Practical 1

Aim : Introduction to Arduino

a. Introduction to Arduino circuits and breadboarding

Practical 1: Introduction to Arduino

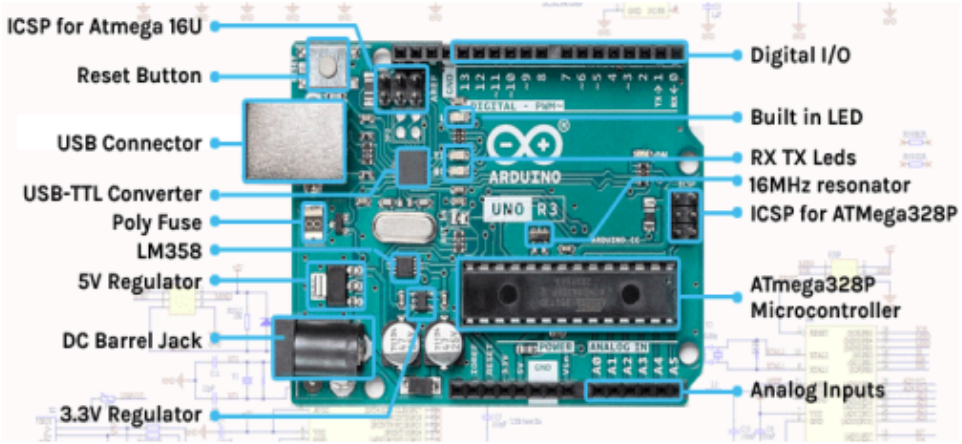
Aim:

1. To study the basics of Arduino circuits and bread-boarding
2. Blinking of LEDs

Simulation Environment: TinkerCAD (Free online simulator)

Part A: Basics of Arduino Circuits

Theory:



Arduino is an open-source electronics platform that has gained immense popularity for its ease of use and versatility. It was created in 2005 by a group of Italian engineers and is now maintained and developed by the Arduino community.

The heart of the Arduino platform is a microcontroller, which is a small, programmable computer on a single integrated circuit (IC) chip.

Arduino boards, which house these microcontrollers, provide a user-friendly environment for creating interactive electronic projects, prototypes, and various applications.

Key Components of Arduino:

1. Microcontroller: The core of an Arduino board is the microcontroller. The most commonly used microcontroller in Arduino is the ATmega series from Atmel (now a part of Microchip Technology). These microcontrollers come in different variations and are the brains behind your Arduino projects.

2. Input/output Pins: Arduino boards have a set of digital and analog pins that can be used to read data (inputs) or send data (outputs). Digital pins work with binary signals (0 or 1), while analog pins can read a range of values. The number and types of pins vary among different Arduino board models.

3. Power Supply: Arduino boards can be powered via USB, an external power supply, or a battery. Some boards have built-in voltage regulators, which make them compatible with a range of power sources.

4. USB Port: Arduino boards often feature a USB port for programming and power supply. This allows you to connect the board to your computer and upload code.

5. Reset Button: A reset button is provided to restart the Arduino, allowing you to upload new code or reset the program.

6. LED Indicator: Many Arduino boards include a built-in LED (Light Emitting Diode) on pin 13, which can be used for testing and basic visual feedback.

Arduino Software:

The Arduino platform comes with its integrated development environment (IDE). The Arduino IDE is a software tool that allows you to write, compile, and upload code to the Arduino board. Key features of the IDE include:

- Programming Language: Arduino uses a simplified version of the C/C++ programming language. It provides a set of libraries and functions tailored for easy interaction with the hardware.

- Code Library: Arduino has a vast library of pre-written code and functions that simplify common tasks, making it accessible to beginners.

- Serial Monitor: The IDE includes a serial monitor that allows you to communicate with the Arduino board and view debugging information.

