# Lesson plan - “whisp-o-meter”

## Using the sensor board (MonkMakes Sensor for micro:bit) component

## Introduction

This lesson gives you time to introduce the sensor board component to the learners. The particular focus of the lesson is on loops and specifically loop in range. They will also further understand the usefulness of functions and variables in programs. It also enables the students to understand how a visual display such as the LED lights on the micro:bit communicates more rapidly with a user. In addition they will learn how to program the LED display on the micro:bit in response to input from the microphone sensor on the sensor board.

## Learning objectives

* To successfully set up a micro:bit/electronics kit circuit and flash code to the micro:bit
* To be able to read and interpret a program containing a for loop
* To be able to use the range command
* To be able to read and interpret a program containing functions
* To be able to successfully modify a program containing a for loop

## Keywords

For, loop, range, variable, function, in-built, user-defined, subroutine

## Preparation

**Subject knowledge:**

This lesson is suitable for a class who already have a basic knowledge of programming using python. The coding for the micro:bit is written in a reduced version\* of python called micropython but for practical purposes the syntax will appear the same to the user. In addition the students should have already used the micro:bit, practising simple set up and smaller micro:bit specific programs so that they are used to building and flashing programs. Initially, building the circuit for the sensor board is a question of copying a diagram so a practical demonstration of handling the components gently and clipping with alligator clips would suffice.

**Pedagogical approach:**

The lesson is planned using the PRIMM pedagogy which stands for:

P - Predict

R - Run

I - Investigate

M - Modify

M - Make

For this reason it is important that the class have quick and easy access to the coding of the program which can be accessed via: <https://tinyurl.com/y59jz2fk>. (The name of the program is actually shoutometer but for the purposes of classroom management it is named “whisp-o-meter” in the resources).

The very first step is for the students to view the code and attempt to predict what the code does. We use a tool called a **code conversation** which provides the teacher with a conversation style **talkthrough** of the micropython code for you to support your students’ developing knowledge.

**Practical set-up and development environment:**

We recommend either using the on-line Python editor (https://python.microbit.org) or Mu (https://codewith.mu). Or you could manage your coding on micro:bit’s own environment <https://classroom.microbit.org/> which enables you to push code to the whole class and manage their activities.

**Resources that you need:**

* Presentation
* Starter worksheet questions
* Starter worksheet answers
* Code conversation solution
* Code conversation template (level 1 - basic)
* Code conversation template (level 2 - intermediate)
* Code conversation template (level 3 - difficult)
* Plenary worksheet
* Plenary worksheet answers
* Hardware Per pair - 1 micro:bit, 1 USB/micro USB cable, 3 alligator clip leads, 1 sensor board.
* whisp-o-meter program in micropython accessible to students
* Access to appropriate development environment (the on-line Python editor, Mu or micro:bit classroom)
* Micropython web page. This is used if the students want to modify the program using images. https://microbit-micropython.readthedocs.io/en/v1.0.1/tutorials/images.html

## How the students’ progress is assessed

The presentation itself begins with a for range question followed by a worksheet testing application for the for range syntax. It is interspersed with pair or whole class discussion opportunities. As this is predominantly a practical lesson observing the students’ success in flashing the program, compiling the circuit and making small successful modifications to the program will also enable you to assess their progress. The learners also complete a worksheet at the end of the lesson which demonstrates their understanding of key principles conveyed during the lesson.

## Plan (with approximate timings)

|  |  |
| --- | --- |
| Starter activity 10 mins | **Range revision**  The first slide in the slide deck presents the students with a simple range question. They are then encouraged to discuss their solution with their neighbour and articulate what happens within a for y in range () loop. They are also encouraged to use the correct terminology to build their confidence. The slide deck gives some further explanation about the simple applications of for and range. This is followed with a 6 question starter worksheet, the answers to which should be shared as a class. |
| Activity 1 5 mins | **Predict: students view code and attempt overview code conversation**  The students are shown the code and are asked to predict what the whole program does i.e. an overview in a couple of sentences.  For example:  If you make a noise near the microphone section of the MonkMakes Sensor Board the LEDs on the micro:bit light to show the volume. The louder the noise, the more LEDs will be lit. The program defines a function called bargraph which firstly clears the LED display then produces a bar graph representing the volume of received sound waves on the LED display. |
| **Activity 2**  10 mins | **Run: students compile the circuit, flash and run the code**  Share the slide with instructions on the board to enable the students to build the circuit, flash and run the code. Ask them if the circuit behaves as they would have expected from the **predict** phase of the lesson. Encourage the students to firstly whisper near the microphone sensor then clap loudly to fill the LED display. |
| Activity 3 20 mins | **Investigate: teacher and students use correct terminology to identify the syntax and features of the program**  Using the A4 worksheet, the learners attempt to complete the code conversation matching the correct explanation to each line of code.  There are three levels of the code conversation:  Level 1 - basic (some less challenging code lines to complete), Level 2 - intermediate (more challenging code lines to completed), Level 3 (whole program to interpret).  Using the slides talk through the different steps in the program. In this program, it is particularly important to have another tab open with the full program available to view. The program contains a nested loop and passes parameters so while we can work through individual lines of code they only really make sense as a whole picture.  This section of the slide deck also provides explanation of the layout of the micro:bit LED display. |
| Activity 4 10 mins | **Modify: students modify the code to create a new program**  Students are invited to modify the program by first declaring what their new program should do, then by making the modification, saving the new program and testing it by flashing it to the micro:bit.  Suggestions for achievable alterations:-  Changing the brightness any of the LEDs  Find out what happens if you don’t use display.clear()  Change the name of any of the variables or parameter  Produce a heart picture on the LED display for a quiet noise and a surprised emoji for a loud noise.  If you have any students who want to explore the use of images on the micro:bit in response to the microphone input this page provides a list of images: https://microbit-micropython.readthedocs.io/en/v1.0.1/tutorials/images.html |
| **Plenary**  5 mins | **Whisp-o-meter plenary worksheet**  Students complete the plenary worksheet testing knowledge of components and micropython. |
| **Homework** | **Consider what other modifications you could make to the program, still using the sensor board component** |

## The Author

This lesson plan and all its parts were created by Dr. Paula Beer of Beer Academic Consultancy in collaboration with Monk Makes Ltd.



Dr Paula Beer has taught Computer Science and IT education to new and established teachers since 2007. Her own research has focused on the use of play and collaboration in computer science. She enjoys supporting practicing teachers by designing accessible lesson planning materials to get students engaged in computer science through play and collaboration. Paula has also produced educational materials for The Raspberry Pi Foundation, been a secondary school teacher, written a successful book (Hello App Inventor!) and has previously worked in IT project management for a media company and for the NHS.

