Year 1 : Algorithms through Bee-Bots

Program Aim: Children learn how to construct algorithms through programming Bee-Bots with instructions to meet a specific goal.

Programming Concepts
-Sequence

Computational Thinking
-Algorithmic Thinking

Handles Ambiguity

Open Ended Problem Solver

Investigates

Persists

Communicates

Adapts

Evaluates

National Curriculum Programs of Study

Pupils should be taught to:

- understand what **algorithms** are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions
- **create** and **debug** simple programs
- use **logical reasoning** to predict the behaviour of simple programs

Computational Thinking
Algorithm is a precise set of instructions or rules to achieve an outcome or solve a problem.

Previous learning suggested for Reception: http://code-it.co.uk/wp-content/uploads/2016/06/guidedplayreception.pdf

Learning path:

Session 1: The Hook

Session 2: Bee-Bot commands

Session 3: Algorithm to code

Session 4: Symbol algorithm

Session 1: The Hook

Decide on a context which links to a project you are doing. Geography or local history are popular but it could be anything. If pupils have not had directed play opportunities in reception, spend two lessons rotating around these activities. Pupils who haven't had open-ended play activities are at a disadvantage when it comes to logical reasoning with the Bee-bot.

Drawing Bee-Bot World: Draw a grid of 15cm2 squares across a large sheet of sugar paper before the lesson. Use the Bee-Bot World slides to help pupils design and create their own world. Use your own theme locations, not the ones of the slides. Pupils work in pairs to design Bee-bot world.

Session 2: Bee-Bot commands

Input	Steps to success	Corrections needed
 Can the children tell you what each of these buttons does? Can they describe these to a partner? Children work in pairs to create a short video to explain what each button does on the Bee-bot. You could give them the context of an instructional video for reception children. Demonstrate what makes a good video.	• Record in good light. Avoid windows as backgrounds as these darken the video foreground. • Record with the camera person facing towards a wall so no one passes behind their video. • Keep the recording device (iPad) still. Tripods are available. • Frame the face and Bee-bot in the picture. • Keep the iPad close to the person speaking so the sound is clearly picked up.	• Demonstrating the Bee-bot moves as this is about explanation not demonstration. • Recording where it is too noisy.

Session 3: Algorithm to code

Session 4: Symbol algorithm

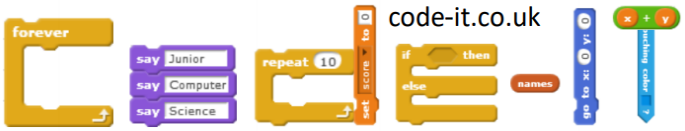
From A to B lesson 1	From A to B creating symbol algorithm first lesson 2 & 3
Can students program the Bee-bot to move from A to B on their maps? Pupils are creating an algorithm in their heads before programming the digital device. One pupil decides on the start and finish point of the route. The other programs this into the Bee-Bot. Once the green GO button is pressed no one is allowed to touch the Bee-bot until it stops. This encourages pupils to be more careful in their programming.	Before the lesson make sure you have sets of Bee-bot algorithm cards printed and cut up and that you have your Bee-bot worlds created. Pupils work in pairs. One pupil creates an algorithm to go from A to B by ordering the cards sequentially. The other pupil converts this algorithm into code by punching the instructions into the Bee-bot Instruct pupils to ONLY press buttons if there is an algorithm card. Often pupils will press GO or clear when there is no GO or clear card in the algorithm.
Success? Teacher observation	
Can they move onto extending their algorithm? A to B to C or moving from A to C avoiding B?	

Debugging

A really good strategy for making the link between **algorithm** and programming and to aid debugging is to encourage pupils to read out their cards as the turtle performs its instructions. This makes it much easier for pupils to spot errors called bugs in programming. This is similar to programmers tracing their more complex code to work out what it does before testing it on a digital device.

Session 4: Symbol algorithm

Some pupils find it very helpful to place the cards on the route the turtle will take instead of alongside the mat.



Where next?

Explore other programming planning at code-it.co.uk/csplanning
Design your own programming planning using computational thinking & problem solving skills.

Author: Philip Bagge
Find more resources at code-it.co.uk

Follow me on twitter @baggiepr

Designer: Amanda de Fraine adefraine@hotmail.com

Follow me on LinkedIn

Complexity	I can break complex problems into parts	I can discover / concentrate on the most important part of a problem	I can explain how I used decomposition & abstraction
Ambiguity	I recognise there is more than one way to solve a problem	I recognise there is more than one way to describe a problem	I can explain how I managed ambiguity
Open Ended	I look for a range of solution to the same problem	I don't just accept the first solution	I can describe how a project can be extended
Adapt	I can adapt existing ideas to solve new problems	I can identify patterns in problems & solutions	I can explain how I adapted a solution to solve a new problem
Evaluate	I can evaluate my solutions against a set criteria	I can design criteria to evaluate my creations	I can explain how evaluation helped me improve a project
Experiment & Debug	I can develop, test and debug until a product is refined	I repeatedly experiment through making, testing & debugging	I can explain how using the iterative cycle improves my work
Persist	I can persevere even if the solution is not obvious	I learn from setbacks and don't let them put me off	I can describe how I overcame problems
Communicate	I can contribute useful ideas to a partner or group	I can encourage others to share their ideas	I can lead using all the people talent in my group

Decomposition Abstraction Generalisation Algorithmic Evaluation Algorithm

Adapted from Behaviour Rubric created with @MarkDorling and linked at http://code-it.co.uk/attitudes/ Adapted with permission

This work is licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/3.0/ or send a letter to Creative Commons, 444 Castro Street, Suite 900, Mountain View, California, 94041, USA