- Community Support: The Arduino community is large and active, offering forums, tutorials, and extensive documentation to help users troubleshoot issues and learn.

b. Blinking of LEDs

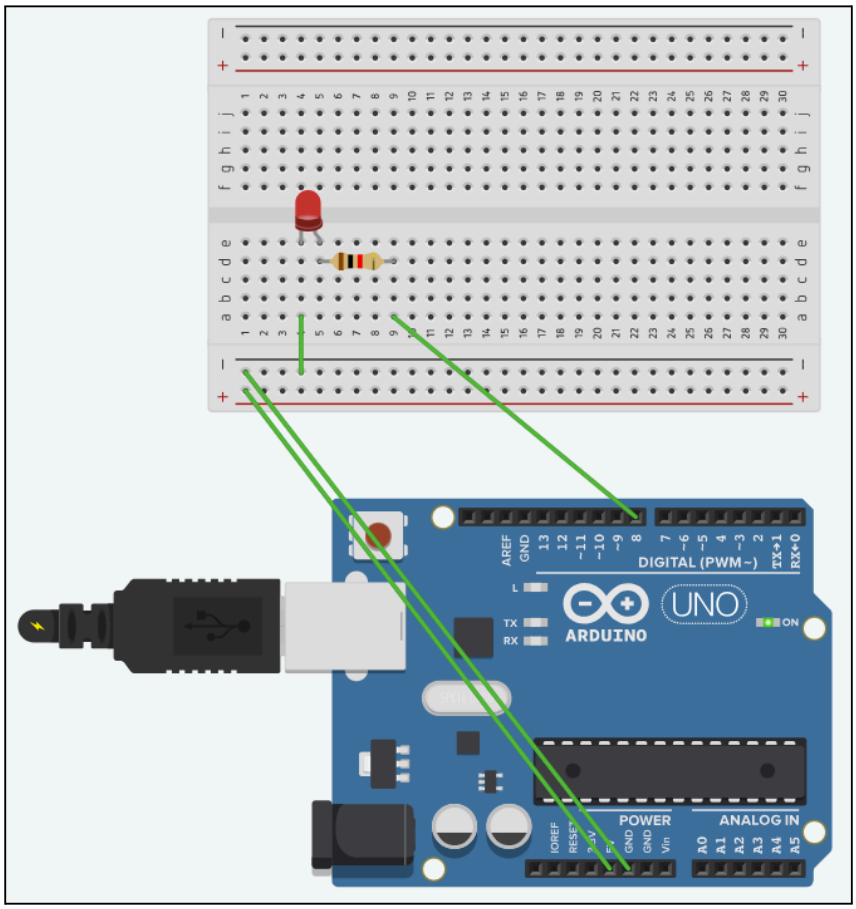
Component :

Name	Quantity	Component
U1	1	Arduino Uno R3
D1	1	Red LED
R1	1	220 Ω Resistor

Code:

```
int ledPin = 8;

void setup()
{
  pinMode(ledPin, OUTPUT);
}

void loop()
{
  digitalWrite(ledPin, HIGH); // On
  delay(1000);                          // 1000 secs delay
  digitalWrite(ledPin, LOW);  // Off
  delay(1000);                          // 1000 secs delay
}
```

Output:



