NAME: SHEKHAR SUMAN		INDEX
CLASS: S.Y.B.Sc.I.T.	**ROLL No**: 48	
SUBJECT: EMBEDDED SYSTEM PRACTICAL		
COLLEGE: ST. ROCK'S DEGREE COLLEGE		

PR No:	TITLE	DATE	SIGN
1.A	Introduction to Arduino	16-12-2024	
1.B	Blinking LEDs.	27-01-2025	
2	Program using light sensitive sensors.	03-02-2025	
3	Program using temperature sensors.	10-02-2025	
4	Programs using gas sensors.	17-02-2025	

PRACTICAL: 1A

AIM: Introduction to Arduino and Breadboarding.

Arduino is an open-source electronics platform based on easy-to-use hardware and software. It is widely used for building electronic projects, prototyping, and learning programming. The most popular model, **Arduino Uno**, is based on the **ATmega328P microcontroller** and features both **digital and analog input/output (I/O) pins**.

Input Pins of Arduino Uno

1. Digital Input Pins (0 - 13)

- Pins 0 and 1 (RX/TX) → Used for serial communication (Receiving and Transmitting data).
- Pins 2 to 13 → Can be used as digital input pins to read HIGH (5V) or LOW (0V) signals.

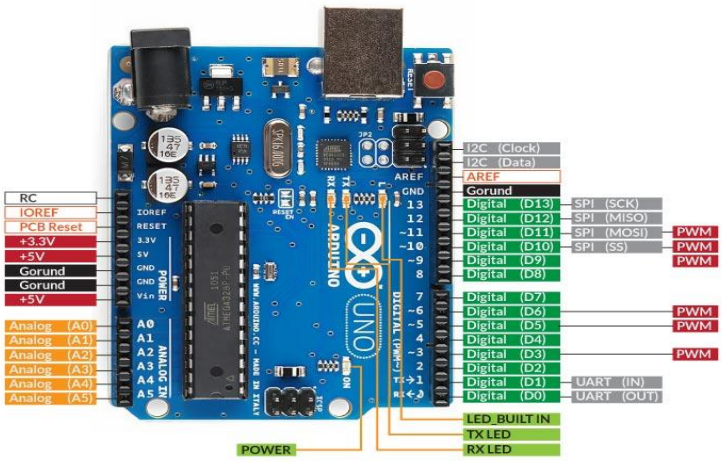
2. Analog Input Pins (A0 - A5)

- Pins A0 to A5 → Used to read analog signals from sensors (values range from 0 to 1023).
- These pins use a 10-bit Analog-to-Digital Converter (ADC) to convert analog voltage into a digital value.

3. Special Input Pins

- AREF (Analog Reference Pin) → Provides a reference voltage for analog input readings.
- RESET Pin → Used to reset the Arduino board when pulled LOW.

ARDUINO IMAGE



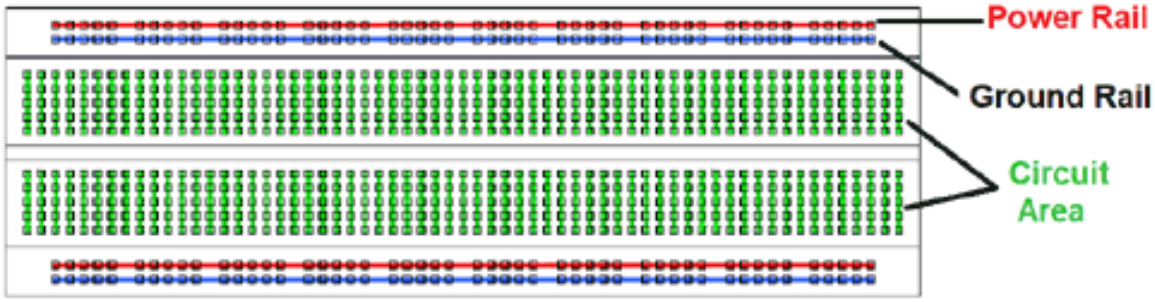
What is a Breadboard?

A breadboard is a reusable prototyping tool used to build and test electronic circuits without soldering. It consists of holes and internal metal strips that create electrical connections when components are inserted.

