ARDUINO

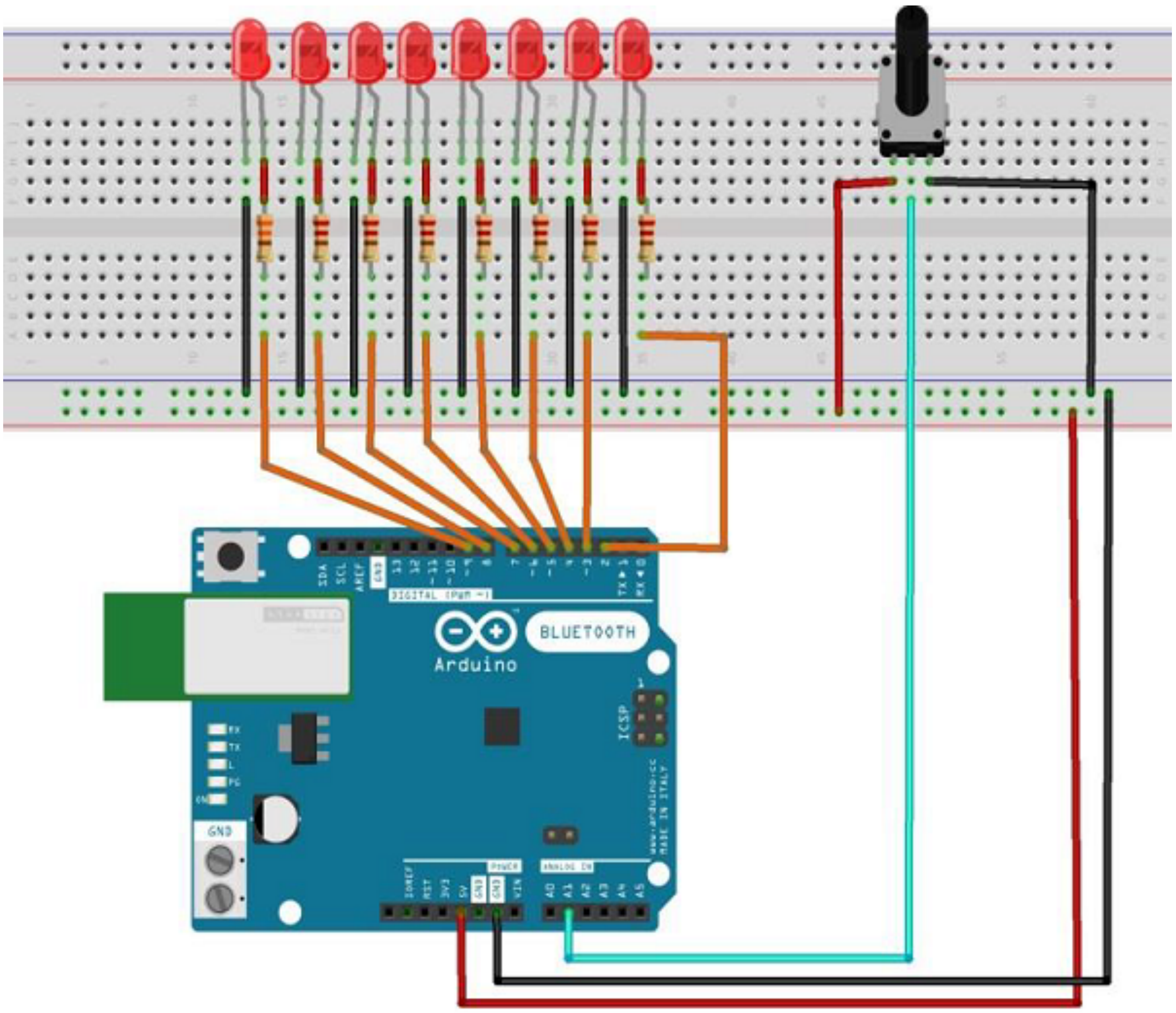
LED BAR GRAPH

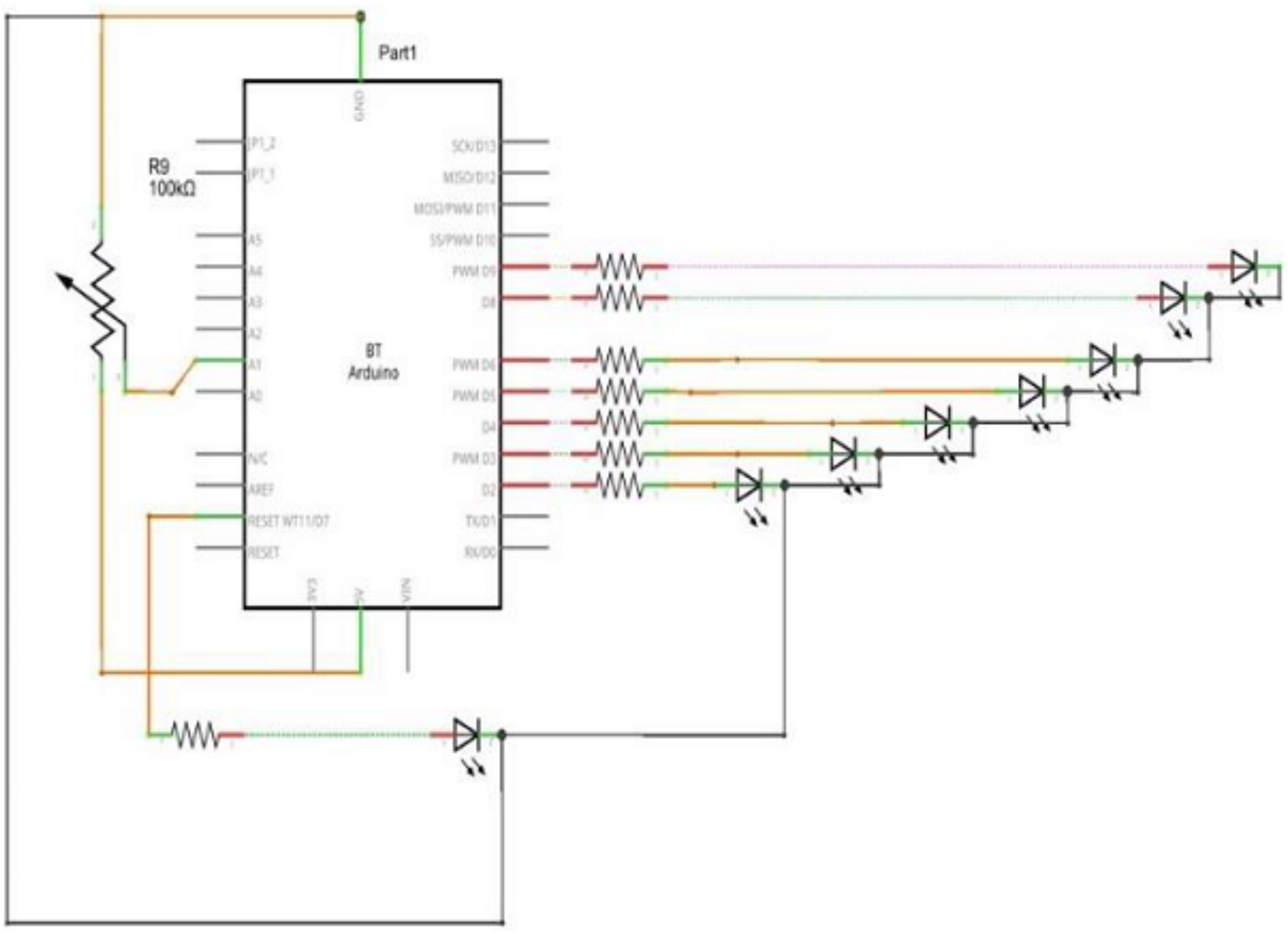
COMPONENTS REQUIRED

- 1 x Breadboard
- 1 x Arduino Uno R3
- 1 x 5k ohm variable resistor (potentiometer)
- 2 x Jumper
- 8 x LED or you can use (LED bar graph as shown in the image below)

