**Getting active**

**Lesson 3: Programming step-counters**

**Introduction**

In this lesson, pupils use their previous learning to consider how activity trackers may use variables to record the number of steps a person takes every day. Pupils identify some of the ways that step-counters work and create algorithms to represent them. They then transfer this knowledge to program BBC micro:bits, using the MakeCode editor, to act as a step-counter. Pupils conclude the lesson by evaluating the use of the micro:bit as an activity tracker.

**Time:** @60 minutes

**Learning objectives**

* To identify the uses of a step-counter
* To write an algorithm for a step-counter
* To program the BBC micro:bit as a step-counter

**Materials needed:** lesson presentation, computers/laptops with access to MakeCode editor, printouts of identifying the actions of a program and identifying the actions of a program support worksheets, large sheet of paper, [step counter to sort MakeCode project hex file](https://makecode.microbit.org/#pub:_eVFDrchss0od), and micro:bit devices (optional).

**Lesson summary**

1. Introduction: How many steps? (10 minutes)
2. Analysing step-counters (20 minutes)
3. Programing step-counters (20 minutes)
4. Comparing step-counters (10 minutes)

**Introduction: How many steps? (10 minutes)**

* Use **slide 3** to explain to pupils that in this lesson they will be looking at devices that people use to track physical activities. Invite suggestions on the brand names of these products (Fitbit, Garmin, Galaxy fit-e, etc), what types of data they record and the potential benefits to people’s health (see suggestions in slide notes).

**Analysing step-counters (20 minutes)**

* Explain to pupils they are going to work with a partner to identify all the steps an activity-tracker needs to take in the process of counting steps. Link to pupils’ existing knowledge of **selection** and **variables** and ask them to record in a thinking map all the things that an activity-tracker does when it is counting steps (**slide 4**). See example thinking map on slide 14.
* Display **slide 5** and review pupils’ ideas by asking them to contribute points from their thinking map to a class thinking map which can be created on a large sheet of paper. As pupils offer ideas, probe their understanding by asking them questions to link the ideas to variables i.e. so when does the variable rest to zero, how does the activity tracker know when the daily target has been reached? etc.
* Ask pupils to work with their partner to rank the actions that the activity tracker carries out into an order. Invite pupils to share their ideas then invite pupils to sequence the actions on the class thinking map. The order depends on the actions identified by the pupils but may look like this:

1. sets to 0 steps at midnight

2. records each step a person takes.

3. shows the number of steps that have been taken so far when the screen is tapped.

4. vibrates and shows an image of a trophy when 10,000 steps have been completed.

* Once the actions on the class activity map have been ordered, explain to the pupils that they are going to help you write a class algorithm to show how a step-counter may work.
* Use a large sheet of paper and working sequentially, ask pupils how each action could be represented in the algorithm. Remind pupils of the need to use **repetition** to ensure that the computer is constantly checking to see if steps are being taken. (See example on slide 15).

**Programing step-counters (20 minutes)**

* Give out copies of **identifying the actions of a programworksheet**. Explain to pupils that this represents all the parts of the program to use a micro:bit as a step-counter and ask pairs to identify how a variable has been used in each part of the program.
* Use **slides 7 - 10** to review their ideas, then ask pupils to work with their partner to write a program, using the [**microbit step counter to sort hexfile**](https://makecode.microbit.org/#pub:_eVFDrchss0od), to use the micro:bit as a step-counter (**slide 11**).
* Remind pupils to make use of **identifying the actions of a programworksheet** to support their programing and to test and debug the program as they go along.
* If you have access to physical micro:bits, download the program onto the devices and allow pupils to test out their step-counters.

**Reviewing micro:bit as a step-counter (5 minutes)**

* Display **slide 12** and ask pupils to think/pair/share ways that they might improve the design of the micro:bit as a step-counter to make it more appealing to the ‘fitness tracker’ market. If you wish, use **slide 13** to review the learning objectives.

**Extension ideas**

* You could ask pupils to use a micro:bit as a step-counter for a fixed period of time. Display the steps counted for each user and ask pupils to represent the data in the form of a graph.
* Pupils could build on their ideas developed when reviewing the micro:bit as a step-counter and create a prototype of their design.

**Differentiation**

**Support:** Pupils could use **identifying the actions of a program supportworksheet** which requires pupils to select the action that the micro:bit will carry out for each part of the program

**Stretch & challenge:** Pupils can build the program their micro:bit step tracker program without the use of the support file. They will need to select the blocks that are needed and locate in the menus. They could also explore how the program is created in the text-based, JavaScript editor in MakeCode.

**Opportunities for assessment**

* Informal observation of pupils understanding of variables through class discussions and paired activities.
* More formal assessment of pupils’ explanation of how variables are used in the micro:bit step-counter program if wished.