Bright Lights

This Unit contains a total of 5 activities that all use the TI-Innovator Hub and its built-in devices. Each activity builds on the coding skills learned in the previous activity.

Objectives:

Programming Objectives:

- Use built-in functions from the TI-Hub Library.
- Use RGB colors to change the LED on the TI-Hub.
- Use the brightness sensor to measure brightness.
- Use the random library to generate integers.
- Use for statements to repeat code.
- Use a while statement to repeat code.
- Use lists to store values.

Key AP Computer Science Principles Standards:

- Represent a value with a variable (AAP-1.A)
- Represent a list or string using a variable (AAP-1.C)
- Write conditional statements (AAP-2.H)
- Write nested conditionals (AAP-2.I)
- Select appropriate libraries or existing code segments to use in creating new programs (AAP-3 D)
- Write iteration statements (AAP-2.K)
- Write expressions that use list indexing and list procedures. (AAP-2.N)
- Write iteration statements to traverse a list. (AAP-2.O)
- Write statements to call procedures (AAP-3.A)
- Develop procedural abstractions to manage complexity in a program by writing procedures. (AAP-3 C)
- For generating random values, write expressions to generate possible values. (AAP-3 E)

This document contains Activities 3 and 4 of the Bright lights activities.

Activity 3: Changing colors

Students use a for loop to repeat either the “Color my world using RGB” activity or the “Randomizing color” activity a set number of times.

Changing colors take 2

Students use a while loop to determine when to stop repeating code.

Activity 4: Lights and Color

Students learn how use the brightness sensor to measure the brightness.

Students will use if statements and the brightness value to create colors.

©2022 Texas Instruments Incorporated

Activity 3: Changing Colors

Use a for loop to repeat either the “Color my world using RGB” activity or the “Randomizing color” activity a set number of times.

Changing Colors 2
Use a while loop to determine when to stop repeating code.

1. Create a new program named “changeColor”.

2. You will need both the TI-hub library and the random library.
   Import both libraries.

3. This project will use a for loop to repeat code.

   Ask the user for the number of times to change the LED color.
   This value should be stored as an integer in a variable n.

4. The basic structure of a for loop is:
   for index in range(num):

   The index should be a variable name.
   num needs to be an integer value.

   Go to Menu> Built-Ins > Control > for index in range(num)

   Add the loop.
   Put a variable i in for the index. Add your variable n as the num.
   Put print(i) in the loop.

   Notice the print(i) must be indented two spaces.
   The indentation formatting will be important in this project.

5. Execute your program. Enter 4 for the number of changes.

   Question 1: What values are printed to the screen?

   Question 2: What is the smallest value for i?

   Question 3: What is the largest value for i?

   Question 4: How many times does the loop execute?

6. The number in the range() determines the number of times the loop executes.

   Add three lines of code. Ask the user for integer values for red, blue and green. Store these values in variables r,g,b.

7. Add two lines.
   One that sets the LED to the color specified by the user.
   One to pause/sleep for 5 seconds before asking for another set of rgb values.

   Make sure all 5 lines of code are tabbed in two spaces (diamonds).
   If your tabbing doesn't line up, the code will throw an error.

8. Run your code. Verify it works.

9. Insert a new page (ctrl i).

   Create a program named "changeColorRand"

10. Go back to the original code on page 1.1.
    Copy the code, then paste it on page 1.3.

11. Deleted the input code.

12. Use what you learned in "Activity 2 Changing Colors" to generate random integers for r,g, and b. Either use the int() and random() function or the randint() function to create integers from 0 to 255.

13. Run your program several times.

If you enter 3 for n does your program produce 3 random colors that each last for 5 seconds?

If you enter 6 for n does your program produce 6 random colors that each last for 5 seconds?

14. Add one more python page. Name this new document "changeColorRnd2"

15. Copy your code from “changeColorRnd” and paste it into this new document.

16. Wouldn’t it be nice if your code would generate colors until you tell it to stop?

Instead of predetermining the number of times through the loop, let it continue until the user presses a button?