Breadboard Layout:

- Power Rails (Red & Blue/Black) → Used for power distribution (VCC & GND)
- Circuit Area (Green Section) → Used for placing components and wires
- Middle Gap → Separates the two halves, typically for integrated circuits (ICs)

BREADBOARD IMAGE



Breadboard Rules

1. Power and ground rails are connected horizontally, making it easier to distribute power.
2. Each vertical column in the circuit area is connected internally, meaning components placed in the same column will share a connection.
3. The middle gap isolates the two halves, allowing ICs to be placed in the center without shorting their pins.

PRACTICAL: 1B

AIM: Blinking of LEDs.

COMPONENTS REQUIRED:

1. **Arduino UNO** (Main microcontroller board)
2. **Breadboard** (For easy prototyping)
3. **LED** (Light-emitting diode, indicates output)
4. **Resistor (220Ω)** (Limits current to protect the LED)
5. **Jumper Wires** (To make electrical connections)

CONNECTION:

Step 1: Gather Components

- Take an **Arduino UNO** and a **breadboard**.
- Ensure you have an **LED, 220Ω resistor, and jumper wires** for connections.

Step 2: Identify LED Terminals

- **Anode (+) → Longer leg** → Connects to the **Arduino's digital pin 13**.
- **Cathode (-) → Shorter leg** → Connects to **GND through a resistor**.

