Year 6 – Programming B – Sensing

Unit introduction

This unit is the final KS2 programming unit and brings together elements of all the four programming constructs: sequence from Year 3, repetition from Year 4, selection from Year 5, and variables (introduced in Year 6 – 'Programming A'. It offers learners the opportunity to use all of these constructs in a different, but still familiar environment, while also utilising a physical device — the micro:bit. The unit begins with a simple program for learners to build in and test in the programming environment, before transferring it to their micro:bit. Learners then take on three new projects in Lessons 2, 3, and 4, with each lesson adding more depth.

Design features prominently in this unit. A design template is introduced in Lesson 3, initially scaffolded to give learners the opportunity to create code from a given design. In Lesson 4 that scaffolding is gradually reduced, then in Lesson 5, learners create their own design, using the same template. In the final lesson, learners will apply their knowledge of the programming constructs and use their design to create their own micro:bit-based step counter.

There are two Year 6 programming units:

- Programming A – Variables in games
- Programming B – Sensing

This is unit B, which should be delivered after unit A.

Overview of lessons

Lesson	Brief overview	Learning objectives
The micro:bit	In this lesson, learners will be introduced to the micro:bit as an input, process, output device that can be programmed. Learners will familiarise themselves with	To create a program to run on a controllable device

	the device itself and the programming environment, before creating their own programs. They will then flash their programs to the device.	• I can apply my knowledge of programming to a new environment • I can test my program on an emulator • I can transfer my program to a controllable device
Go with the flow	In this lesson, learners will explore how if, then, else statements are used to direct the flow of a program. They will initially relate if, then, else statements to real-world situations, before creating programs in MakeCode. They will apply their knowledge of if, then, else statements to create a program that features selection influenced by a random number to create a micro:bit fortune teller project.	To explain that selection can control the flow of a program • I can identify examples of conditions in the real world • I can use a variable in an if, then, else statement to select the flow of a program • I can determine the flow of a program using selection
Sensing inputs	In this lesson, learners will initially use the buttons to change the value of a variable using selection. They will then develop their programs to update the variable by moving their micro:bit using the accelerometer to sense motion. Finally, they will learn that a variable can be displayed after it is updated or in response to an input.	To update a variable with a user input • I can use a condition to change a variable • I can experiment with different physical inputs • I can explain that if you read a variable, the value remains
Finding your way	In this lesson, learners will initially work at code level by applying their knowledge from the previous lesson to make their micro:bit perform the function of a compass. They will then design a program which will enable the micro:bit to be	To use an conditional statement to compare a variable to a value

	used as a navigational device. To code this, they will adapt the code they completed to make the compass.	• I can explain the importance of the order of conditions in else, if statements • I can use an operand (e.g. <>=) in an if, then statement • I can modify a program to achieve a different outcome
Designing a step counter	In this lesson, learners will be working at the design level. They will pick out features of a step counter, a piece of technology with which they are likely to be familiar. They will then relate those features to the sensors on a micro:bit. Having seen a simulated example of a micro:bit step counter, learners will pick out features which they will be able to include in their design. In the main activity, learners will design the algorithm for their step counter project. Finally, they will connect the battery pack to their micro:bit to set it up as a portable device.	To design a project that uses inputs and outputs on a controllable device • I can decide what variables to include in a project • I can design the algorithm for my project • I can design the program flow for my project
Making a step counter	In this lesson, learners will use the design that they have created in Lesson 5 to make a micro:bit-based step counter. First they will review their plans, followed by creating their code. Depending on their level of confidence, they can use a scaffolded or part-complete project, otherwise they can start a new project. Learners will test and debug their code, using the emulator and then the physical device. To successfully complete this project, learners will need to use all four programming constructs: sequence, repetition, selection, and variables.	To develop a program to use inputs and outputs on a controllable device • I can create a program based on my design • I can test my program against my design • I can use a range of approaches to find and fix bugs

Resources

This unit of work is based around the micro:bit. It has been designed to be taught with the physical computing device and this is how it will be most effective. However, the makecode.microbit.org website has an emulator (an interactive, on-screen micro:bit) that schools can use if micro:bits are unavailable.

The micro:bit will need the following peripherals:

- A micro USB to USB lead
- A battery pack
- 2 x AAA batteries per micro:bit (if you are using your own micro:bits, rather than those provided in the NCCE hub kits, check the battery size — some are AA)

This unit uses a desktop computer and the makecode.microbit.org website to program the micro:bit. To flash the micro:bit code onto the micro:bit, support is provided to download the code to the computer being used and then transfer it to the micro: bit. Direct download is supported by the micro:bit website, which simplifies the code download process above, however it needs some prior set up. This is not a requirement, but there is support on the micro:bit website for further information: https://support.microbit.org/support/solutions/articles/19000105428-webusb-troubleshooting

Progression

This unit presumes that learners are already confident in their understanding of sequence, repetition and selection independently within programming. If learners are not yet ready for this, you may wish to revisit earlier programming units where these constructs are introduced.

Please see the learning graph for this unit for more information about progression.

Curriculum links

National curriculum 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

Assessment

Formative assessment

Assessment opportunities are detailed in each lesson plan. The learning objectives and success criteria are introduced in the slide deck at the beginning of each lesson, and then reviewed at the end. Pupils are invited to assess how well they feel they have met the learning objective using thumbs up, thumbs sideways, or thumbs down.

We recommend that teachers collect the programming work which the learners complete either by learner's sharing the URLs with their teacher when they select 'Share' and 'Publish Project' or by downloading the code file and saving it on the school's computer system. This will aid assessment throughout this unit.

To open a downloaded code project (.hex file), create a blank project on the MakeCode editor and then drag the code project into the code area. If any changes are made to the project, the project will be needed to be saved and downloaded again. For more information on saving micro:bit projects in MakeCode, visit the micro:bit help and support web page.

Summative assessment

- Please see the 'Assessment rubric' document for this unit.

Subject knowledge

This unit focuses on developing pupils' understanding of variables in a different programming environment and using a physical device. It also enables pupils to combine their knowledge and understanding of programming constructs introduced in previous years. This unit continues to advance pupils' understanding of design in programming, using the approach outlined below.

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

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

Spending time at the 'task' and 'design' levels before engaging in writing code can aid pupils in assessing the 'do-ability' of their programs. It also reduces the cognitive load for pupils during programming.

Pupils will move between the different levels throughout the unit, and this is highlighted within each lesson plan:

- Lesson 3 - pupils work at the 'code' and 'running the code' levels from a given design
- Lesson 4 - pupils move from 'design' to 'code', to 'running the code' with some scaffolding
- Lesson 5 - pupils work at the 'design' level with increasing independence
- Lesson 6 - pupils work at the 'code' and 'running the code' levels, using their own design

Enhance your subject knowledge to teach this unit through the following training opportunities:

Online training courses

- Raspberry Pi Foundation online training courses

Face-to-face courses

- National Centre for Computing Education face-to-face training courses

Resources are updated regularly — please check that you are using the latest version.