10 Minutes of Code: Python Modules

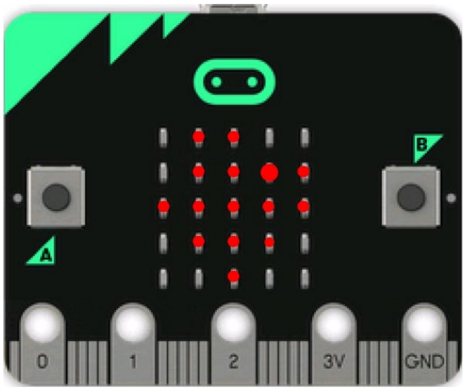
TI-NSPIRE™ CX II

MICRO:BIT: THE LIGHT SENSOR

micro:bit — The Light Sensor

In this activity, you will monitor the micro:bit light sensor and store the data in a TI-Nspire list for further analysis.

1. The micro:bit can read the ambient light level using the display LEDs. Yes, the LEDs can also be used as an input device!

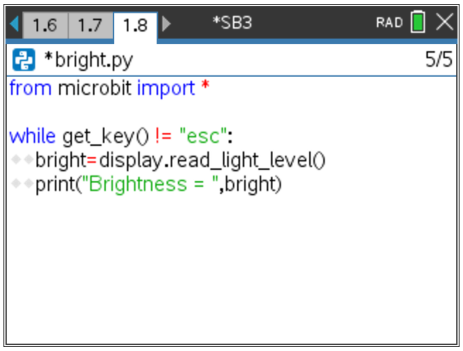


2. Start a new blank python program. We named the program **bright.**

   To check the values that the light sensor can detect, write the following short program using the BBC micro:bit menus:

```
from microbit import *

while get_key() != "esc":
♦♦bright = display.read_light_level()
♦♦print("brightness = ",bright)
```



```
from microbit import *

while get_key() != "esc":
••bright=display.read_light_level()
••print("Brightness = ",bright)
```

   Find **var= read_light_level()** on **[menu] > BBC micro:bit >Sensors**

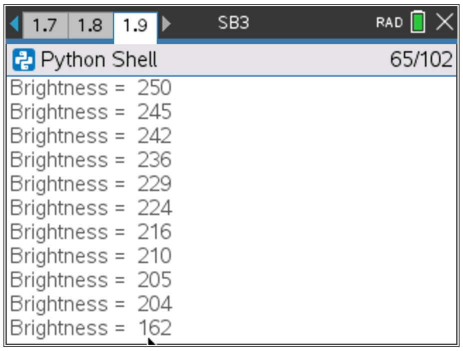
   bright will be the variable name that you type.

   Reminder: **[menu] > More Modules > BBC micro:bit > Commands> while get_key() != "esc":**

   Remember to *indent* the last two statements so that they are the **while** loop body.

3. Run the program and point the display of the micro:bit at a light source. It does not matter what is showing on the display. Move the micro:bit toward and away from the light and observe the changing values on the TI-Nspire CX II screen. You should see values varying between 0 and 255.

   As you probably expect, the further from the light source, the lower the light level value. Now you will add code to the program to collect the light level data to create a scatter plot of light vs. time.



```
Brightness =  250
Brightness =  245
Brightness =  242
Brightness =  236
Brightness =  229
Brightness =  224
Brightness =  216
Brightness =  210
Brightness =  205
Brightness =  204
Brightness =  162
```

   Press **[esc]** to end the program and go back to the Editor.

4. Create <u>two</u> empty lists before your **while** loop:

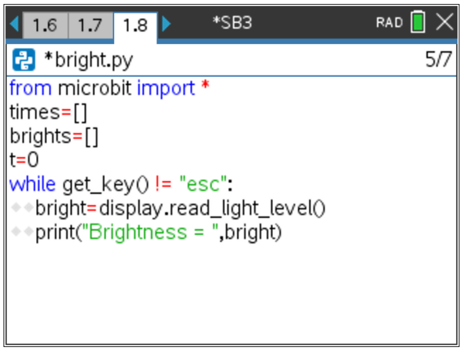
   times = [ ]
   brights = [ ]