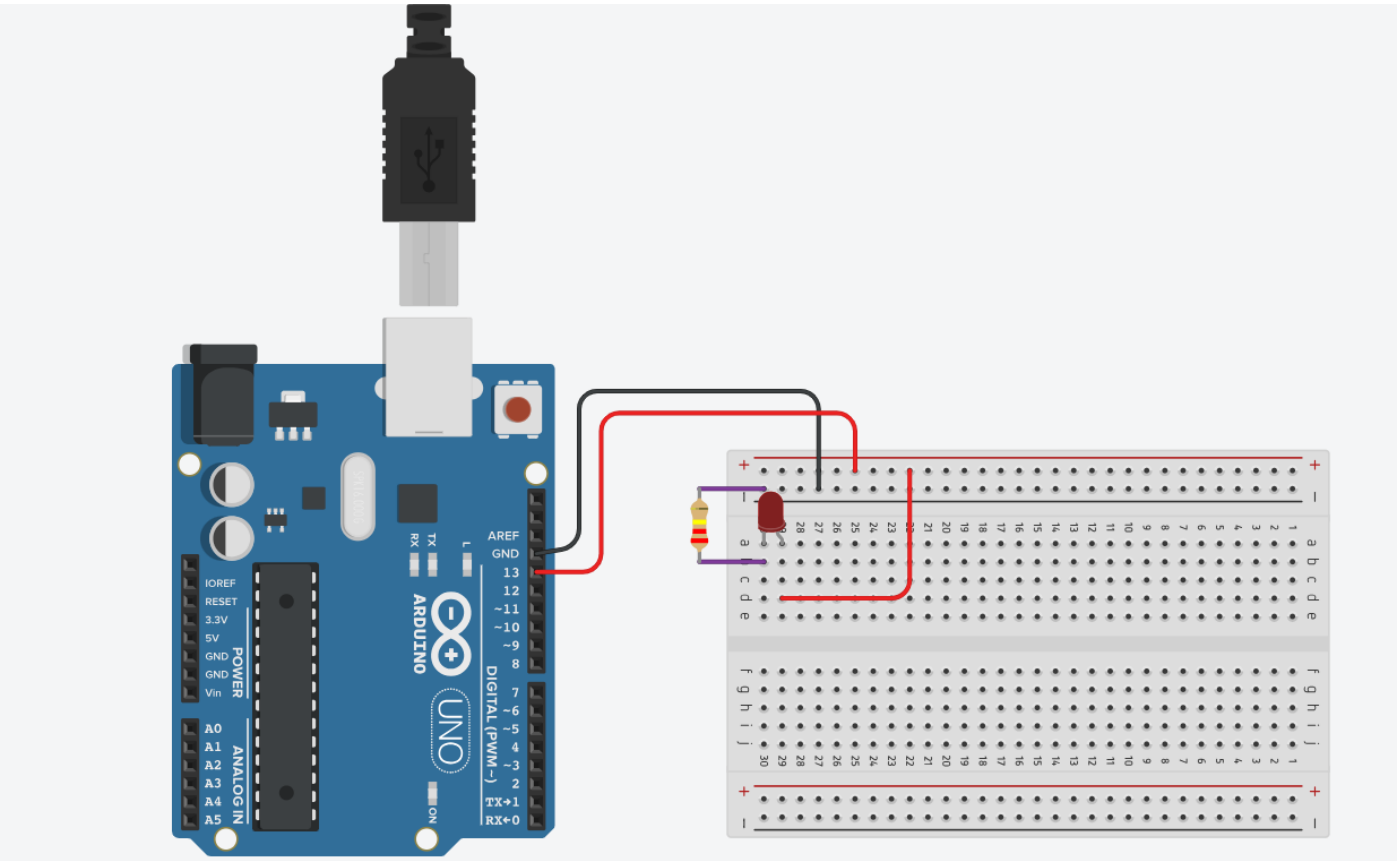
Step 3: Connecting the Resistor

- Insert the **220Ω resistor** into the breadboard.
- One end of the **resistor connects to the LED's cathode (-)**.
- The **other end of the resistor connects to GND** of Arduino (to limit current and protect the LED).

Step 4: Connecting the LED Anode

- The **LED's anode (+) connects to digital pin 13** on the Arduino board using a jumper wire.

