Unit 7: The TI-RGB Array | **Skill Builder 3: Sequencing**

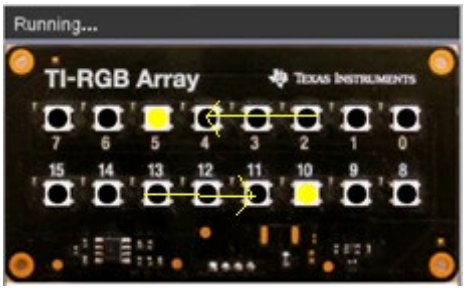
In this lesson, you will learn to control two LEDs at once in a loop to create a 'marquee' effect.

Objectives:

- Use a **for** loop to light up a single varying LED.
- Use a mathematics expression to control another LED at the same time.

The overhead sign outside of some movie theatres has a border of lights flashing in sequence like ants marching. You can create a similar effect on the TI-RGB Array by turning on and off lights *in sequence*.

Your program in this lesson will light up two LEDs at a time, one on the top row going from right-to-left (0 to 7) and one on the bottom row going from left-to right (15 to 8). What is the relationship between the top row sequence and the bottom row sequence?



(demo3.1.gif)

1. Begin a new Python Hub Project.

   Make a variable using the **rgb_array()** constructor and use the **esc** keypress loop as in the last lesson.

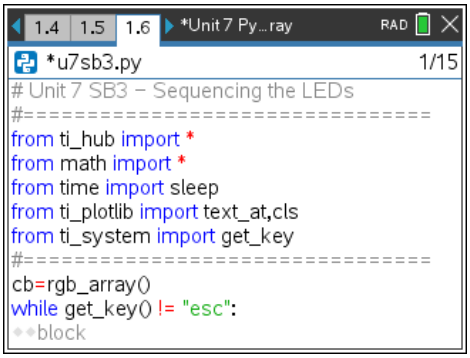
   We use the variable **cb** again, but you are free to choose your own variable.

   Add the statement

   while get_key() != "esc":

   block

   found at **menu > TI Hub > Commands**.



2. Use a **for** loop to light up the top row in sequence from right to left (0 to 7).

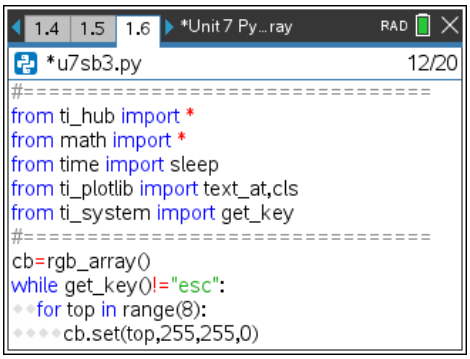
   for top in range (8):

   cb.set(top, 255,255,0) *(This is yellow.)*

   Use the variable **top** because this controls the top row.

   Remember that **range(8)** processes the numbers from 0 to 7.

   Test your program now.

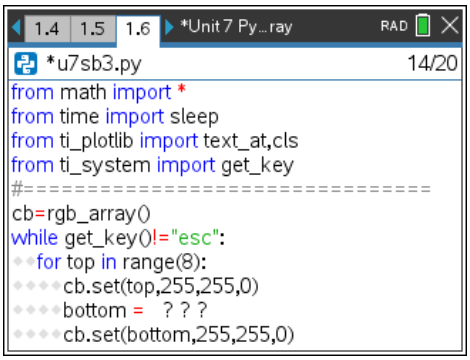


3. All 8 LEDs light up very quickly and, at the end of the program, all the top row LEDs are on.

   Next deal with the **bottom** row. The bottom row must go from 15 to 8. What is the relationship between bottom and top?

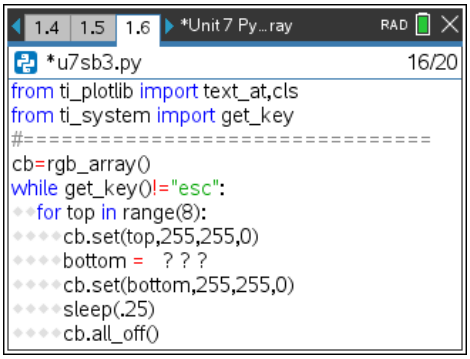
   bottom = ? ? ?

   cb.set(bottom, 255, 255, 0)

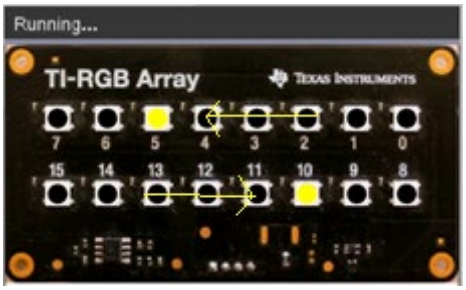


4. Run the program now. All 16 LEDs light up very quickly. Add two statements: a **sleep()** statement to slow things down and a statement to turn ***all*** LEDs off at the bottom of the loop block.

   sleep(.25)
   cb.all_off()



5. Try your program again. Adjust the **sleep()** value and perhaps add another **sleep()** after **all_off()**.



(demo 3.1.gif)