

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 1 and 2 of the 5 total Bright Lights activities.

Activity 1: TI-Hub Built-In Devices

Students use functions in the TI-Hub library to change the colors on the hub's LED.

Activity 2: Randomizing color

Students use the randint function from the Random library to select valid RGB colors.

Students use the random and the integer function to select valid RGB colors.

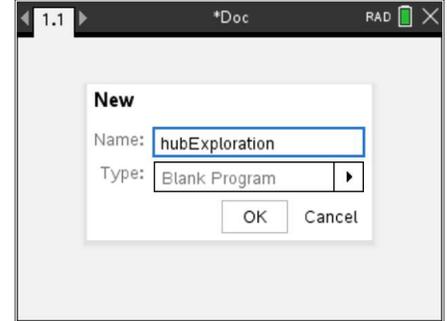
Activity 1: TI-Hub Built-In Devices

Students use functions in the TI-Hub library to change the colors on the hub's LED.

1. Create a new program named "hubExploration".

Choose Blank Program for the default type.

*You can name your project "hubexploration" all in lower case. However, using capital letters for the E makes it easier to read the name of the file. Often, programmers capitalize the first letter of each new word in a variable name. This form of naming is referred to as "camel case". It doesn't change how the program runs; it does however make it easier for the programmer to read.



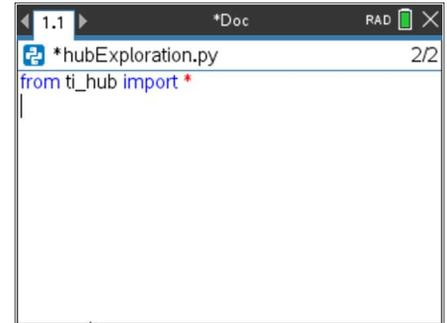
2. In order to use the functions that communicates with the TI-Innovator Hub, you need to import the TI-Hub library.

To import this library, you need the line: `from ti_hub import *`

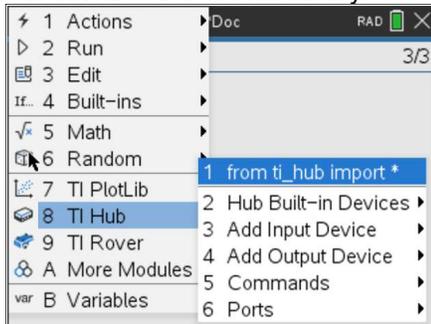
Don't type this entire line of code.

You can use the tools in the menu to select and enter this line.

Use the menu items to add this line of code.



3. Let's explore the functions in the TI-Hub library.



4. Fill in the blanks below.

The TI-Hub Built-In Devices has four main options.

5. The Built-In Devices are just that, built-in to the hub.

LED:



The library contains several useful functions:

`color(red, green, blue)`

Creates any color you can imagine.

Values for red, green and blue can be any integer from 0-255.

`blink(frequency, time)`

Frequency can be anything between 0.1 and 20 Hz.

Time can be anything between 0.1 and 100 seconds.

`color.off()`

`light.on()`

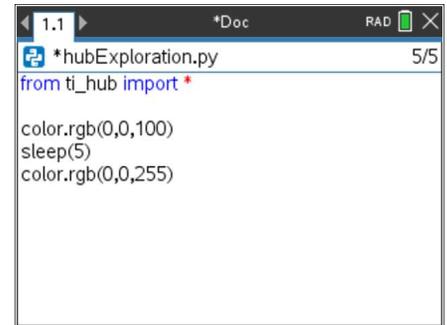
`light.off()`

6. Add the following lines of code to your project.

```
color.rgb(0,0,100)
sleep(5)
color.rgb(0,0,255)
```

*Sleep can be found using Menu > TI-hub > Built-Ins > Command

Execute your code (ctrl r)



Question 1: What color is the initial color of the LED?

Question 2: After 5 seconds, the LED changes. How does the LED change? Why does this happen?

Teacher Tip:

Question 1: The LED is a pale blue.

Question 2: The LED is still blue, however it is brighter/bolder.

7. Change the RGB (red, green, blue) values. What colors can you make?

Fill in the values below:

Pink/magenta `color(____,____,____)`

Yellow `color(____,____,____)`

Green `color(____,____,____)`

Orange `color(____,____,____)`

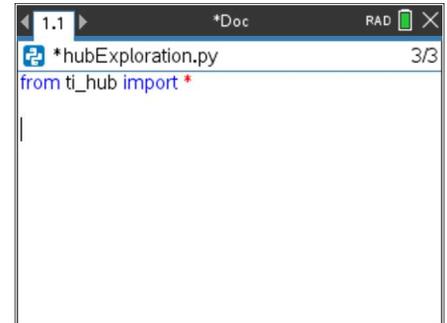
White `color(____,____,____)`

Red `color(____,____,____)`

Teal `color(____,____,____)`

Your favorite color `color(____,____,____)`

8. Remove all your code except for the library import.



9. Use the light (not the color) functions to turn the light on, wait 10 seconds, then turn the light off. Hint: You'll need the sleep function.

Write the code you used below.