CONNECTION DIAGRAM



CODE

```
void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

PRACTICAL: 2

AIM: Program using Light Sensitive Sensors.

Components Required:

- **Ambient Light Sensor (Phototransistor)**
  (Detects light by varying current)
- **1kΩ Resistor**
  (Limit current in circuits)
- **3 × AAA 1.5V Batteries**
  (providing 4.5V in series)
- **Battery Holder**
  (Holds Batteries)
- **Resistance Multimeter**
  (Measure electrical Resistance)

Connections:

1. **Prepare the Power Source:**
   - Insert **three AAA batteries** into the battery holder to create a 4.5V supply.
   - Identify the **positive (+)** and **negative (-)** terminals of the battery pack.

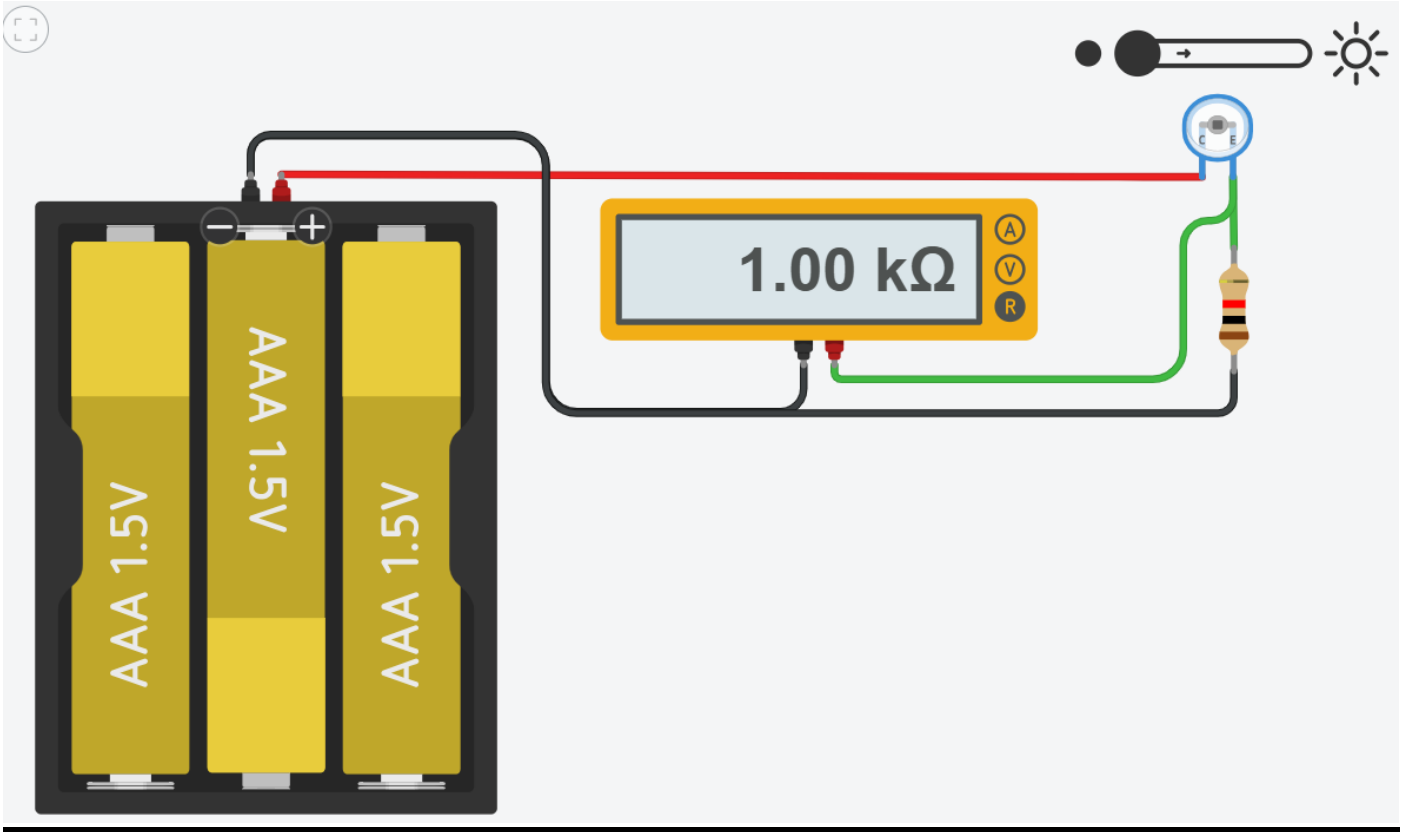
2. **Connect the Phototransistor:**
   - The phototransistor has **two terminals: Collector (C)** and **Emitter (E).**