   Find the square brackets on the keypad **[ctrl] [left parenthesis]** or on **[menu] > Built-ins > Lists**

   Also before the while loop, add a statement to start a time counter variable (**t**) at 0:

   t = 0

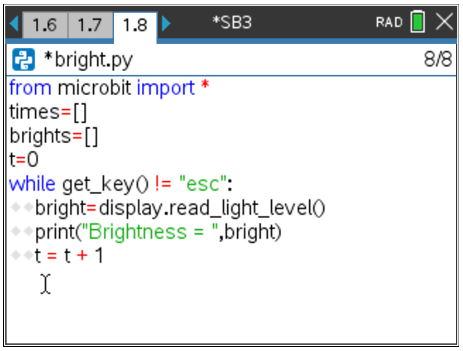
   Avoid using the word 'time' as a variable because there is a time module. Also, it is a good practice to pluralize list names because they contain many values.



```
from microbit import *
times=[]
brights=[]
t=0
while get_key() != "esc":
◆◆bright=display.read_light_level()
◆◆print("Brightness = ",bright)
```

5. In the loop body, after the **print** statement, add a statement to increase the timer variable **t**. We will use a one second time interval between light readings, so use:

   ◆◆t = t + 1



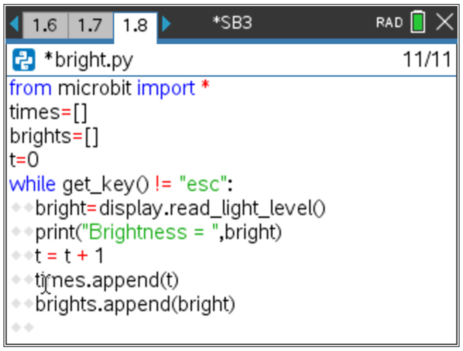
```
from microbit import *
times=[]
brights=[]
t=0
while get_key() != "esc":
◆◆bright=display.read_light_level()
◆◆print("Brightness = ",bright)
◆◆t = t + 1
```

6. **Append** (add) the values of **bright** and **t** to their respective lists using the statements:

   ◆◆times.append(t)
   ◆◆brights.append(bright)

   .append( ) is found on **[menu] > Built-ins > Lists**

   These statements add the current bright value and t (time) value to the lists.



```
from microbit import *
times=[]
brights=[]
t=0
while get_key() != "esc":
◆◆bright=display.read_light_level()
◆◆print("Brightness = ",bright)
◆◆t = t + 1
◆◆times.append(t)
◆◆brights.append(bright)
◆◆
```

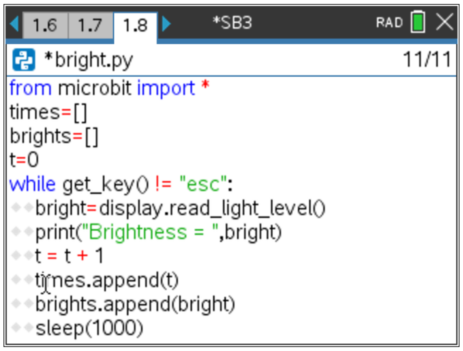
7. To control the timing interval of the sampling add:

   ◆◆sleep(1000)

   after the two **.append** statements. This pauses data collection for one second between samples. **sleep( )** is found on:

   [menu] > BBC micro:bit > Commands

   Recall that when using micro:bit, ***sleep( )*** *uses milliseconds for its argument. This sampling occurs once a second.*

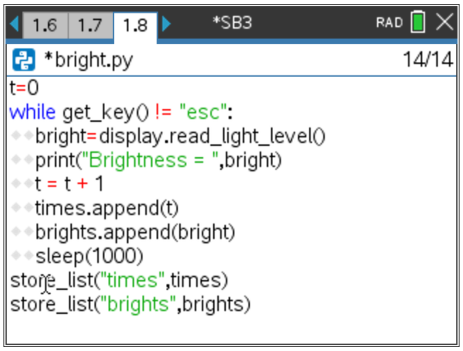


```
from microbit import *
times=[]
brights=[]
t=0
while get_key() != "esc":
◆◆bright=display.read_light_level()
◆◆print("Brightness = ",bright)
◆◆t = t + 1
◆◆times.append(t)
◆◆brights.append(bright)
◆◆sleep(1000)
```

