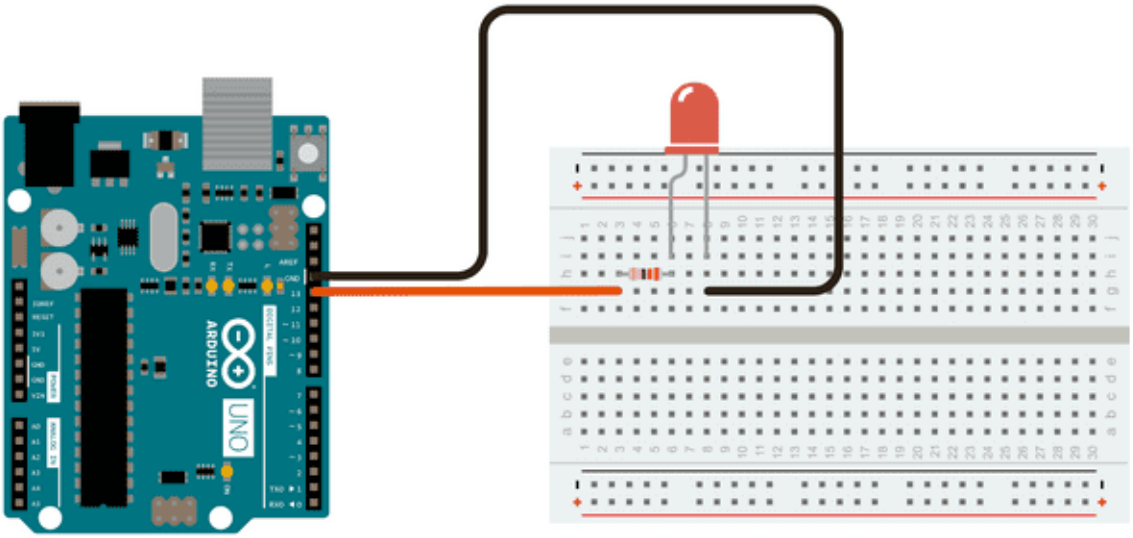
Exercise 1: Blink an LED

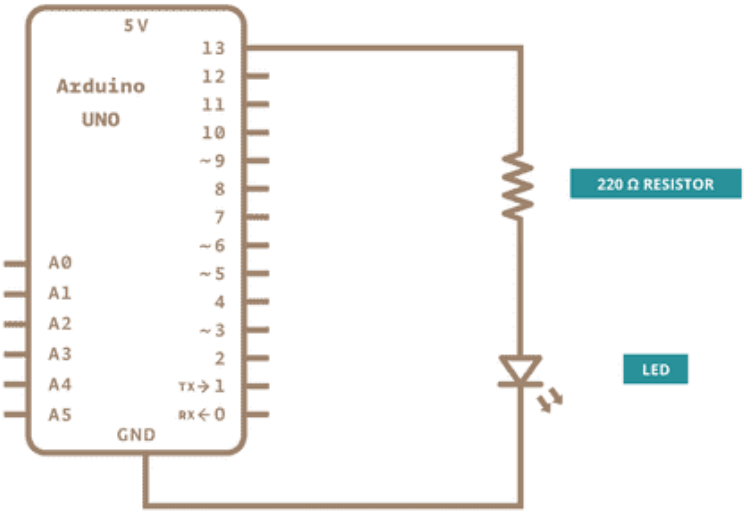
Build the circuit below, pay attention to where the wires plug in!

Use the 220 ohm resistor (red, red, black, black, brown)

The LED needs to be the right way around, it will have a shorter leg (called the cathode), which should be on the right side, connected to the black wire and ground.



The circuit can also be represented as the circuit diagram below, see that the LED and resistor have their own simplified pictures that make drawing the circuit easier.

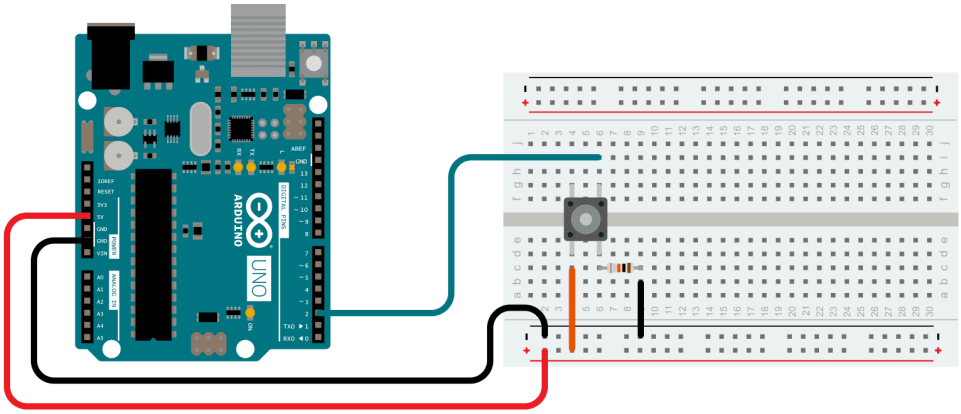


Exercise 2: Use a Push Button

Add the circuit below to your Arduino to use a push button.

Use a 10 kOhm resistor. (brown, black, black, red, brown)

Don’t unplug the LED or change the circuit from before!

