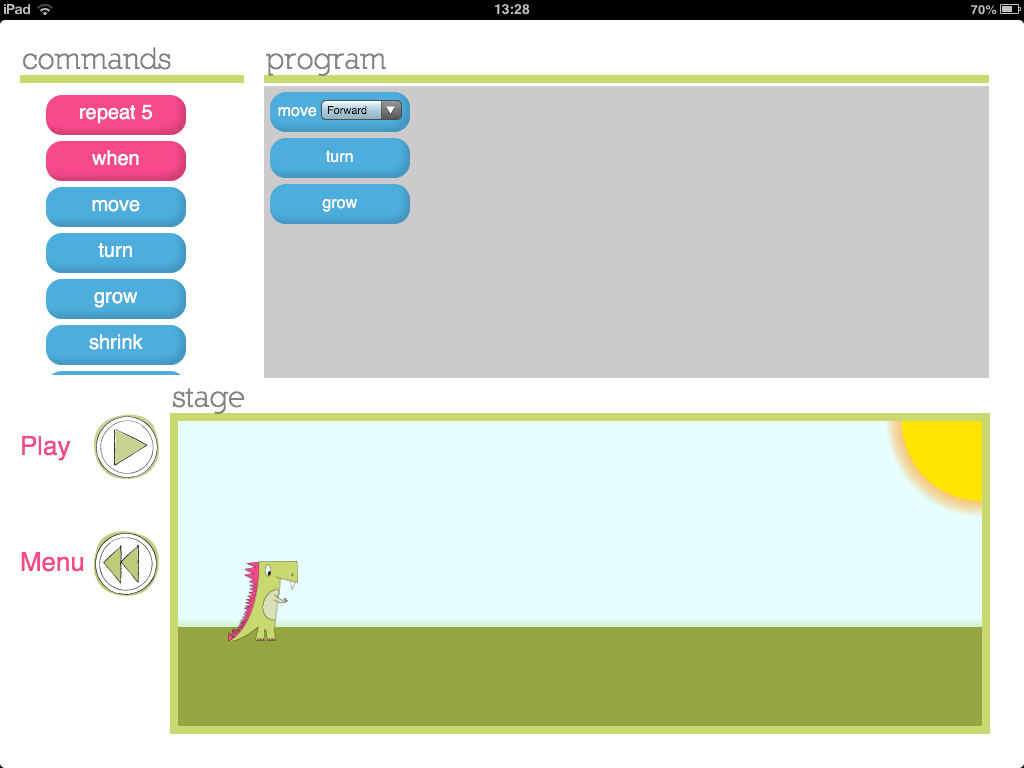
It is envisaged that a series of lessons will start with children giving and receiving commands to each other. These will be given one at a time and include turning and straight movements. Whilst this activity could initially be open-ended, it could be progressed so children have to reach a desired location. Children could discuss how there may be more than one route that can be taken and discuss the advantages/disadvantages of different routes – can they find the shortest route? Finding the shortest route is a good introduction to programming, since there are often different **algorithms (step-by-step procedures)** that can be used to solve a problem but some are simpler more efficient solutions than others. If obstacles are also introduced then children will have to alter their instructions to avoid these. Progressing from these activities, children can use the Bee Bots or the Bee Bot app,  which can be found on all the iPads, to start to explore the outcomes when a sequence of instructions are given (this could be incorporated in the previous activities if one child writes a sequence of instructions and hands this to the other child). The Bee Bot app has a series of levels, which require the pupil to enter a sequence of instructions so the Bee Bot reaches the flower.

Coding Challenge: **How many levels can pupils successfully complete on the Bee-Bot app? More points are scored for quicker more efficient solutions, so who can score the highest**

**Unit 2: Daisy Dino/Bee Bots (app)**

* Discuss/explore what will happen when instructions are given in a sequence.
* Give a sequence of instructions to complete a simple task.
* Instructions use both movement commands and additional commands.