Practical 2

Aim : Program using Light Sensitive Sensors

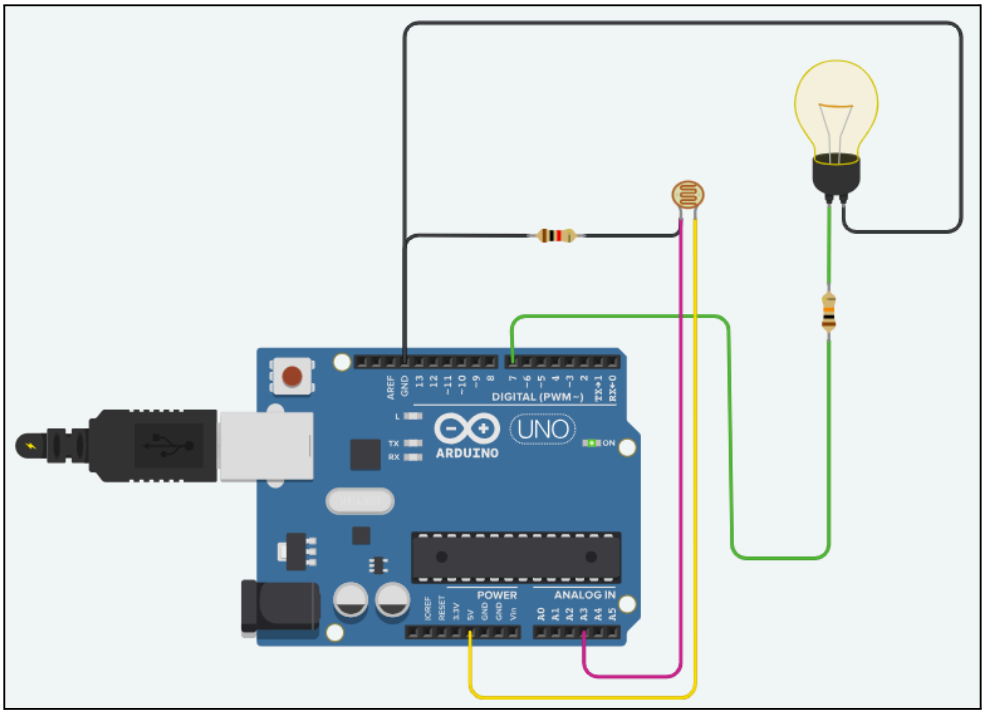
Component:

Name	Quantity	Component
L1	1	Light bulb
U1	1	Arduino Uno R3
R1	1	1 kΩ Resistor
R3	1	10 kΩ Resistor
R2	1	Photoresistor

Code:

```
int bulb = 7;
int ldr = A3;

void setup()
{
  pinMode(bulb, OUTPUT);
  pinMode(ldr, INPUT);
}

void loop()
{
  if(analogRead(ldr) > 500)
    digitalWrite(bulb,0);
  else
    digitalWrite(bulb,1);
}
```

Output:



Practical 3

Aim : Program using temperature sensors

Component:

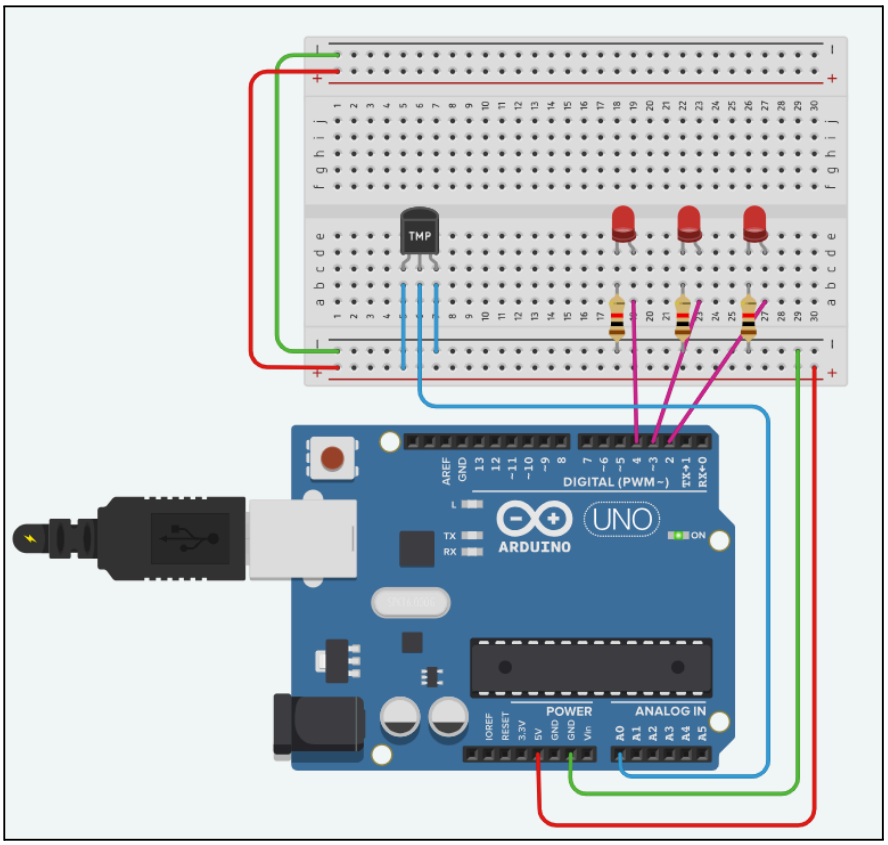
Name	Quantity	Component
U1	1	Arduino Uno R3
U2	1	Temperature Sensor [TMP36]
R1 R2 R3	3	1 kΩ Resistor
D1	1	Red LED
D2	1	Blue LED
D3	1	Green LED

