Unit 1.1: We are treasure hunters

Solving problems using programmable toys

Software: Blue-Bot app (optional, alternatives: programming interface for alternative toys)

Hardware: Blue-Bot (alternatives: Cubetto, Bee-Bot, Roamer Too, STEM Robot Mouse). If robot toys are not available the Blue-Bot app or the Scratch Bee-Bot emulator can be used instead.

Overview

In this unit, pupils learn basic programming ideas through experimenting and solving problems with simple, programmable robots, such as the Blue-Bot. In:

- **Session 1** they take on the role of a **robot**, responding to instructions they are given
- **Session 2** they take on the role of a robot-pirate to work out a sequence of instructions (an **algorithm**) to find their way to an objective
- **Session 3** they explore the Blue-Bot controls
- **Session 4** they follow, create and test sequences of instructions to solve problems with the Blue-Bot
- **Session 5** they predict what the Blue-Bot will do when given different sequences of instructions
- **Session 6** they correct mistakes in Blue-Bot **programs**.

Alternatives

Sessions 3–6 give step-by-step guidance to using a Blue-Bot to carry out the activities. This unit could also be carried out with a little modification using:

- other programmable toys (see *Hardware* above)
- an on-screen emulator of a programmable toy, such as the Blue-Bot app or a Bee-Bot emulator in Scratch.

Knowledge, skills and concepts

In this unit, pupils will learn:

- that a programmable **robot** can be controlled by inputting a sequence of instructions
- to develop and record sequences of instructions as an **algorithm**
- to program a robot to follow their algorithm
- to predict how their **programs** will work
- to **debug** programs.

Progression

In Key Stage 1:

- Pupils will build on their knowledge of algorithms in **Unit 1.2: We are TV chefs**.
- They will build on their programming work in **Unit 2.1: We are astronauts**.

In Key Stage 2:

- Pupils are introduced to more complex programming ideas, including selection and repetition.

Assessment – by the end of the unit:

All pupils can:

- follow instructions to move around a large space
- record a set of instructions for a Blue-Bot
- **program** a Blue-Bot to move by giving one instruction at a time
- program a Blue-Bot to move by giving a sequence of instructions.

Most pupils can:

- give other pupils instructions to move around a large space
- understand **input**, **program** and **output** in the context of a Blue-Bot
- create a program to move a Blue-Bot to a particular location
- **debug** a Blue-Bot program.

Some pupils can:

- predict where a set of instructions will take someone moving in a large space
- predict where a Blue-Bot will end up from a set of instructions
- understand input, program and output in more general contexts
- look for ways in which a Blue-Bot program could be made more efficient.

Background information

- In creating step-by-step instructions, pupils are creating **algorithms**. Programming involves converting these instructions into a formal language understood by the **computer** – in this instance, a series of button presses on a programmable **robot**. The robot follows the instructions in the order they were given. This essential idea – of the computer following the instructions it is given in order – will form the basis of all the programming that pupils do at primary school and beyond.
- When looking at others' algorithms, pupils should have a clear idea of what the **program** will do by using **logical reasoning** to predict what will happen from the instructions.
- Sometimes, pupils' programs will not work as planned and will contain **bugs**. Pupils will need to correct (**debug**) their programs to fix mistakes.

Key vocabulary

Algorithm: a sequence of precise instructions or steps (sometimes a set of rules) to achieve an objective

Bug: an error or mistake in a program or algorithm, causing the computer or robot to behave in a manner that was not originally intended

Computer: a device that accepts input, processes it according to instructions or rules and produces output

Debug: correct mistakes in a computer program or algorithm

Input: data supplied to a computer, in this case, pressing buttons on the robot

Logical reasoning: to be able to give a reason for something which others would have to accept as correct

Output: information produced by a computer – in this case, movements of the robot

Program: a sequence of instructions (or sometimes a set of rules) that can be followed by a computer

Robot: a computer that can move, or that can move part of itself

Differentiation

See each session (pages 13–18) for ways to increase support and add challenge to this unit.

Some pupils may need support in recording their instructions and may also need to **program** their toy with shorter and/or less complicated sequences of instructions. If pupils find the idea of a sequence of instructions challenging, Blue-Bot can be given one instruction at a time, in a more interactive mode. It would still be worth getting pupils to plan their whole sequence of instructions (their **algorithm**) in advance.

It can be helpful to give some (or all) pupils printed instructions cards to use to work out their programs. It can also be useful for pupils to work with a 'fake bot' cut out of a Blue-Bot from above, to test out their instructions.