The Daisy the Dino app  allows the user to program Daisy using a series of commands which includes movements as well as changes in appearance. The app allows Daisy to be programmed by simply dragging and dropping command blocks into the program window. This ‘style’ of programming (called GUI – Graphic User Interface) is also used in other software, e.g. Scratch, which pupils will encounter as they progress through KS1 and KS2. When a sequence of (or just one) command(s) is dragged into the program window, the program can be ‘run’ by clicking the ‘Play’ button and Daisy will execute the program in the display window.



List of commands available to program Daisy

Click play to run program

Daisy will execute program in this display window

Drag command blocks into the program window

In this second unit on programming, it may be worth initially recapping activities from Unit 1 on giving and following a sequence of instructions. It is then intended that the use of a different app with a different method of programming (i.e. using drag and drop command blocks) will provide pupils with experience of a different programming environment. Initially pupils may be given the opportunity to have a ‘free play’ at programming Daisy, exploring and talking about the outcomes. In a similar manner to the Bee Bot app, the Daisy the Dino app also has a series of challenges where the desired outcome of Daisy is specified and the children have to generate the program to create this outcome. This way of working, i.e. coding to produce a desired outcome, underpins the problem-solving nature of coding at higher levels so should form a significant part of the work at KS1. Children could work together to storyboard what they would like Daisy to do then code for this for example, or challenge each other to write code so, ‘Daisy does……’ etc. To continue to extend learning within Year 1, unlike the Bee Bot app, the Daisy the Dino app includes commands beyond simple movements so coding sequences can be more lengthy and complex.

Coding Challenge: **Can you write an algorithm to make Daisy dance? Draw a storyboard first to show the dance moves you would like Daisy to do then write a program for her – do the moves match?**

**Year 2**

**Use Bee-Bots for practical programming**

**Knowledge objectives (from CAS: A curriculum for schools)**

* Steps can be repeated within algorithms

* Algorithms can be represented in simple formats [storyboards and narrative text]
* Particular tasks can be accomplished by creating a program for a computer. Some computers allow their users to create their own programs.

**Knowledge objective notes/explanations -** *The knowledge objectives above should be covered across the two programming units in Year 2.* Pupils should be aware that steps can be repeated in algorithms which they will encounter when using the repeat command during unit 1: Daisy the Dino. They may also use the repeat function to draw geometric shapes with unit 2: Move the turtle. During Unit 1: Daisy the Dino, pupils should practise representing their algorithms in different formats using a storyboard or narrative to say what they want Daisy to do before coding for this outcome. From unit 2: Move the turtle, pupils should start to become aware that we can create programs to accomplish particular tasks such as drawing geometric shapes or other patterns.

