Unit 1: Getting Started with the TI-Innovator™ Hub | **Application: Traffic Light**

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

Objectives:

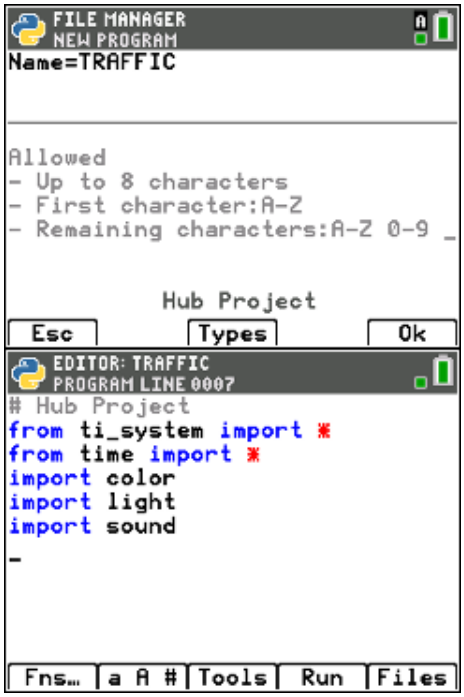
- Control the timing of the red, yellow, and green states of a traffic signal
- Combine light, color, and sound into one project
- Use a **while** loop to repeat a procedure until escape is pressed

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 sides 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 numbers 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.

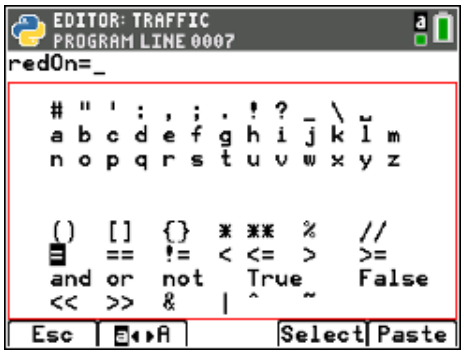
   Start with a **new Python Hub Project**. Ours is called **TRAFFIC**.



2. Since we know that we will be using **light**, **color** and **sound**, add all three **import** statements to your code using the **[math]** menu.
   The order of these three statements is not important.

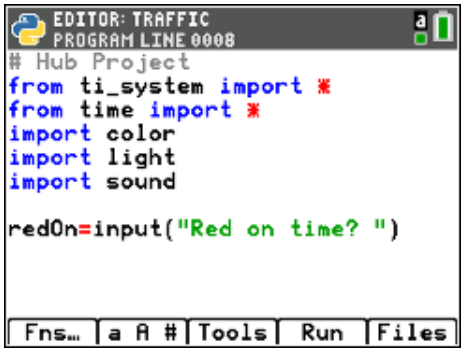
3. Write three input statements that accept and store the times that each color light will be on. For long text writing (to use more meaningful variable names) try using the **<a A #>** Character Map. In this editor lowercase alpha-lock is automatically on and you can simply type your letters. For a capital letter select the **< a◄►A>** toggle. Some special symbols are also on this screen so use the arrow keys and either **<Select>** or **[enter]** to add them to your line at the top of the screen.

   When you are done, select **<Paste>** to paste the text into the Python Editor. You can write partial lines of code and complete them in the Editor.



4. After typing redOn= in the Character Map, select **<Paste>** to return to the Editor. Then get the input() function from **<Fns…> I/O**.
   Turn on alpha-lock ([2nd] [alpha] and type the prompt inside the parentheses. Recall that the quotation mark (**"**) is on the **[+]** key.

   Note: you can sue shorter variable names to reduce the typing. Meaningful variable names help make your code easier to read.

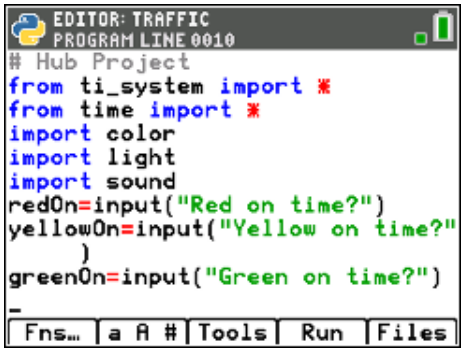


5. Repeat for the other two colors of a traffic light.

   Or use the **<Tools> Copy line** and **<Tools> Paste line below** then edit the new statement.