   - Connect the **collector (C)** of the phototransistor to the **positive terminal (+)** of the battery pack.

3. **Attach the Resistor:**
   - Connect a **1kΩ resistor** between the **emitter (E)** of the phototransistor and the **negative terminal (-)** of the battery pack.

4. **Connect the Resistance Multimeter:**
   - Set the **multimeter** to the **resistance (Ω) mode** to measure the resistance of the phototransistor.
   - Connect the **positive (red) probe** of the multimeter to the **collector** of the phototransistor.
   - Connect the **negative (black) probe** to the **emitter** of the phototransistor.

5. **Observe the Readings:**
   - The resistance of the phototransistor changes based on the amount of ambient light it receives.
   - In **bright light**, the resistance decreases, allowing more current to flow.
   - In **low light or darkness**, the resistance increases, reducing the current flow.

DAIGRAM



PRACTICAL: 3

AIM: Program using Temperature Sensors.

Components Required:

- **Arduino Uno** (Microcontroller Board)
- **TMP36 Temperature Sensor (**Precision analog temperature sensor with linear voltage output.**)**
- **Jumper Wires** (Male-to-Male or Male-to-Female, as needed)

CONNECTION:

Step: 1: Identify TMP36 Pins:

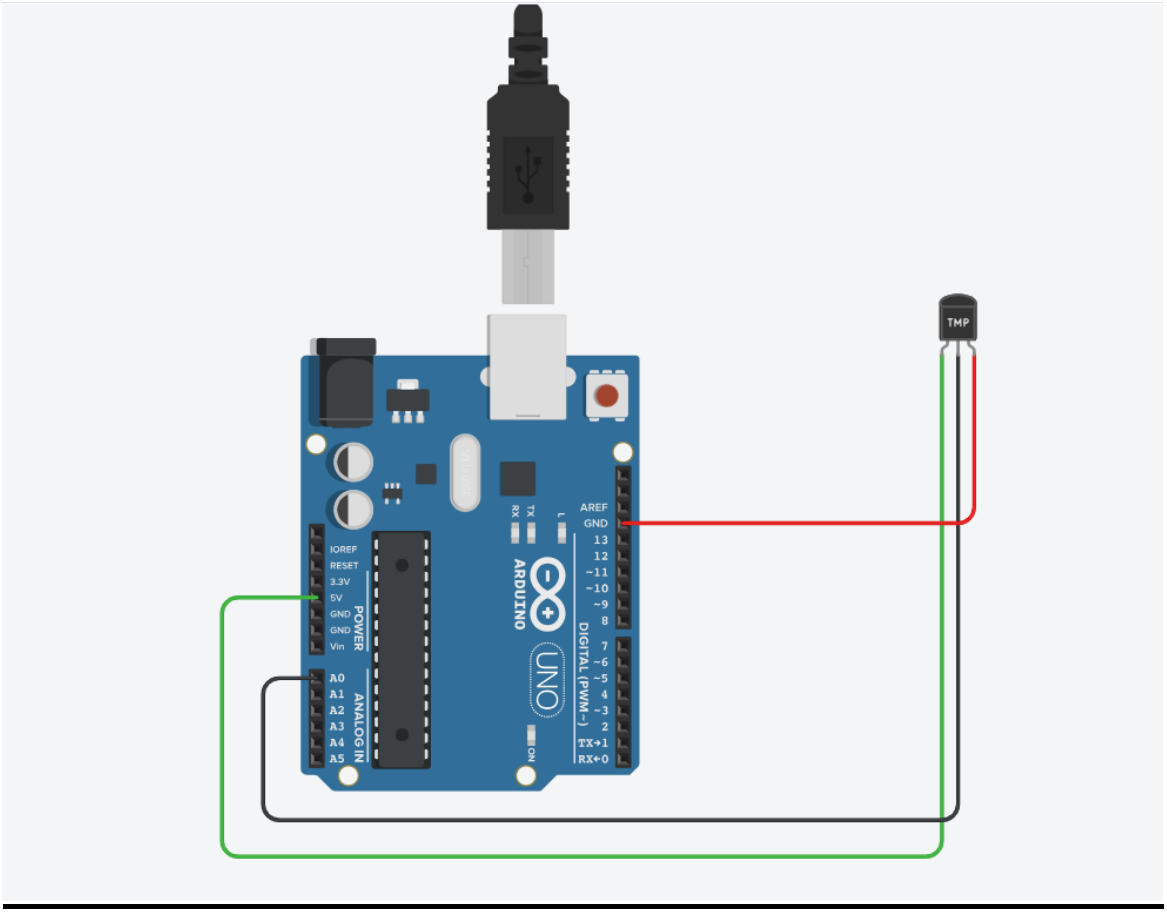
- The TMP36 sensor has three pins:

  1. VCC (Left pin): Power supply (3.3V or 5V)
  2. VOUT (Middle pin): Output voltage (temperature data)
  3. GND (Right pin): Ground

Step: 2: Connect the TMP36 to the Arduino Uno:

- Connect the VCC (left pin) of the TMP36 to the 5V pin on the Arduino.
- Connect the GND (right pin) of the TMP36 to the GND pin on the Arduino.
- Connect the VOUT (middle pin) of the TMP36 to A0 (Analog Pin 0) on the Arduino.

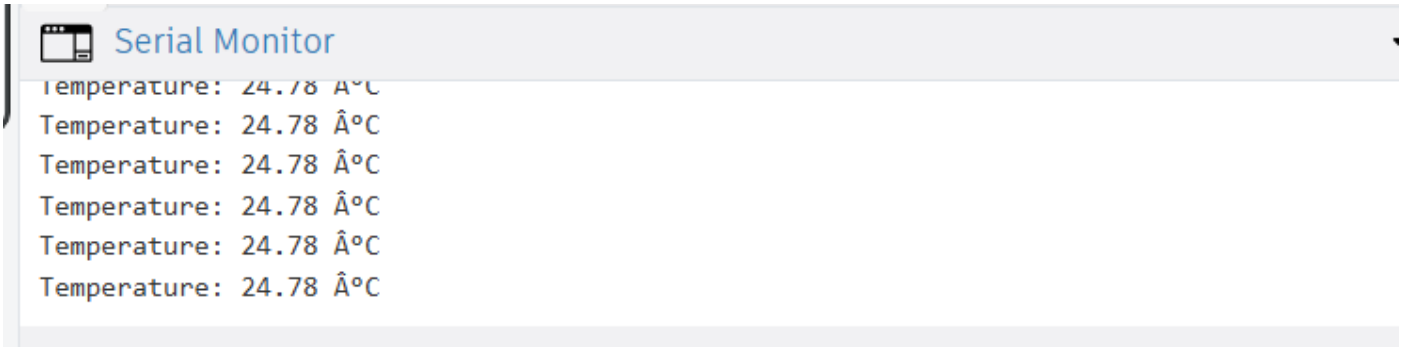
DIAGRAM



CODE