**Unit 1: Daisy the Dinosaur (app)**

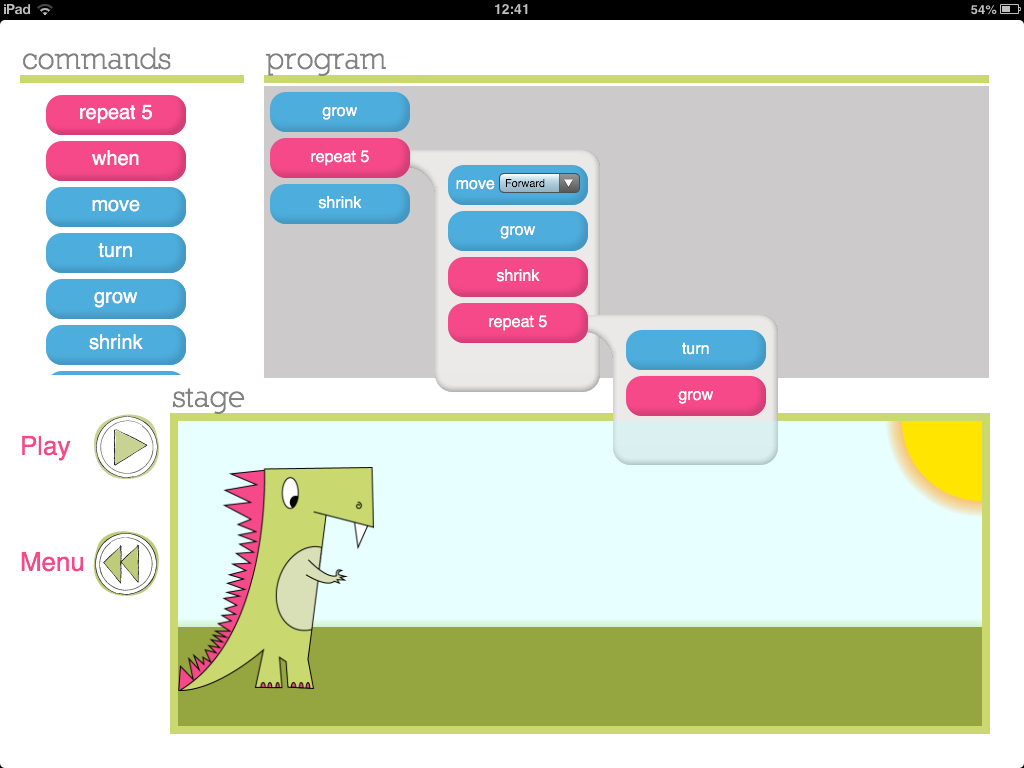
* Use the ‘repeat’ (loop) and ‘if….then’ (conditional statement) command within a series of instructions.
* Plan a short ‘story’ for a sprite and write the commands for this.
* Edit/refine a sequence of commands.

In Year 2, this first programming unit returns to an app pupils used in Year 1 called Daisy the Dinosaur. Please read the section above under Year 1 for an introduction to the app and details of the learning of Year 1 pupils. In Year 2, pupils learn to use the ‘repeat’ and ‘if… then’ command, as well as starting to develop their knowledge and ability to ‘debug’ programs.

The repeat command in programming can be used to shorten code. For example, if we want Daisy to: ‘move forward, turn, move forward, turn, move forward, turn’. We could use the repeat function to ‘Repeat 3 times: move forward, turn’. Note - within the Daisy the Dino app, the number of times that a command is repeated is fixed at 5.

**Using the repeat command within Daisy the Dino**

The screenshot below shows the use of the repeat command within Daisy the Dino. As you can see, when the repeat block is dragged into the program window, the block produces a ‘speech box’ style window to the right, into which the commands to be repeated can be dragged.



The commands within the repeat ‘bubble’ will be repeated

Note – repeat commands can be placed within repeat commands (**nested**).

As you can see from the screen shot above, commands can come before or after the repeat command in the program window. When the program is run, Daisy will follow the commands before repeat, then repeat the commands within the repeat block, before moving onto the commands after the repeat block. A significant (and quite advanced) function of the Daisy the Dino app is the ability to place the repeat command within the repeat command (termed nesting within programming language). This is also shown in the screenshot above.

Within this unit, it is envisaged that teaching time will initially be spent briefly recapping how to program Daisy using the commands introduced in Year 1, before introducing pupils to the ‘repeat’ function. The repeat function could be modelled with pupils themselves following a sequence of commands, which include the repeat command, and/or it can be shown on the Daisy the Dino app. When modelling with the app it is good to get the pupils predicting the outcome of a program you have written, then playing the program and comparing Daisy’s actions with what was predicted. A key point to convey when introducing the repeat command is that it can be used to ‘shorten code’ – this will be emphasised within the skill: *edit/refine a sequence of commands (see below)*

To further extend learning using this app in Year 2, pupils are introduced to the notion of an ‘input’ to trigger a command in programming. This is fundamentally different to the programming they have completed up to this point, which has focused on writing a sequence of commands then sitting back and watching the result when the command sequence is executed.