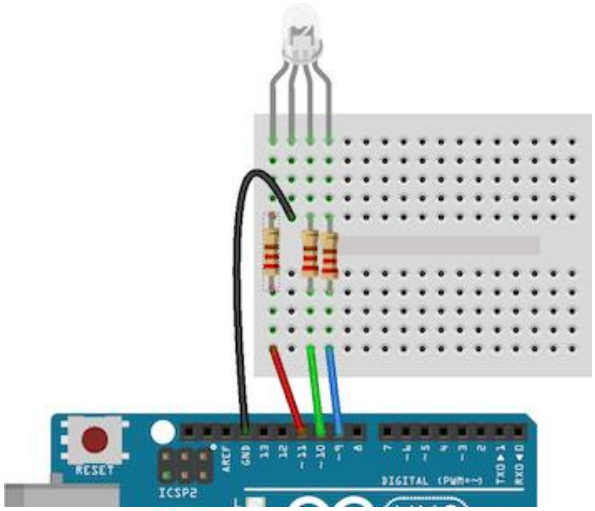


Program the Arduino with the code below

```
const int buttonPin = 2;  // the number of the pushbutton pin
const int ledPin = 13;    // the number of the LED pin

// variables will change:
int buttonState = 0;  // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Exercise 3: Light an RGB

Clear your board of all the other circuits, and now build the circuit below with an RGB (Red Green Blue) LED. This is a single component with three LEDs inside.



Upload the code below to the Arduino. The LED should light up red, green and blue in sequence.

The line RGB_color(255, 0, 0) sets the colour of the RGB, setting how much red, green and blue should be lit up, with a number between 0 and 255.

What happens if you change it to RGB_color(255,255,255)?

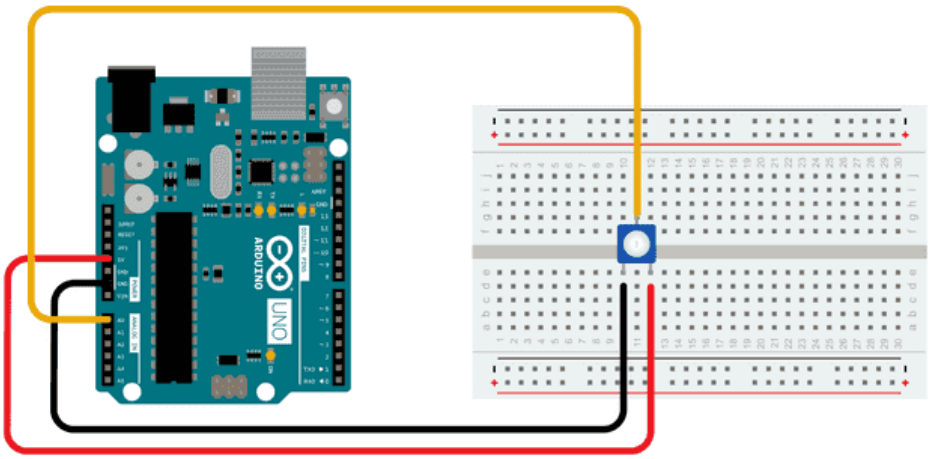
Try different combinations of numbers and see which colours you can make.

```
int red_light_pin= 11;
int green_light_pin = 10;
int blue_light_pin = 9;
void setup() {
  pinMode(red_light_pin, OUTPUT);
  pinMode(green_light_pin, OUTPUT);
  pinMode(blue_light_pin, OUTPUT);
}
void loop() {
  RGB_color(255, 0, 0); // Red
  delay(1000);
  RGB_color(0, 255, 0); // Green
  delay(1000);
  RGB_color(0, 0, 255); // Blue
  delay(1000);
}
void RGB_color(int red_light_value, int green_light_value, int blue_light_value) {
  analogWrite(red_light_pin, red_light_value);
  analogWrite(green_light_pin, green_light_value);
  analogWrite(blue_light_pin, blue_light_value);
}
```

Exercise 4: Read a Potentiometer

A potentiometer is a “resistor” (something which opposes or restricts the flow of electricity) which can be adjusted

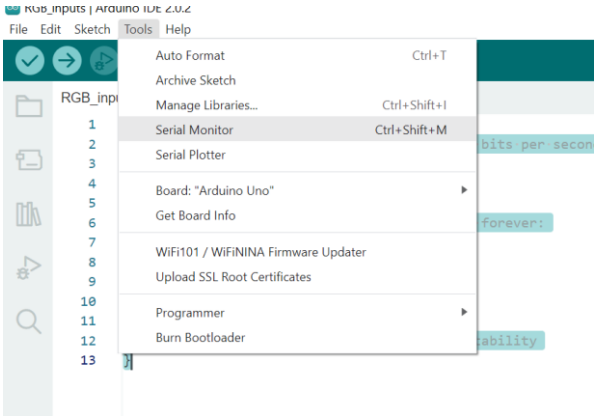
Build the circuit below, the middle pin of your potentiometer should go to A0, the left to GND, and the right to 5V on the Arduino.



Upload the code below to the Arduino

```
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);  // delay in between reads for stability
}
```

Open the “Serial Monitor”, you can find it under the “Tools” menu of the software.



What happens now when you turn the potentiometer?

Challenge Exercise: Combine any of the above circuits

Can you combine any of the exercises above by yourself?

For example, you could try use a pushbutton to control the RGB, or the potentiometer to control how bright the LED is. Think about which lines of code you might need to use.

Some examples of INPUTS are:

In exercise 2, the line

```
buttonState = digitalRead(buttonPin);
```

takes a digital reading (1 or 0) of buttonPin (which we defined as button 2 earlier on)

In exercise 4, the line

```
int sensorValue = analogRead(A0);
```

creates a variable called sensorValue, and assigns it the value of an analog reading of the pin A0. (a number between 0 and 1023)

Think about which lines then use these numbers from the examples above and how you can modify the code.