Activity 3: Button with RGB LED:

Description:

This circuit includes RGB LED and a button that turns on different colors of RGB LED in every push of the button. When pressed once color red will turn on, pressed twice color green will turn on, pressed thrice color blue will turn on.

Materials:

1 Arduino Uno R3
1 Breadboard
1 Pushbutton
1 10 kΩ Resistor
3 400 Ω Resistor
1 LED RGB
Wires

Codes:

```
const int buttonPin = 3;
const int red = 13;
const int green = 6;
const int blue = 11;
int counter = 0;

void setup() {
  pinMode(buttonPin, INPUT);
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}

void loop() {
  int buttonState;
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    counter++;
    delay(100);
  }
  else if (counter == 0) {
    digitalWrite(red, LOW);
    digitalWrite(green, LOW);
    digitalWrite(blue, LOW);
  }
 else if (counter == 1) {
    digitalWrite(red, HIGH);
    digitalWrite(green, LOW);
    digitalWrite(blue, LOW);
  }
 else if (counter == 2) {
    digitalWrite(red, LOW);
    digitalWrite(green, HIGH);
    digitalWrite(blue, LOW);
  }
  else if (counter == 3) {
    digitalWrite(red, LOW);
    digitalWrite(green, LOW);
    digitalWrite(blue, HIGH);
  }
  else {
    counter = 0;
  }
}
```

Explanation on Codes:

const int buttonPin = 3;
const int red = 13;
const int green = 6;
const int blue = 11;
int counter = 0;

- This section declares the variables with corresponding names. The const you see before the int declares that the variable will be constant, meaning the value cannot be changed.

void setup() {

```
  pinMode(buttonPin, INPUT);
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}
```

- This section sets up the pins to be either output or input.
- As you might have noticed, there is only one RGB LED in the board and we declared 3 variables. Because of the RGB LED, there are exactly 4 pins under the LED.
- Each pin has corresponding colors except for 1 which is a cathode that you are supposed to connect it in negative flow on the board.
- The colors are red, green, and blue.

```
void loop() {
  int buttonState;
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    counter++;
    delay(100);
  }
```

- It started on “int buttonState;” to declare the name “buttonState” on the code, otherwise it will result in compiler error. You can put it above the void setup.
- Next is the declaration of the readings on the buttonPin or the button to be the “buttonState”.
- Next is the if statement with the condition of “if the button is pressed it will add 1 to the counter with a delay of 100 millisecond or 0.1 second.” This means that the counter will store the number of presses on the button in the variable “counter”.

```
else if (counter == 0) {
    digitalWrite(red, LOW);
    digitalWrite(green, LOW);
    digitalWrite(blue, LOW);
  }
```

```
else if (counter == 1) {
   digitalWrite(red, HIGH);
   digitalWrite(green, LOW);
   digitalWrite(blue, LOW);
 }
else if (counter == 2) {
   digitalWrite(red, LOW);
   digitalWrite(green, HIGH);
   digitalWrite(blue, LOW);
 }
 else if (counter == 3) {
   digitalWrite(red, LOW);
   digitalWrite(green, LOW);
   digitalWrite(blue, HIGH);
 }
```

```
 else if (counter == 3) {
     digitalWrite(red, LOW);
     digitalWrite(green, LOW);
     digitalWrite(blue, HIGH);
  }
```

`counter == 3:` : The condition

```
digitalWrite(red, LOW);
digitalWrite(green, LOW);
digitalWrite(blue, HIGH);
```

: The action if the condition is true

- The following codes are the else if statements. On their parenthesis, there are conditions of "if the "counter" is equal to a specific number or number of presses on the button it will light up one of the colors on the RGB LED."

```
else {
   counter = 0;
 }
```

- Lastly, if the counter is greater than 3, it will return back to 0.