Code:

```
int baselineTemp = 0;
int celsius = 0;
int fahrenheit = 0;

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(5000);

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}

void loop()
{
  baselineTemp = 40;
  celsius = map(((analogRead(A0) - 20) * 3.04), 0, 1023, -40, 125);
  fahrenheit = ((celsius * 9) / 5 + 32);
  Serial.print(celsius);
  Serial.print("C");
  Serial.print(fahrenheit);
```

```
  Serial.print("F");

  if (celsius < baselineTemp)
  {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp && celsius < baselineTemp+10)
  {
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 10 && celsius < baselineTemp+20)
  {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
  }

  if (celsius >= baselineTemp + 20 && celsius < baselineTemp+30)
  {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  if (celsius >= baselineTemp + 30)
  {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  delay(1000);
}
```

Output:



Practical 6

Aim :Programs using Ultrasonic Sensors

Component:

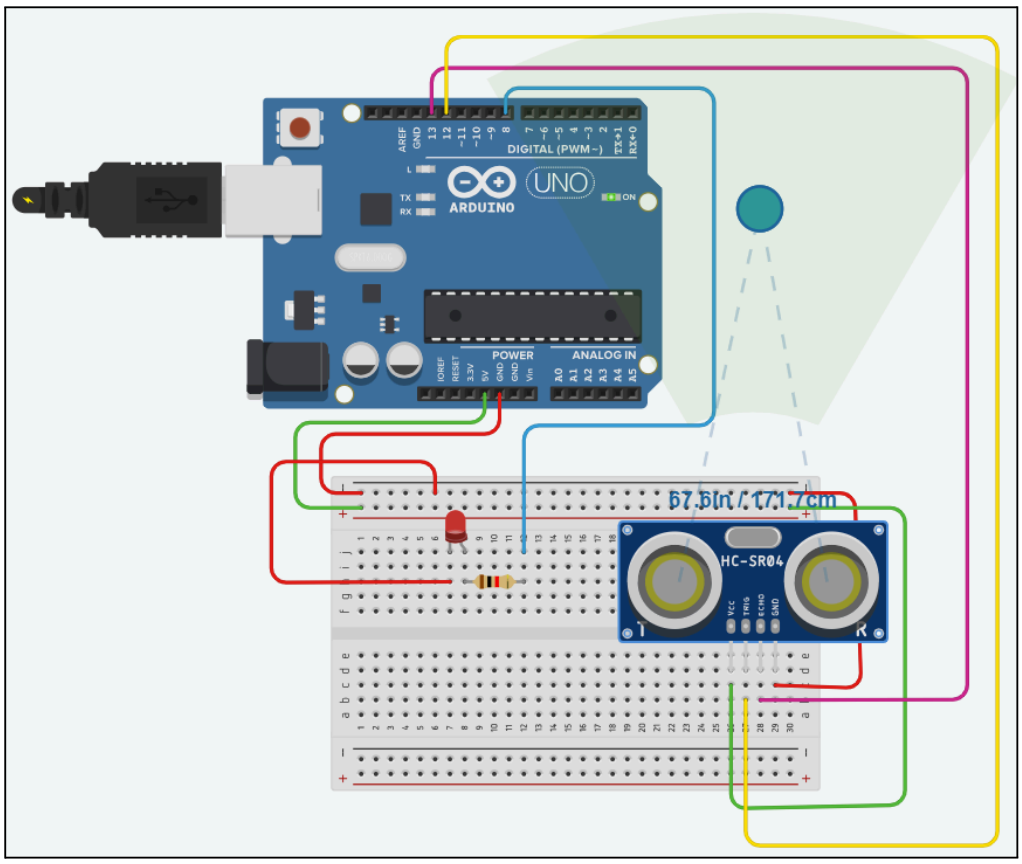
Name	Quantity	Component
U1	1	Arduino Uno R3
DIST1	1	Ultrasonic Distance Sensor
R1	1	1 kΩ Resistor
D1	1	Red LED

