Coding Fundamentals

Micro:bit Python Programming
Buttons and Conditionals

Overview
In this lesson, students learn about conditional statements using the micro: bit buttons using Python.

Objectives
- Integrate and use the Micro: bit's buttons
- Use conditional statements and while loops
- Program actions to correspond with pressing buttons

Materials
- micro:bit and micro-USB cord
- Computer with access to the internet

Approx. Time Required
1-2 hours

Cyber Connections
- **Programming** – Students will program in Python.
- **Hardware and Software** – Students will utilize small electronics and learn how a computer is programmed while using microcontrollers.

CYBER.ORG
THE ACADEMIC INITIATIVE OF THE CYBER INNOVATION CENTER

This content is based upon work supported by the US Department of Homeland Security's Cybersecurity & Infrastructure Security Agency under the Cybersecurity Education Training and Assistance Program (CETAP).

Teacher Notes:

Buttons and Conditionals

Remind students of the line that should appear at the top of every program.

```
from microbit import*
```

In this lesson, students will learn to use two of the most powerful tools in programming—conditional statements and loops—while familiarizing themselves with the buttons on the micro:bit.

Conditional Statements

Conditional statements are prevalent in every programming language because they have a multitude of uses. The idea behind them is that IF a condition is true, THEN another action or block of code will trigger and execute. In Python, conditional statements take the following format (elif stands for else if):

```
if something is True:
    # DO something
elif some other thing is True:
    # DO something else
else:
    # DO the final option
```

The worksheet for this lesson has examples for students to grasp this computational thinking concept, as well as space for them to write their own "everyday" conditional statements.

Loops

The second tool, loops, goes hand-in-hand with the computational thinking style that is required for conditional statements. There are two main types of loops: while loops and for loops. **For loops** execute a specific block of code a specific number of times. This will be covered in a later lesson. **While loops**, which are covered in this lesson, loop a specific block of code WHILE a certain condition remains true. Python uses the format below for while loops.

```
while something is True:
    # DO this code
```

The worksheet also has examples and spaces for while loops that match the style of the conditional statements above.

The main loop for this lesson is an indefinite loop. This can be set up easily with the following line:

```
while True:
```

This block of code will repeat itself while true is true. This tool can be used any time you want a section of a program to loop or stay activated until a certain event is triggered. In this case, we want a program that responds to the button presses to stay active as long as power is connected to the Micro:bit, so we use `while True:`

Buttons

The final command required for this lesson is tied to using the buttons. Each button (`button_a` and `button_b`) is its own object and therefore has its own commands programmed to work with it. In this case, the command is `button_a.is_pressed()`. This command will return a value of "true" if the button is being pressed, which works well with the conditional statements from earlier in the lesson. Once they have this line, students can program a button press to do anything they already know how to do! For example:

```
If button_a.is_pressed():
   display.show(boat)
elif button_b.is_pressed():
   display.clear()
```

This code displays the boat image created in a previous lesson *IF* the A button is being pressed, and clears the display *IF* the B button is being pressed. Allow students to play around with this structure of code (not the same exact code unless the boat is programmed in). What do they notice?

The code provided above runs only once upon start up, and this is where the `while True:` loop comes in to play. For this code to run as long as the micro:bit has power, only one line needs to be added and indents need to be in order. An example of the same code running forever and displaying a Christmas tree is provided below:

```
from microbit import *

while True:
    if button_a.is_pressed():
         display.show(Image.XMAS)
    elif button_b.is_pressed():
         display.clear()
```

Encourage advanced students to code their array of images to cycle forwards or backwards one image by pressing either button instead of using `scroll` command.