Cross-curricular opportunities

English: Pupils provide clear instructions for moving around a map. Pupils could read stories about pirates, or have some pirate stories shared as class texts.

Geography: Pupils use geographical language and could use and make their own maps. They could learn about places where pirates would have been found.

History: Pupils could learn about piracy in the past.

Maths: Pupils recognise movements in a straight line (translations) and rotations, and combine them in simple ways. They start to recognise and make whole, half and quarter turns, and learn to recognise a right angle.

PSHE and citizenship: Pupils could consider the effect that pirates' actions had on others.

Preparation for teaching the unit

Things to do

- Check you have access to Blue-Bots or other programmable toys (see *Alternatives* on page 10) and that their batteries are charged.
- If you are using the Blue-Bot app, add it to the devices the pupils will be using and check that Blue-Bot connects to the app.
- Read pages 10–11 to get an overview of the unit.
- Read the steps in the unit sessions (pages 13–18) and look at the associated online resources, printing out the worksheets as required.
- Watch the CPD video for this unit, which gives some background knowledge on the basics of programming.
- Work through the unit yourself, so you know what is expected of the pupils.

Resources needed

- **Software:** Optionally, the Blue-Bot app or an emulator of a programmable toy (one is provided in the online resources)
- **Hardware:** Blue-Bots (enough for each group/pair of pupils) or equivalent programmable toys (see *Alternatives* on page 10)
- The Treasure Island map (Worksheet 1.1c) drawn out on a grid. Its size should equate to the distance your toy moves in one step, e.g. the Blue-Bot moves in 15 cm steps, so the 5 × 5 grid should measure 75 cm × 75 cm
- Clipboards, paper and pencils

Online resources provided

Session resources

- Worksheet 1.1a: Instruction cards
- Worksheet 1.1b: How does a Blue-Bot work?
- Worksheet 1.1c: Map of the Treasure Island
- Worksheet 1.1d: 'Fake bot' template
- Worksheet 1.1e: Buggy programs 1
- Worksheet 1.1f: Buggy programs 2
- Worksheet 1.1g: End-of-unit quiz
- Worksheet 1.1h: Pupil self-assessment
- Teaching slides 1.1a–1.1e
- Interactive end-of-unit quiz 1.1

Additional resources

- CPD video: Introduction to programming

Alternatives

Scratch file 1.1: Treasure Island Bee-Bot emulator can be used instead of the physical robots: **scratch.mit.edu/projects/20050141/editor**

Online safety

- If pupils use the Blue-Bot app or an online emulator, ensure the usual appropriate safeguards are in place concerning Internet access.
- If pupils are filmed working with the robots, ensure any necessary permission has been obtained.

Collaboration

It is best to have pupils work in pairs or a small group for programming projects, ideally swapping roles regularly, so each pupil has a turn in the different roles.

Useful links

Online tutorials

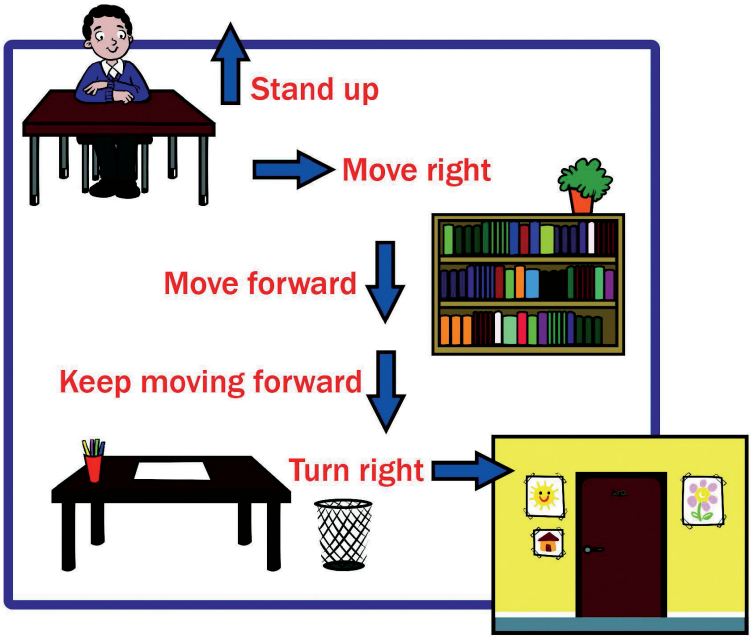
- Blue-Bot introduction: **www.youtube.com/watch?v=T6SyP7Imygs**
- Blue-Bot app on the App Store.
- Bee-Bot a class introduction: **www.youtube.com/watch?v=52ZuenJlFyE**
- Roamer Too introduction: **www.vimeo.com/49152214**
- Cubetto introduction: **www.youtube.com/watch?v=Kp1p2lh2D64**
- STEM Robot Mouse introduction: **www.youtube.com/watch?v=_tGb9bLe0YA**