There is a built-in function for that.
Delete the lines:

n = int(input(“number of LED changes”))
for i in range(n):

17. Go to the ti_hub menu.
In the commands, find: while get_key() != “esc”:

This loop executes continuously.
When the user presses the “esc” key the loop will exit.

The light will stay on.
If you want to turn the light off, add the line:

color.rgb(0,0,0)

after the loop.

Notice it shouldn’t be indented in the code.

Tech Tip: To exit an infinite loop
TI-Nspire Software on Windows© : press F12 and Enter Repeatedly
TI-Nspire Software on a Mac ©: pres F5 then Enter repeatedly
Handheld: Hold the ON/Enter button for a few seconds.

18. Execute your program. Does pressing the “esc” key exit the loop?

Acitivity 4: Lights and Color

Learn how to use the brightness sensor to measure brightness.

Use if statements and the brightness value to create colors.

1. In activity 1, you learned how to import the ti-hub library and use the function color to change the color of the LED.

   Create a new program named lightsColor.
   Set the Type to “Blank Program”.

   Import the ti_hub library.

2. In this activity, you will use measurements from the built-in brightness sensor to change the colors on the LED. You may recall from the first activity the brightness sensor is located on the side of the hub.

   Brightness:

   measurement()
   Measures the brightness from 0-1.
   range(min,max)
   Allows you to rescale the values reported for brightness.

   In activity 1, you used the code below to collect a brightness measurement and print it to the screen.

```
m = brightness.measurement()
print(m)
```

3. For this activity, you will continually take a brightness measurement every 0.5 seconds and print it to the screen until the user presses the “esc”

   Code the four lines of code to make this happen.

   Run your code, verify it works.

4. Does your code match the code to the right?

```
1.1 1.2 *Doc RAD
lightsColor.py 7/7
from ti_hub import *

while get_key() != "esc":
  m=brightness.measurement()
  print(m)
  sleep(0.5)
```

5. Execute your code. You need to determine three brightness ranges: low, medium, bright. To create these ranges you might consider covering the sensor or shining a light source at the sensor.

   Record reading ranges for the classroom.

Low	
Medium	
High	

6. Now to add some color variety. If the brightness value is low, display a blue light. If the brightness value is medium, display a yellow light. For a high brightness value, display a red light.

   To make this happen. You'll need an if..elif..else statement. This can be found using Menu > Built-ins > Control > if..elif..else

   Add this statement after you print the value but before the sleep statement.

```
1.1 1.2 *Doc RAD
*lightsColor.py 12/14
while get_key() != "esc":
  m=brightness.measurement()
  print(m)
  if BooleanExpr:
    block
  elif BooleanExpr:
    block
  else:
    block

  sleep(0.5)
```

7. Use your measurements from step 5 to display:
   blue light- low measurements
   yellow light- medium measurements
   red light- high measurements

```
1.1 1.2 *Doc RAD
*lightsColor.py 11/14
while get_key() != "esc":
  m=brightness.measurement()
  print(m)
  if m<___:
    color.rgb(___,___,___)
  elif m>___:
    color.rgb(___,___,___)
  else:
    color.rgb(___,___,___)

  sleep(0.5)
```

Replace ___ with your values

8. Recall, the hub can play sounds.

Sound:

tone(frequency,time)
Play a tone based on the frequency and time.
Frequency 0-8000 hz. Time 0.1-100 seconds

note(“note”, time)
Play a tone based on a note.
A1 is an A in the first octave. A4 is A in the 4th octave.

tone(frequency,time,tempo)
Same as the original tone function, except you can set the tempo using integers from 1-10.

note(“note”, time,tempo)
Same as the original note function, except you can set the tempo using integers from 1-10.

9. Add a distinct tone or note for each selection.

10. Application: You may have run across several different types of lights that use sensors to control their behavior.

Did you know some night lights use a sensor to turn on the light if the surrounding area is too dark?

Some lights have built-in “motion detectors” that turn lights on if motion is detected.

Can you list at least 5 places you have seen lights that use a sensor to turn the lights on or off? Are any of these light systems tied to a sound system?