```
const int sensorPin = A0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(sensorPin);
  float voltage = sensorValue * (5.0 / 1023.0);
  float temperature = (voltage - 0.5) * 100;

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");

  delay(1000);
}
```

- The Output of this Shown on Serial Monitor Like this



PRACTICAL: 4

AIM: Program using Gas Sensors.

Components Required:

- **Arduino Uno** (Microcontroller Board)
- **2 Resistor(1 kΩ and 220 k Ω) (**220k **Ω** for protecting the LED, and 1k **Ω** for gas sensor**)**
- **Gas Sensor** (For Detecting the Gas)
- **LED (**For Blink when gas is detected)
- **Bread Board (**For Better Connection**)**

CONNECTION:

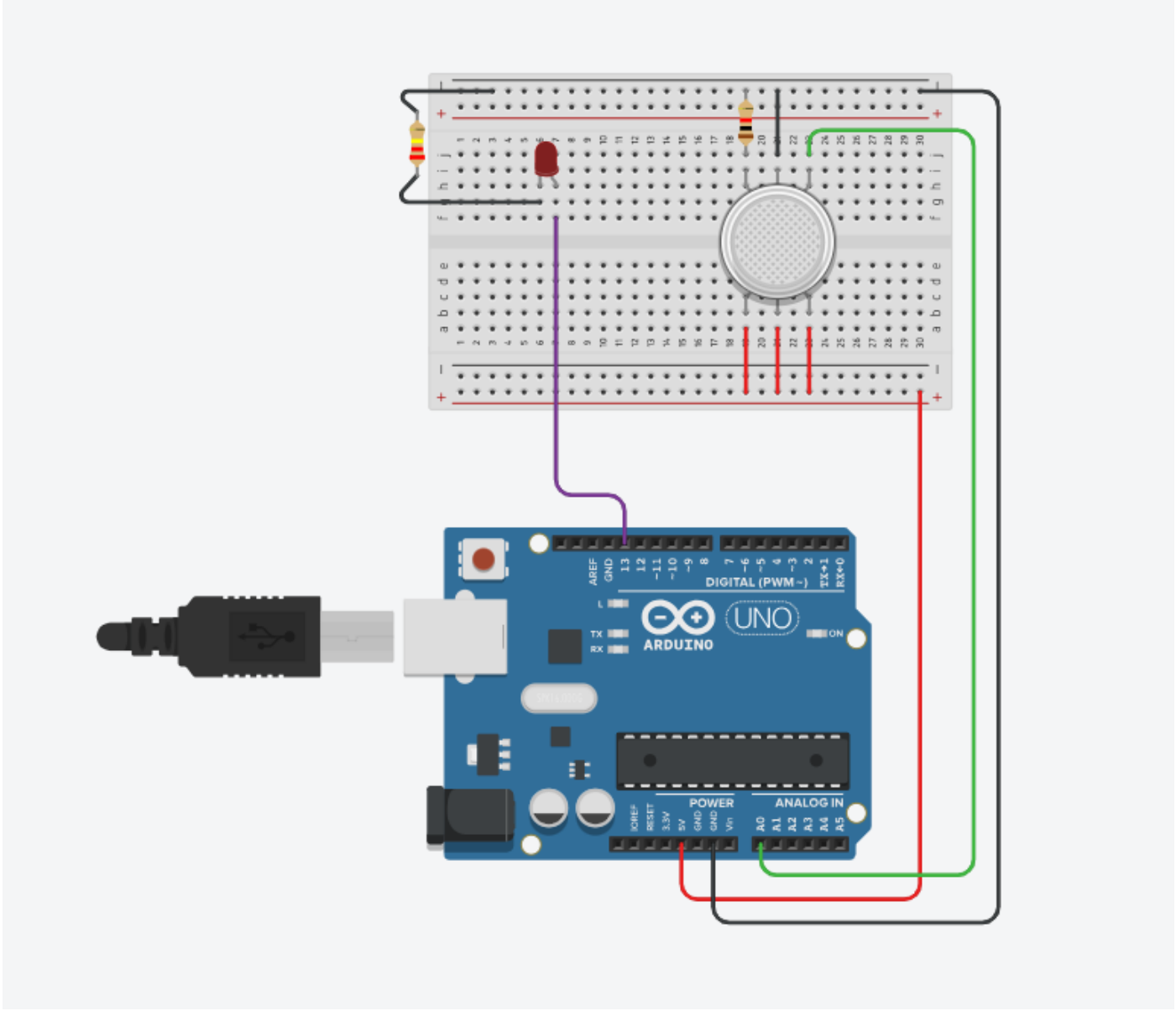
Step 1: Connect the Gas Sensor

1. Place the Gas Sensor on the breadboard.
2. Connect the VCC pin of the gas sensor to the 5V pin on the Arduino.
3. Connect the GND pin of the gas sensor to the GND pin on the Arduino.
4. Connect the Analog output (A0) of the gas sensor to Arduino's Analog pin A0.
5. Connect a 1kΩ resistor between the A0 output pin and GND (this acts as a pull-down resistor for stable readings).

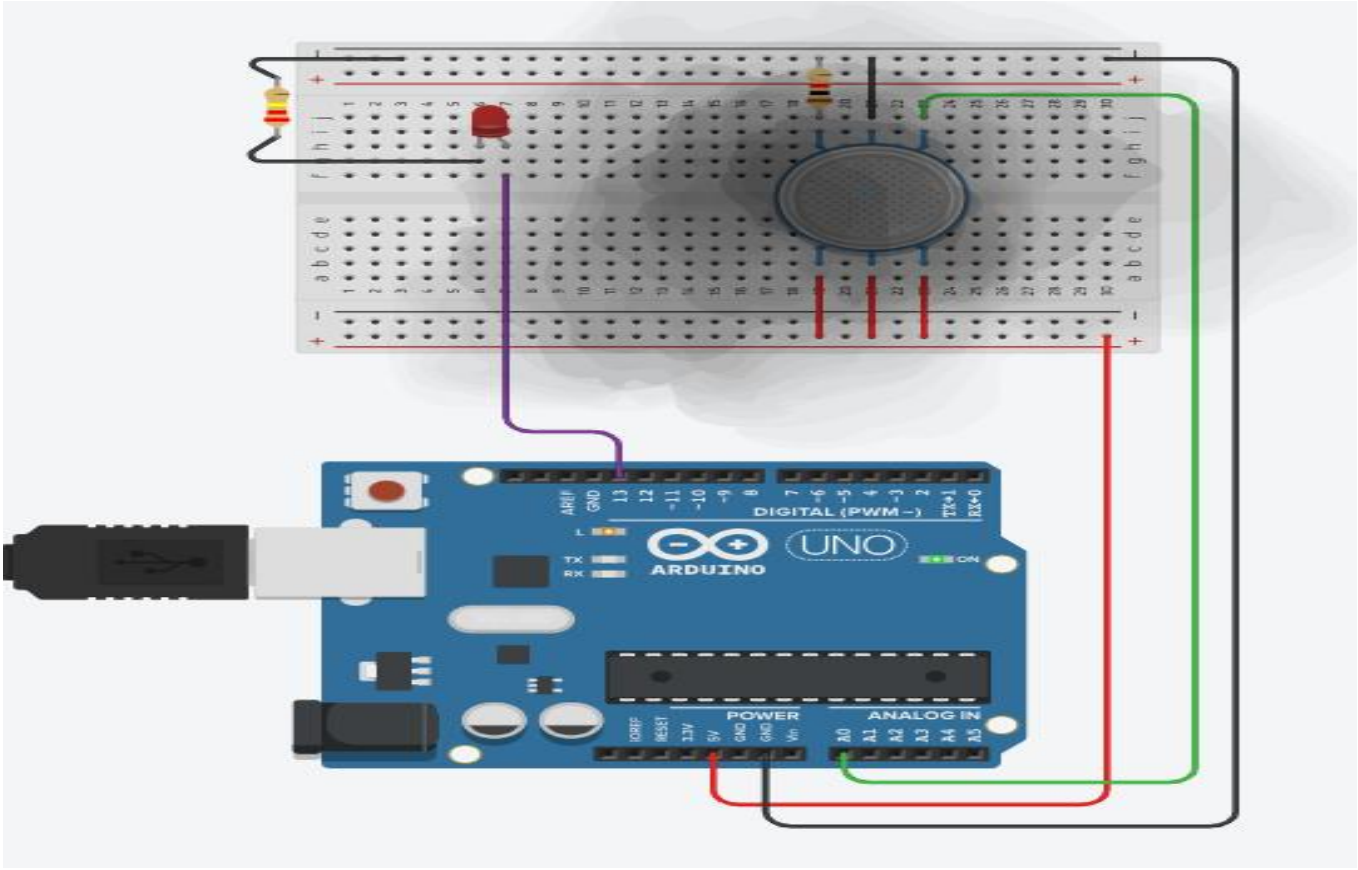