Teacher Tip: `from ti_hub import *`
 `light.on()`
 `sleep(5)`
 `light.off()`

10. The sound functions are as follows:

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.

11. Set the tone to 264hz for 2 seconds. This value will play a "C".

Sleep for 5 seconds.

Change the tone to 400hz for 2 seconds.

Copy your three lines of code below:

Question: Does the second tone play “higher” or “lower” than the first?

```
Teacher Tip:      from ti_hub import *
                  sound.tone(264,2)
                  sleep(5)
                  sound.tone(400,2)
```

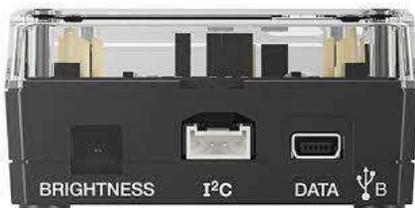
- 12. Remove the sleep(5) from your code.
Play the 264hz for 2 seconds, the 400hz for 2 seconds.

What happens when you run your code now?

Teacher Tip: Both tones play at the same time. The calculator processes the code faster than it plays the sound. Therefore, students need a sleep command in between the two in order to “pause” before the next tone is executed.

- 13. The brightness sensor is located on the side of the hub.

Brightness:

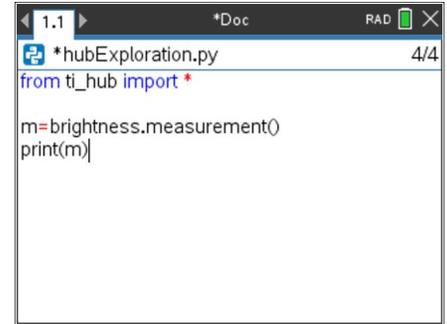


measurement()
Measures the brightness from 0-1.
range(min,max)
Allows you to rescale the values reported for brightness.

14. Use the brightness sensor to take and store a measurement into a variable m. Display the measurement to the screen.

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

Run the code.



15. Question 1:
When the code runs, it takes one measurement, what is the initial measurement?

Question 2:
Cover the brightness sensor with your hand. Re-run the code.
What value is this new measurement?

Question 3:
Find a bright spot in the room. You might use a flashlight or a flashlight app to shine light on the sensor. With the additional light, what is the measurement of the sensor?

Teacher Tip:

- Question 1: Answers will vary
- Question 2: Answers will vary but it should be less than the initial measurement.
- Question 3: Answers will vary but it should be the highest value the student records.

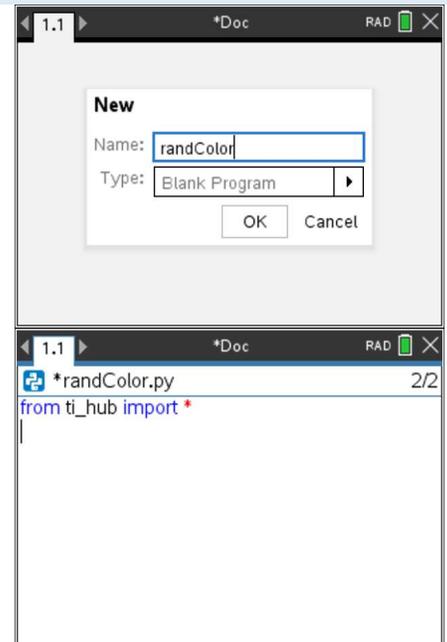
Activity 2: Random Color

Use the randint function from the Random library to select valid RGB colors.
Use the random and the integer function to select valid RGB 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.

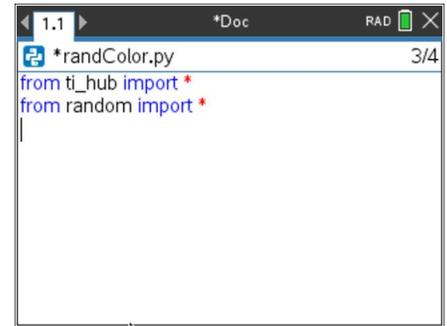
Create a new program named randColor.
Set the Type to “Blank Program”.

Import the ti_hub library.



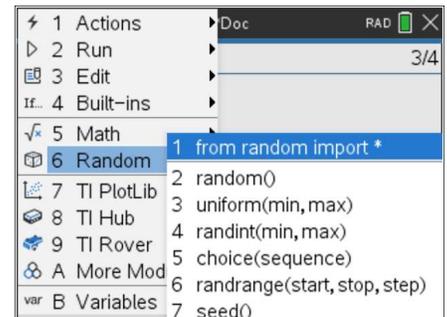
2. This project will use functions in the random library.