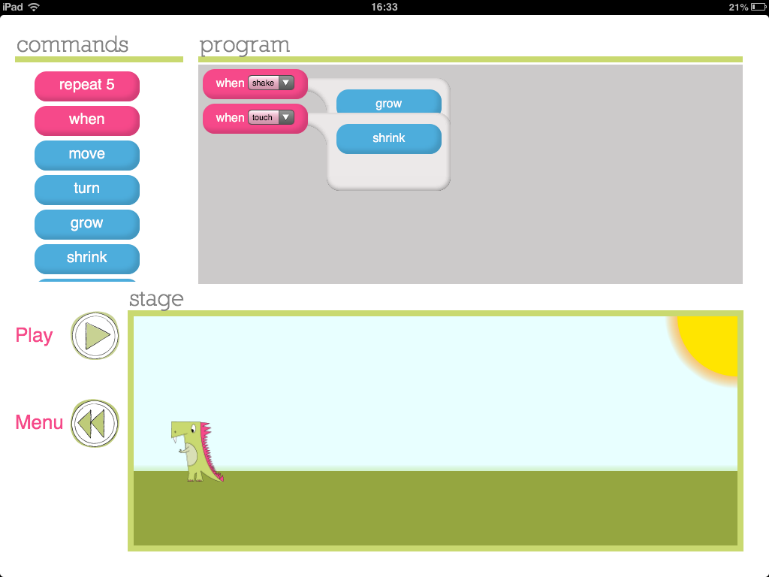
In this unit, in addition to a sequence of commands being executed, the program written will also feature an input to trigger a command. This will enable pupils to interact with the program they have written as it is running. This will form the basis of pupils going on to write programs for controllable games in subsequent years.

When introducing the notion of a trigger within programing, it may be best to use role-play to model the effect. For example, you may ask a pupil to execute a repeated series of commands similar to those that we can give to Daisy e.g. (move forward, turn, move forward, turn… etc) and say that every time you tap the table they are also going to jump. Having completed this activity you could explain to pupils that in this imaginary program there was an input triggering a command and ask them what that input was? You could then ask pupils what types of inputs are used with a computer (as we don’t tap the table to input commands!). Pupils should be able to express different ways in which we interact with computers – touch screen, mouse, keyboard, games controller for example.

**Using triggers in Daisy the Dino**

Introduce the ‘when’ command for Daisy (as shown in the screenshot below). Explain that using this piece of code enables us to tell Daisy to do something ‘when’ something happens. Referring back to the role-play example then it is the same as: **When** we tapped the table, the pupil jumped. In exactly the same way we can tell Daisy what to do for two different input commands on the iPad: these are ‘**touch’** and ‘**shake’**.

Below is a good example of a simple program that could be written with the class using ‘when’ commands in Daisy the Dino - as can be seen, the program written makes Daisy grow every time the iPad is shaken and shrink when the screen is touched.



Use of two ‘when’ commands allows the two different trigger inputs to control Daisy in two ways

After introducing the repeat and ‘if..then’ function, pupils should gain experience of programming using this two important command. Such experience could take a 3 stage approach i.e. exploring outcomes; programming for a desired outcome, and; debugging a program. In this 3-stage approach, pupils could initially have time exploring the use of the functions. They can predict what will happen from code they have written and talk about whether this differs from Daisy’s actions when the code is executed – why are there differences? Can they debug their code so it produces the desired outcome?

Coding Challenge: **Can you debug and improve the code?**

A key skill for pupils to develop within programming is the ability to critically evaluate a program and amend it so that it achieves, or better achieves, the purpose for which it was written.

****

For this coding challenge, produce a storyboard including text for pupils detailing what you would like Daisy to do (including the use of if…then commands) when the program is run and incorrectly code a solution. **Can pupils debug and change your program so Daisy’s actions match your storyboard?** Also make your code deliberately long-winded and inefficient – **Can pupils improve your code using the repeat function?**

**Unit 2:**

Course 1 from Code.org

<http://studio.code.org/s/course1>