8. After the loop body (*no more indenting!*), **store** the two python **list**s into two TI-Nspire lists using the same names in both environments:

   Start at the *beginning* of a new line (no indent spaces!).
   Find **store_list()** on **[menu] > BBC micro:bit > Commands**. It takes two arguments: the “TI-Nspire list name” in quotes and the python list name without quotes. Follow the tooltip prompts.

   In this program the python lists are only stored to TI-Nspire lists just before the program ends when you press **[esc]** *to exit the loop*

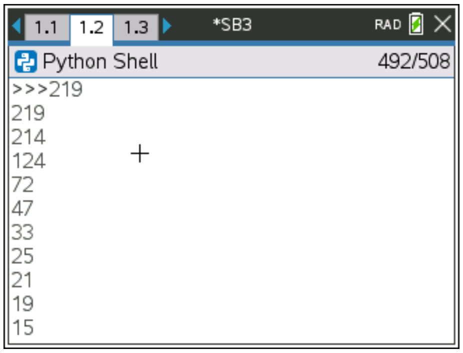


```
t=0
while get_key() != "esc":
  bright=display.read_light_level()
  print("Brightness = ",bright)
  t = t + 1
  times.append(t)
  brights.append(bright)
  sleep(1000)
store_list("times",times)
store_list("brights",brights)
```

9. Run the program. Start with the micro:bit close to your light source. An exposed light bulb or a smartphone flashlight work well. Slowly but *steadily* move the micro:bit away from the light at a uniform rate until the brightness reading is less than 10.
   Press **[esc]** to end the program.

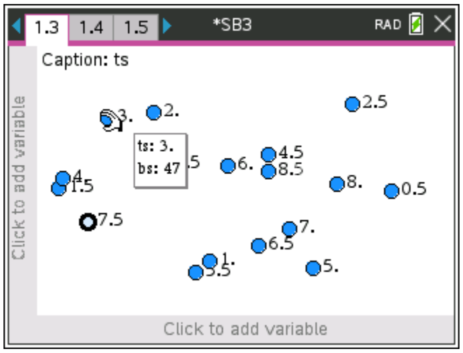
   Repeat the process until you feel you have ‘good’ data.

   Tip: A smartphone flashlight works great!

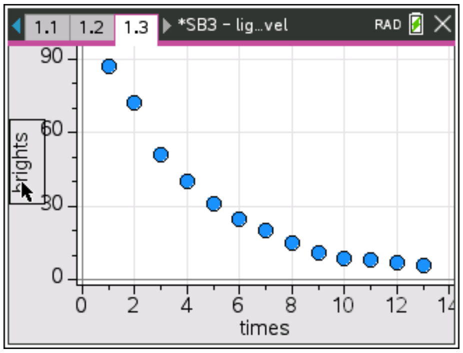


```
>>>219
219
214
124
72
47
33
25
21
19
15
```

10. When the program ends, add a page to your document by pressing **[ctrl] [doc]**. Select **‘Add Data & Statistics’.**
    You will see a page similar to this image. One of your lists appears with the points scattered around the screen. You now have to organize the plot…



11. Click the prompt '*Click to add variable*' on the bottom of the screen and select the list **times**. This is the *independent* variable.



Click the prompt '*Click to add variable*' on the left side of the screen and select the list **brights.** This is the *dependent* variable. This organizes your scatter plot like the image shown here.

You may have to run the program *several times* to get a good, smooth plot like the one shown here. You may also have to adjust the viewing window when using different data.

You may also adjust the time interval between samples but be sure to edit both the **sleep(1**000 **)** *argument and the counter value (t = t +* ***1****).*

What do you notice?

- *What pattern do you see? What mathematical model fits this physical phenomenon?*
- *Use the data analysis tools of the TI-Nspire to determine a good mathematical model for your data.*