



Unit 7: The RGB Array

Application: Smart Lights

In this final lesson, you will develop a program that uses the BRIGHTNESS sensor on the TI-Innovator to control the number of LEDs lit up on the TI-RGB Array.

Objectives:

- Use a sensor to control the TI-RGB Array
- Use a keypress to terminate the program

The TI-Innovator has an onboard BRIGHTNESS sensor. Imagine a lighting system that adapts to the brightness of the room: When the room is bright, little extra illumination is needed. As the room darkens, the number of extra lights needed increases. Using the TI-RGB Array we can simulate this lighting system by making more LEDs light up as the BRIGHTNESS decreases. This application will build that lighting system simulation.

1. Begin a new program – we call it **britelites()** - with the usual starting commands:

Start with **Send("CONNECT RGB")**

and write a **While** loop that monitors the keypress so that the loop ends when **[esc]** is pressed:

While getKey(0) ≠ "esc"

After the loop, make sure all LED's are turned off using

Send "SET RGB ALL 0 0 0".

2. Start the body of the **While** loop by reading the BRIGHTNESS sensor using

Send "READ BRIGHTNESS"

Get b

3. The **BRIGHTNESS** sensor gives a value from 0 to 100, but you can customize this range for your own lighting situation. Sometimes 0 is difficult to attain so we'll settle for 5 as a lower bound and we decided that 50 is 'bright enough'. Note the two **If statements** we added to the program:

If b < 5 : b := 5

If b > 50 : b := 50

You may choose to use different limits depending on your lighting situation, but the maximum range is 0 to 100. You may find a smartphone with a 'flashlight' helpful to get a wider range of values from the sensor.

Your task is to devise a 'conversion formula' to change the brightness value into a number of LEDs to light up. Convert the brightness value **b**, ranging from 5 to 50 into an appropriate number of LEDs **n**, ranging from 16 to 0. Use the **int()** function since we always want to control a whole number of LEDs.

Hint: The equation of the line through two points (a, b) and (c, d) is:

$$y = m * (x - a) + b \text{ where } m \text{ is the slope of the line and}$$

$$m = (d - b) / (c - a)$$

Our two points are (5,16) and (50,0). Yours might be different.

```

2.1 2.2 2.3 ▶ *10MoC...Sim RAD 0/11
* britelites
Define britelites()=
Prgm
Send "CONNECT RGB"
While getKey(0)≠"esc"
[ ]
EndWhile
Send "SET RGB ALL 0 0 0"

```

```

2.1 2.2 2.3 ▶ *10MoC...Sim RAD 7/15
* britelites
Prgm
Send "CONNECT RGB"
While getKey(0)≠"esc"
Send "READ BRIGHTNESS"
Get b
[ ]
EndWhile
Send "SET RGB ALL 0 0 0"

```

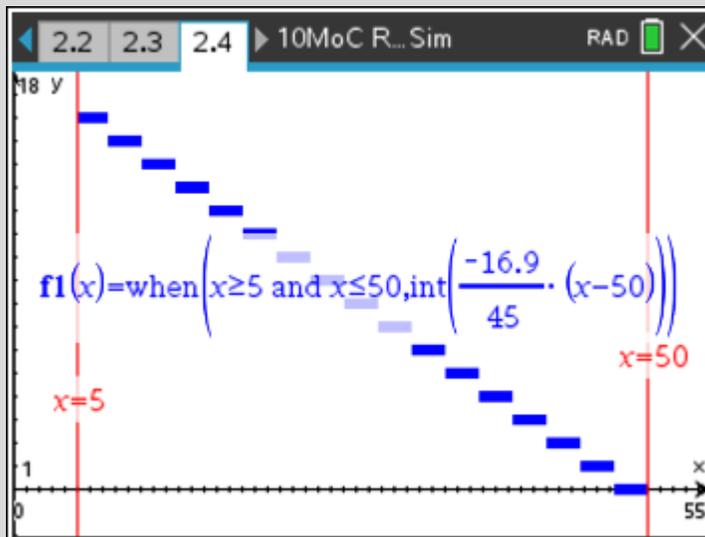
```

2.2 2.3 2.4 ▶ *10MoC...Sim CAPS RAD 10/31
* britelites
While getKey(0)≠"esc"
Send "READ BRIGHTNESS"
Get b
If b<5:b:=5
If b>50:b:=50

```




Teacher Tip: If statements may be necessary to handle the extreme conditions. We don't want to cause an error on the TI-Innovator.



The linear relation used in the program below gives 17 intervals between 5 and 50 so that 0 to 16 LEDs will be lit at some point.

Sample solution:

```

Define britelites()=
Prgm
:Send "CONNECT RGB"
:While getKey(0)!="esc"
:  Send "READ BRIGHTNESS"
:  Get b
:  If b<5:b:=5           below 5 is too dark
:  If b>50:b:=50       50 is 'bright enough'

:  n:=int((-16.9)/(45)*(b-50))
:  DispAt 1,b,n        for information purposes

:  For i,1,n
:    Send "SET RGB eval(i-1) 255 255 255"
:  EndFor

:  For i,n+1,16
:    Send "SET RGB eval(i-1) 0 0 0"
:  EndFor

:EndWhile
:Send "SET RGB ALL 0 0 0"
:EndPrgm
    
```