Press the menu button.
 Find the random library.
 Select “from random import *” from this menu.



```
*randColor.py 3/4
from ti_hub import *
from random import *
```

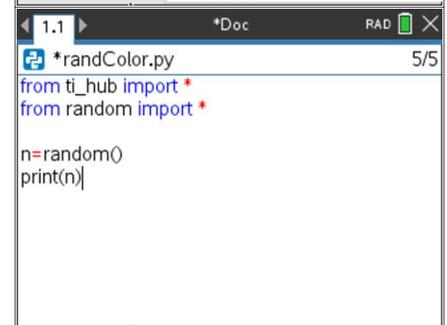
3. The random library has many useful functions. This activity will use two different functions, random() and randint(min,max).



```
1 Actions
2 Run
3 Edit
4 Built-ins
5 Math
6 Random
7 TI PlotLib
8 TI Hub
9 TI Rover
A More Mod
var B Variables
```

```
1 from random import *
2 random()
3 uniform(min,max)
4 randint(min,max)
5 choice(sequence)
6 randrange(start, stop, step)
7 seed()
```

4. Add the lines:
 n = random()
 print(n)



```
*randColor.py 5/5
from ti_hub import *
from random import *

n=random()
print(n)
```

5. Run your program 12 different times.
 Record the 12 different values for n printed to the screen.

_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____

6. If possible, compare with another student. How do your answers compare to their answers?

7. What do you think the largest value for n could be?
 What do you think the smallest values for n could be?

Teacher Notes:

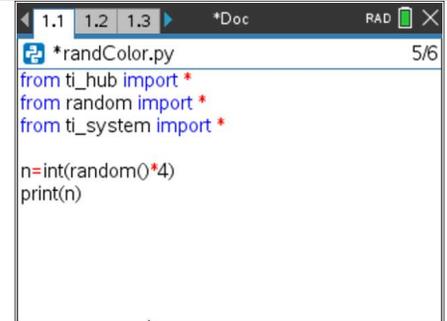
Student answers may vary. They could say values range from [0, 1].

random() returns a floating point number from 0 to 1.0

8. The random() function generates floating point numbers from 0 to 1, excluding 1.

Modify your code to be:

```
n = int(random()*4)
print(n)
```



```
*randColor.py 5/6
from ti_hub import *
from random import *
from ti_system import *

n=int(random()*4)
print(n)
```

Run the code 12 times. What values do you get for n?

_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____

Teacher Tip:

All values should be between 0 and 3.

9. Modify your code so that n is any integer between 0 and 8 including 0 and 8.

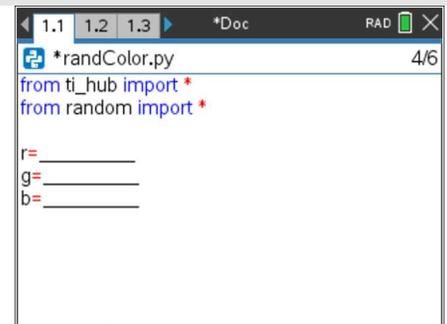
Execute your code several times to ensure your formula works.

Teacher Tip:

```
n = int(random()*9)
```

10. Now to apply what you have learned to generate random colors.

Create variables r, g and b. Use the int() and random() functions to generate a random integer for each variable between 0 and 255.

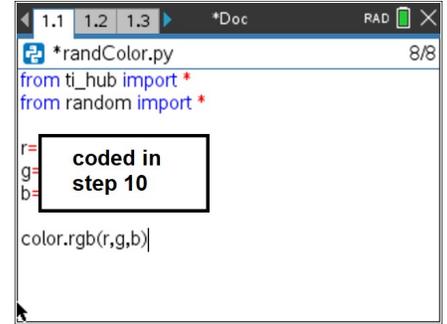


```
*randColor.py 4/6
from ti_hub import *
from random import *

r= _____
g= _____
b= _____
```

11. Add the line of code to change the color of the Ti-Hub's LED.

Run your code several times. Is the light a random color each time?



```
*randColor.py 8/8
from ti_hub import *
from random import *

r=
g=
b=
color.rgb(r,g,b)
```

Teacher Notes:

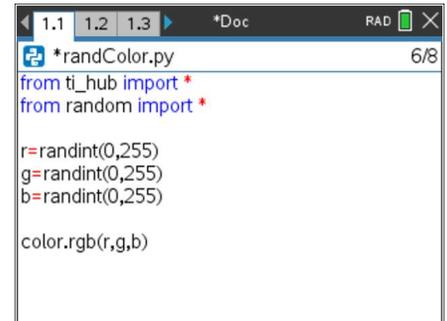
```
r=int(random()*256)
g=int(random()*256)
b=int(random()*256)
```

12. The random library contains a function named randint(max,min).

If you type the line “n=randint(1,6)”, n will be a random integer from 1 to 6 including both 1 and 6. This one function replaces the int() around the random function.

13. Replace your code from set 11. Use the randint function instead of the random function.

Run your code several times. Verify this code also generates a random color each time.



```
*randColor.py 6/8
from ti_hub import *
from random import *

r=randint(0,255)
g=randint(0,255)
b=randint(0,255)

color.rgb(r,g,b)
```

Teacher Notes:

After completing Activity 1 and 2 some student extensions could be:

- 1.) Write a program that randomly generates a frequency between 400 and 800. Play the frequency for 5 seconds.
- 2.) Write a program that randomly generates and displays a color for 2 seconds. Randomly generate another color and display it for 4 seconds. Generate a third random color and display it for 2 seconds. Finally, turn off the light. Hint: You will need to use sleep() statements in between each color.