Code:

```
int trigger = 12;
int echo = 13;
int led = 8;

long duration = 0;
int cm = 0;
int inch = 0;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(trigger, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(led, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(trigger, LOW);
  digitalWrite(trigger, HIGH);
  digitalWrite(trigger, LOW);

  duration = pulseIn(echo, HIGH);
  cm = duration*0.034/2;
  inch = duration*0.0133/2;

  if (inch < 50){
    digitalWrite(led, HIGH);
  } else {
```

```
    digitalWrite(led, LOW);
   }

   if (inch < 100 ) {
    Serial.print("Inches: ");
    Serial.println(inch);
    Serial.print("Cm: ");
    Serial.println(cm);
   }
   delay(5000);
}
```

Output:


DIGITAL (PWM~)
UNO
ARDUINO
POWER
ANALOG IN
HC-SR04
Vcc
Trig
Echo
GND

Practical 4

Aim : Program using Humidity Sensors.

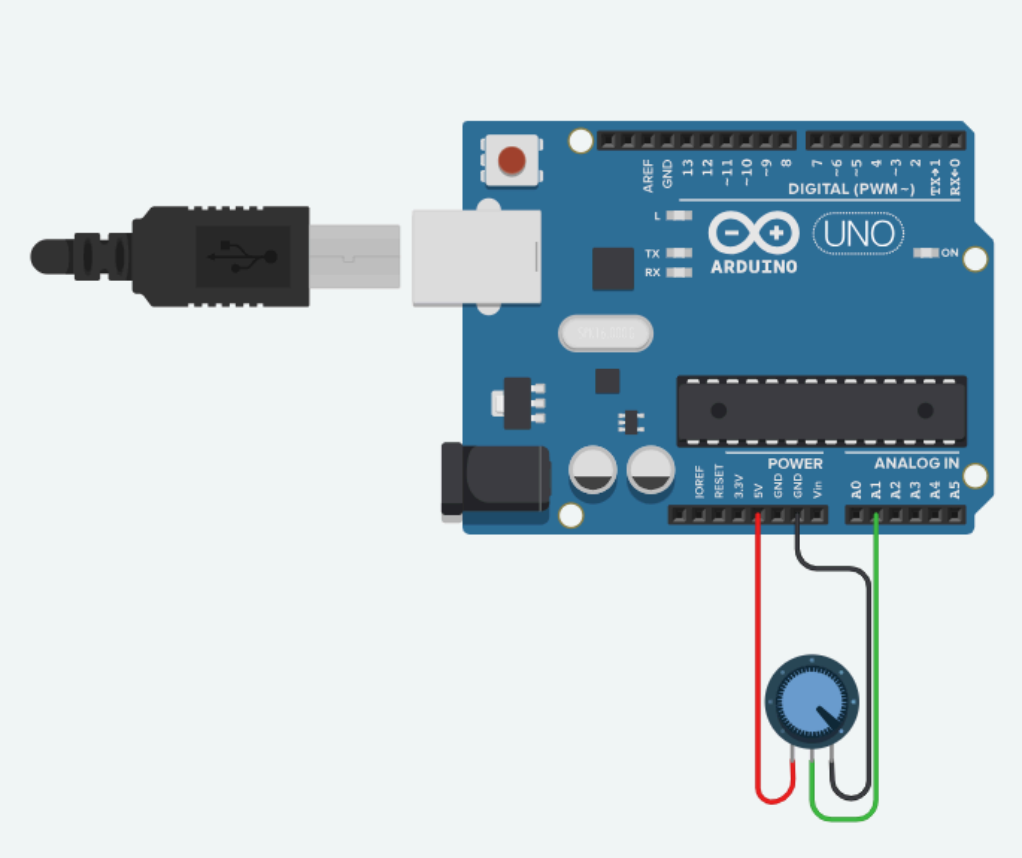
Component:

Name	Quantity	Component
U1	1	Arduino Uno R3
Rpot1	1	250 kΩ Potentiometer

Code:

```
// C++ code
//
const int analogIn=A1;
int humiditySensorOutput=0;
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  humiditySensorOutput=analogRead(analogIn);
  int humidityPercentage = map(humiditySensorOutput, 0, 1023, 10, 70);
  Serial.print("\nhumidity:");
  Serial.print(humiditySensorOutput);
  Serial.print("%");
  delay(5000);
}
```

Output :



Serial Monitor

```
Humidity:10%
Humidity:10%
Humidity:62%
Humidity:44%
Humidity:31%
Humidity:33%
```

Practical 9

Aim: Program using Servo motors

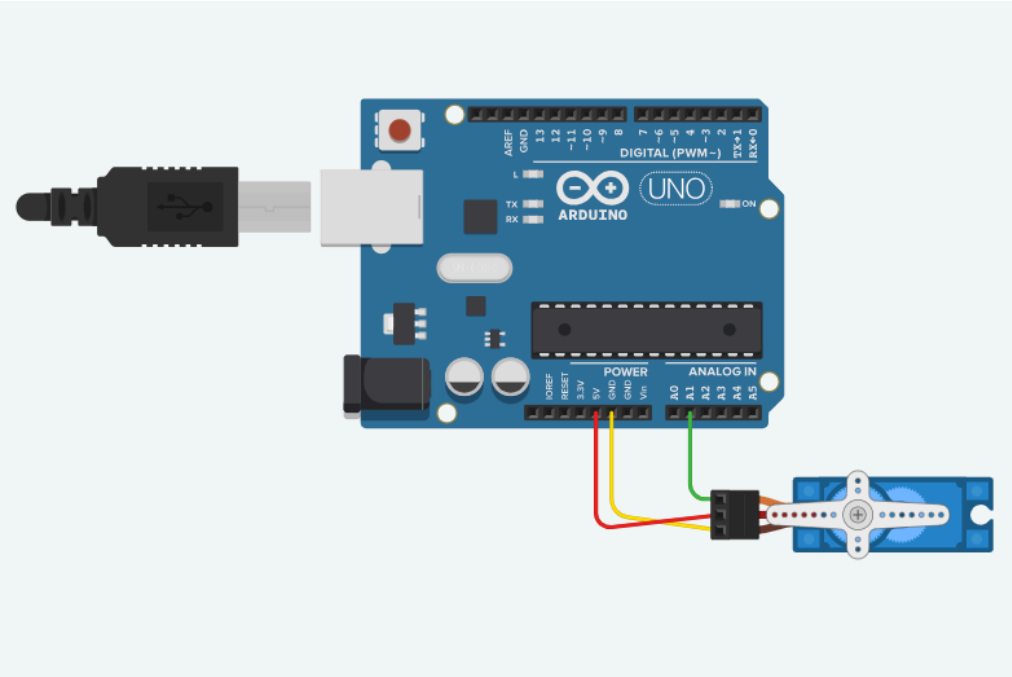
Component:

Name	Quantity	Component
U1	1	Arduino Uno R3
SERVO1	1	Positional Micro Servo

Code:

```
// C++ code
//
#include<Servo.h>
Servo servoBase;
void setup()
{
  servoBase.attach(A1);
  servoBase.write(0);
}

void loop()
{
  for(int i=0; i<=180; i+=10)
  {
    servoBase.write(i);
          delay(2000); // Wait for 1000 millisecond(s)
}
}
```

Output :