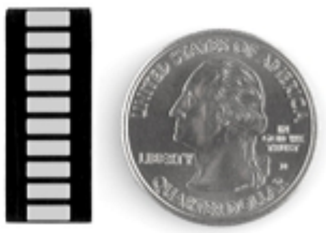
PROCEDURE

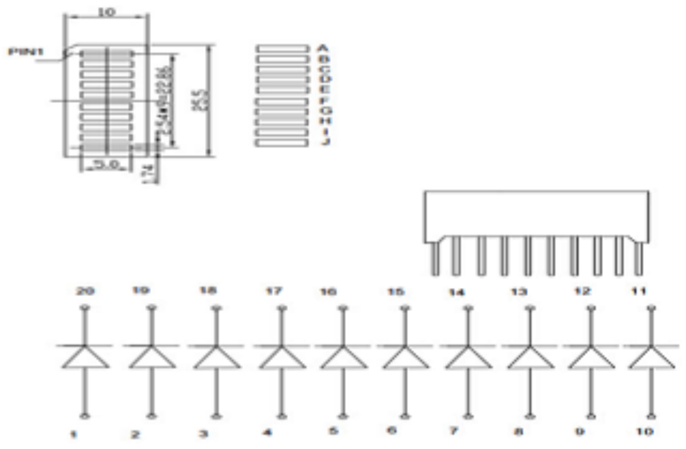
Follow the circuit diagram and hook up the components on the breadboard as shown in the image given below.





10 Segment LED Bar Graph





These 10-segment bar graph LEDs have many uses. With a compact footprint, simple hookup, they are easy for prototype or finished products. Essentially, they are 10 individual blue LEDs housed together, each with an individual anode and cathode connection.

Note – The pin out on these bar graphs may vary from what is listed on the datasheet. Rotating the device 180 degrees will correct the change, making pin 11 the first pin in line.

CODE

```
/*
  LED bar graph
  Turns on a series of LEDs based on the value of an analog sensor.
  This is a simple way to make a bar graph display.
  Though this graph uses 8LEDs, you can use any number by
    changing the LED count and the pins in the array.
  This method can be used to control any series of digital
    outputs that depends on an analog input.
*/

// these constants won't change:
const int analogPin = A0; // the pin that the potentiometer is attached to
const int ledCount = 8; // the number of LEDs in the bar graph
int ledPins[] = {2, 3, 4, 5, 6, 7, 8, 9}; // an array of pin numbers to which LEDs
are attached

void setup() {
  // loop over the pin array and set them all to output:
  for (int thisLed = 0; thisLed < ledCount; thisLed++) {
    pinMode(ledPins[thisLed], OUTPUT);
  }
}

void loop() {
  // read the potentiometer:
  int sensorReading = analogRead(analogPin);
  // map the result to a range from 0 to the number of LEDs:
  int ledLevel = map(sensorReading, 0, 1023, 0, ledCount);
  // loop over the LED array:
  for (int thisLed = 0; thisLed < ledCount; thisLed++) {
    // if the array element's index is less than ledLevel,
    // turn the pin for this element on:
    if (thisLed < ledLevel) {
      digitalWrite(ledPins[thisLed], HIGH);
    }else { // turn off all pins higher than the ledLevel:
      digitalWrite(ledPins[thisLed], LOW);
    }
  }
}
```

CODE TO NOTE

The sketch works like this: first, you read the input. You map the input value to the output range, in this case ten LEDs. Then you set up a **for-loop** to iterate over the outputs. If the output's number in the series is lower than the mapped input range, you turn it on. If not, you turn it off.

RESULT

You will see the LED turn ON one by one when the value of analog reading increases and turn OFF one by one while the reading is decreasing.