Step 2: Connect the LED

6. Place the LED on the breadboard (longer leg is anode (+), shorter leg is cathode (-)).
7. Connect the anode (long leg) of the LED to one end of the 220Ω resistor.
8. Connect the other end of the 220Ω resistor to Arduino digital pin 13.
9. Connect the cathode (short leg) of the LED to GND on the Arduino.

DIAGRAM



1. **In this diagram the gas is detected and the LED is turn ON.**



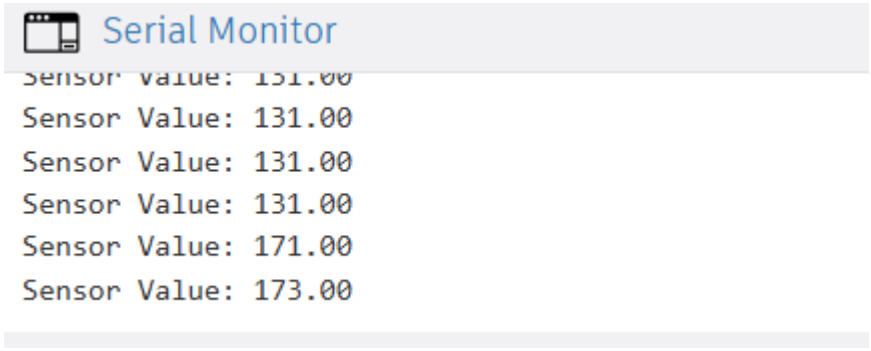
OUTPUT

2. **And the data is shown on the Serial monitor.**

Serial Monitor

```
328.00 | GAS DETECTED
328.00 | GAS DETECTED
328.00 | GAS DETECTED
328.00 | GAS DETECTED
328.00 | GAS DETECTED
328.00 | GAS DETECTED
```

3. **If the gas is little bit far from the gas sensor, then we can see the sensor value.**



CODE

```
int LED = 13;
int MQ2pin = A0;

void setup() {
  Serial.begin(9600);
  pinMode(LED, OUTPUT);
}

void loop() {
  float sensorValue;
  sensorValue = analogRead(MQ2pin);

  if (sensorValue >= 250) {
    digitalWrite(LED, HIGH);
    Serial.print(sensorValue);
    Serial.println(" | GAS DETECTED");
  } else {
    digitalWrite(LED, LOW);
    Serial.print("Sensor Value: ");
    Serial.println(sensorValue);
  }

  delay(1000);
}
```