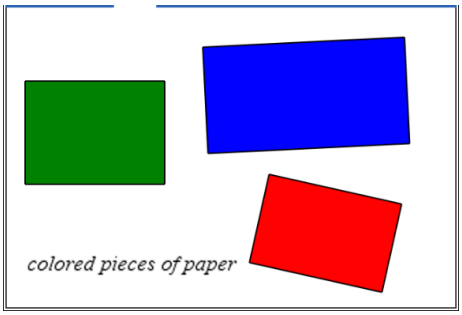
Unit 5: Rover's Sensors | **Skill Builder 3: Rover's Sensors**

In this lesson, you will learn to use Rover's color sensor to change direction when a color is detected.
**Colored paper or large colored shapes are needed for this lesson.

Objectives:

- Use the color sensor to detect and react to a color

Rover has a color sensor on the bottom. You can see a light shining on the floor under the color sensor. The light helps Rover to see the color beneath it. First write a 'test' program to see what kind of values the color sensor produces and then you will be able to write a program to react to different colors. You will need some colored paper like construction paper or just print out some colored shapes like the rectangles at the right. They should be large enough for Rover to 'see'.



On the menu **[math] ti_rover… Inputs**, there are five different color measurements available. The function **color_measurement()** returns a value from 1 to 9 where:

1=red, 2=green, 3=blue, 4=cyan, 5=magenta, 6=yellow, 7=black,

8=white, 9 = gray

The other four measurements return the amount of the indicated color in the range 0…255, as shown on the menu.

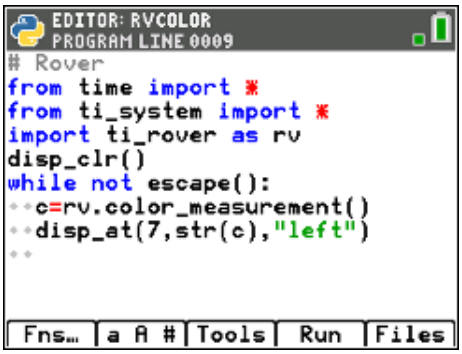
```
2:color_measurement()      1-9
3:red_measurement()      0-255
4:green_measurement()    0-255
5:blue_measurement()     0-255
6:gray_measurement()     0-255
```

Tip: do not confuse **rv.color_measurement()** (an input command that reads the color sensor on the bottom of Rover) with **rv.color()** (an output command that controls the color LED on top of Rover.

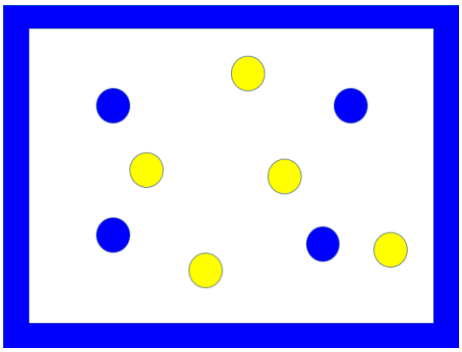
1. Here is a short 'test' program using the Rover Coding template to determine the values that the color measurement functions produce.

```
while not escape( ):
    c = rv.color_measurement()
    disp_at(7, str(c), "left")
```

Try all five color measurements (**color**_, **red**_, **green**_, **blue**_, and **gray**_) on various colored surfaces and observe the values displayed.



2. Depending on your floor color (preferably white), make some colored pieces of paper (like 'sticky notes' or colorful construction paper) to place in front of Rover so that Rover can 'see' the change in color from the floor. Test these patches first to see what colors Rover 'sees'.



Write a program to get Rover to react to the different colors. In the sample page to the right, the blue border should keep Rover on the page:

- When Rover 'sees' blue, turn around.
- When Rover 'sees' yellow, turn right or left.

You can also add statements to the program to control the color LED on Rover and to display the color value seen.