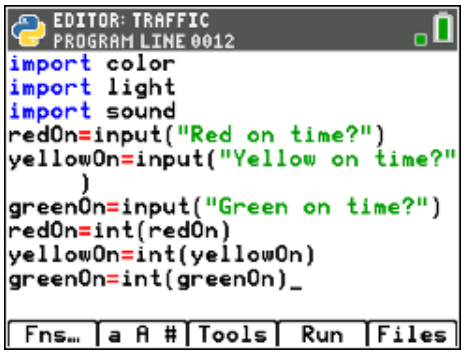
   Note that we are using a capital 'O' in the variable names. This capital letter must be used whenever the code refers to one of these three variables. The capital letter makes reading the variable name easier.

   Does your code look like this this so far?



6. Convert each string to an **int**eger (or a **float** for decimals).
   You can combine the **int( )** function with the **input( )** function to reduce typing.

   Example: **r = int(input("Red on time?"))**
   (not shown and using the variable name **r** *instead of* **redOn***) Be careful about the positions of the parentheses.*



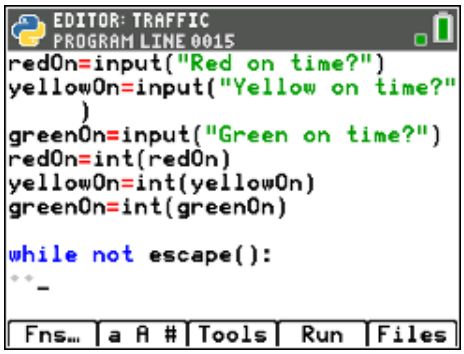
7. Since we want our traffic light to operate continuously we introduce a special **while** loop:

   while not escape():

   ♦ ♦

   found on both **[math] ti_system…** and **[math] ti_hub… Commands**

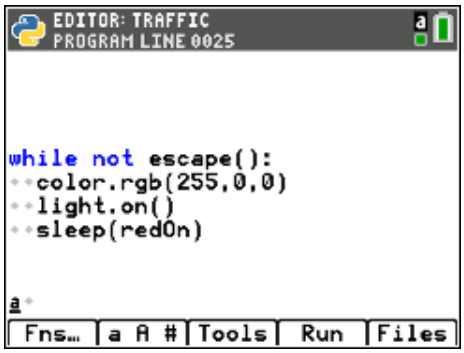
   A programming loop is a section (block) of code that is executed repeatedly until some condition occurs.



   This loop executes the (indented) **block** of code as long as the 'escape' key is not pressed. Two indentation spaces are provided in the beginning of the next line and all subsequent lines in the block must also be indented this same number of spaces. The Editor provides these spaces for you until you delete them using the **[del]** key.

   Note that there is no 'escape' key on your keypad: use the **[clear]** *key as an 'escape' key to exit the loop*.

8. In the block (the indented section of code), write the code to control the lights, colors, and sounds. Use **sleep( )** functions (found on **[math] time…**) to control the timing of the lights and synchronize the sound with the green light.
   - When the light is red we want both the red LED (light) and the color LED (in red) to be on.
   - Use the redOn variable in sleep( ) function to keep the lights on for the right amount of time.
   - For the green and yellow lights, turn the red LED off (**light.off()**) and just use the color LED.
   - When the light is **green** make a beeping sound (use the 'tempo' option in the .tone( ) or .note( ) function.



9. All of the statements above are inside the **while not escape()** loop (indented two spaces) and are not displayed in this image.

10. When you press the **[clear]** key to end the loop, turn all the lights and sound off. *Dedent* (unindent) your cursor to the left side of the screen using the **[del]** key or use **<Tools> Dedent**.
    Write the statements to turn both lights off.

    Do you need to turn the sound off? No. It will stop by itself. How could you turn the sound off at this point?
    Note: the code for the while block is missing. The two **.off( )** *functions are not part of that loop since they are not indented.*