Unit 1: Getting Started with the TI-Innovator™ Hub

Application: The Traffic Signals

In this application you will write a program to simulate a traffic signal: both lights and sounds (for the blind).

Objectives:

- Control the timing of the red, yellow, and green states of a traffic signal
- Combine light, color, and sound into one project
- Use a **while** loop to repeat a procedure until esc is pressed

A traffic light has three separate bulbs in red, yellow, and green so that color-blind people can tell the state of the light just by the position of the lit bulb.

On the side of the road there is also an audio signal to let blind pedestrians know when it is safe to cross the street. You will make the sound as part of your traffic signal system.

1. Write a program to input the number of seconds the traffic light is red, yellow, and green. When the traffic light is red, both the color LED and the red LED should be on. When the traffic light is yellow or green, only the color LED should be on. When the traffic light is green, there should also be an audio signal indicating that it is safe to go.

   Start with a new Python Hub Project. Ours is called u1app.

2. Add three **input()** statements for the times (in seconds) that the light is red, yellow, and green. For simplicity, we use only whole numbers so remember to use the **int()** function around the **input()** function to convert the entered string into a number.

   The screenshot only shows one *sample* statement.

   Note the capital 'O' in **redOn**. Python is case-sensitive, so all references to this variable must be written the exact same way.

3. Start with the red light. Turn both the **color** LED red *and* turn the **light** (red LED) on, too.

   Use the **sleep()** function to pause the program for the **redOn** number of seconds using the statement:

   sleep(redOn)

   Be sure to turn the **light** off when the **color** changes to yellow.

4. Turn the color LED yellow for the yellow signal and then green for the green signal.

   While the green light is lit, also make a sound (use either .**tone** or .**note**) using the speaker. The sound should stay on as long as the green light is lit and then turn off.

   Run your program to test it. Watch the lights on the TI-Innovator Hub.

5. If your program worked properly, it ended after one cycle through the three colors. To allow the program to cycle though the three colors repeatedly, you will add a *loop* to your code.

6. A **loop** is a programming 'control structure' that processes a *block* of code over and over. One type of Python loop is the **while** <*condition*>**:** loop. The **<***condition***>** is a Boolean expression, one that evaluates to either **True** or **False**.

   An example: **while x<10:** (the colon (:) is required)
   block (*block* is intentionally indented)

   As long as **x** is less than 10 the statements in the block will be executed repeatedly. When **x** is greater than or equal to 10, the loop will end, and processing passes to the first statement below the *block*.

7. Insert a blank line in your program below the three input statements. Then get the statement

   while get_key() != ”esc”:

   from **menu > TI Hub > Commands**.

   Notice that the light gray word ‘*block’* is *indented* two spaces and that these spaces have light gray diamond placeholders (they really are just spaces).

   != is the Python symbol for ‘does not equal’.

8. The **while** loop executes all the statements in its ‘*block’* over and over until the **esc** key is pressed. The block is defined by the indent (two spaces).

   Delete the word ‘*block’* (an ‘inline prompt’) and then indent the rest of your program two spaces. You could type two spaces at the beginning of each line, press **tab** at the beginning of each line, or take this shortcut:

   S*elect all* the statements below the **while** statement (using **shift+down arrow**) and *then* press the **tab** key. This indents each selected line.

9. Check to make sure that all your color, light, and sound statements are indented the same number of spaces (two). This makes all your light code become the **while** *block*. Caution: Improper indenting can lead to errors.

   Run your program again. Press **esc** when you are ready to stop the program. There may be a delay in stopping the program because it must go through all the block code and stops the loop only after the green light finishes.

   When the program ends, what is the state of the LEDs on the TI-Innovator Hub? How would you make sure that they are both turned off?