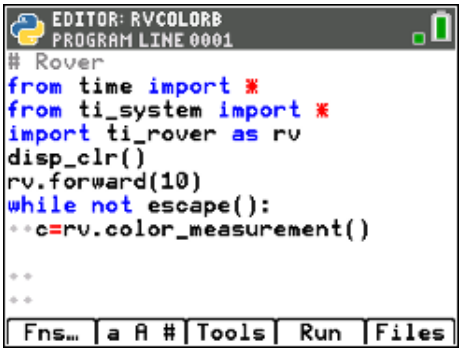
3. Copy the color 'test' program you started. Add a statement before the loop to start Rover moving forward:

```
rv.forward(10)
```

Next, inside the loop, monitor the color below Rover…

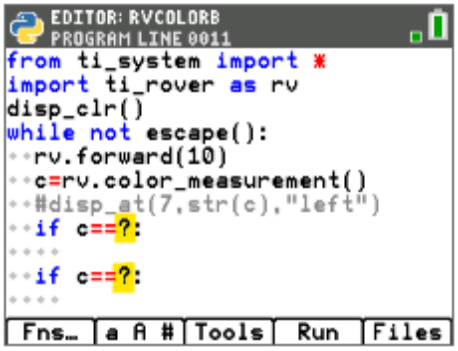
```
    c = rv.color_measurement()
```

When **color** is blue, make Rover turn 180 degrees and, when the color is yellow, make Rover turn 90 degrees. This requires two **if** statements.



4. Add two **if** statements and use the values you got from testing your **color** samples in place of the **?**s:

```
if c == ?:
    
if c == ?:
    
```



5. The two **if** blocks are similar. Just remember blue - 180 degrees and yellow – 90 degrees.

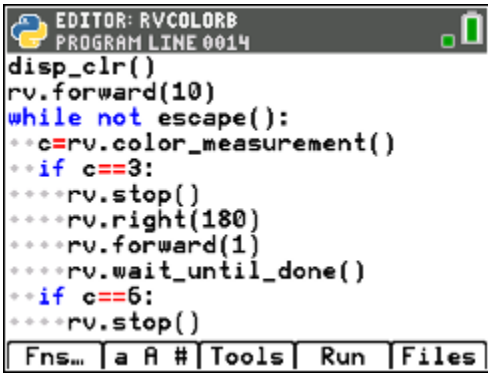
Each block will:

a) Stop Rover
b) Turn (90 or 180)
c) Go forward a little bit to move away from the colored spot
d) Tell the TI-84 to *wait until* Rover is *done* with these three tasks

Try it yourself before proceeding…

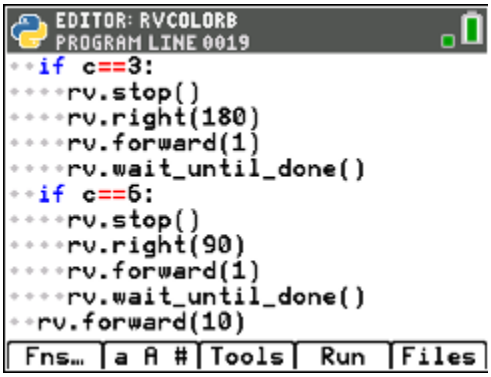
6. Do your **if** blocks resemble this?

♦♦**if c == 3:** (*Blue; use another number if your color is not blue.)*
♦♦♦♦ **rv.stop()**
♦♦♦♦**rv.right(180)**
♦♦♦♦**rv.forward(1)** **# to move off the blue spot**
♦♦♦♦**rv.wait_until_done()**



7. When Rover 'sees' blue (when c==3) it turns around and goes forward a bit to get away from the blue spot.
When Rover 'sees' yellow (when c==6) it turns 90 degrees and goes forward a bit.

At the bottom of the **while** loop get Rover moving again.

♦♦**rv.forward(10)**

Try using other color measurement functions and other driving actions to see what works best for your environment.