Information and ideas

- Barefoot Computing has a free collection of Bee-Bot-based lesson plans: **www.barefootcomputing.org/primary-computing-resources**
- An activity using a more complex map and more sophisticated robot: **www.bbc.co.uk/programmes/p01661yg**
- Bee-Bot collaborative dance: **www.learningismessy.com/coding/bee-bot-collaborative-dance**

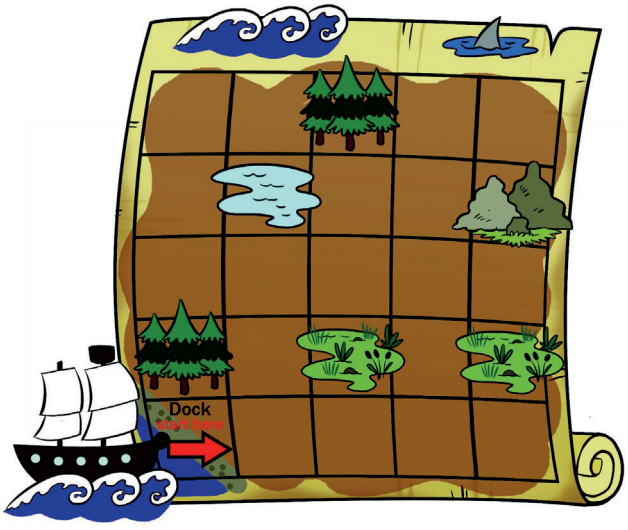
Unit outcomes

Below are some examples of the outcomes you could expect from this unit.

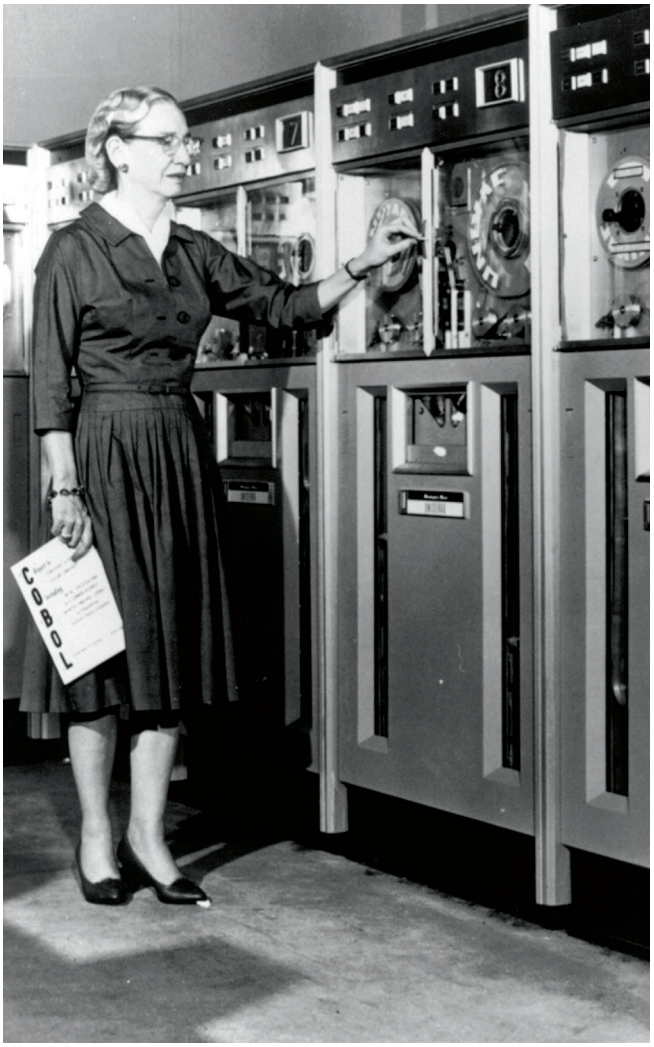


Sessions 1 and 2: Can pupils put together a set of instructions to get from their seat to the door or to direct a pirate-robot to the treasure? Do they understand this is an example of an algorithm?

Session 3: Learning about the Blue-Bot



Sessions 4 and 5: Using the treasure map



Session 6: Learning about Grace Hopper and how her team coined the term 'debugging'