Practical 7

Aim : Programs using digital infrared motion sensors

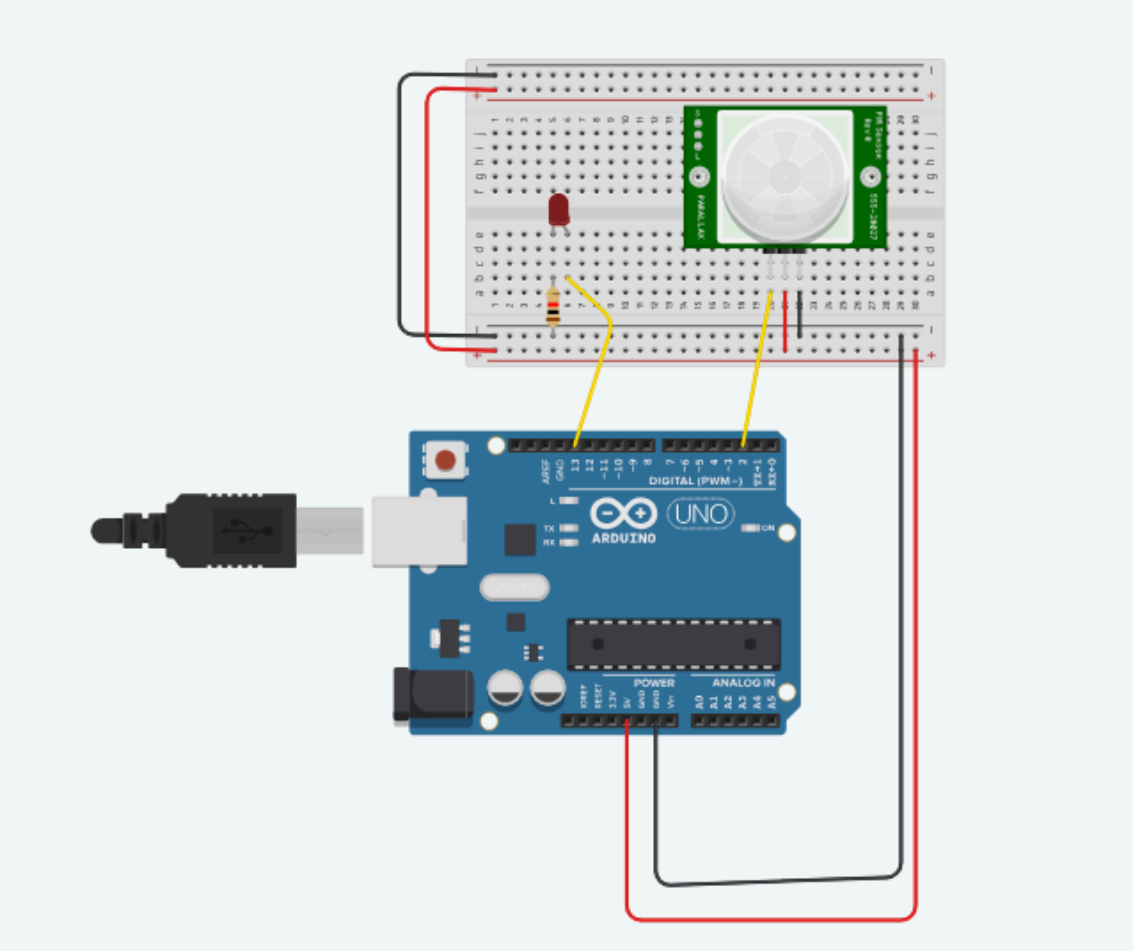
Component:

Name	Quantity	Component
U1	1	Arduino Uno R3
PIR1	1	-10.632802562664722 , -180.6458356715302 , -198.64995089786763 PIR Sensor
R1	1	1 kΩ Resistor
D1	1	Red LED

Code:

```
int sensorState= 0;
void setup()
{
  pinMode(2, INPUT);
  pinMode(LED_BUILTIN,OUTPUT);
}
void loop()
{
  sensorState = digitalRead(2);
  if(sensorState==HIGH)
{
    digitalWrite(LED_BUILTIN,HIGH);
  }
else
  {
    digitalWrite(LED_BUILTIN,LOW);
  }
delay(10);
}
```

Output :



Practical 8

Aim : . Programs using gas sensors

Component:

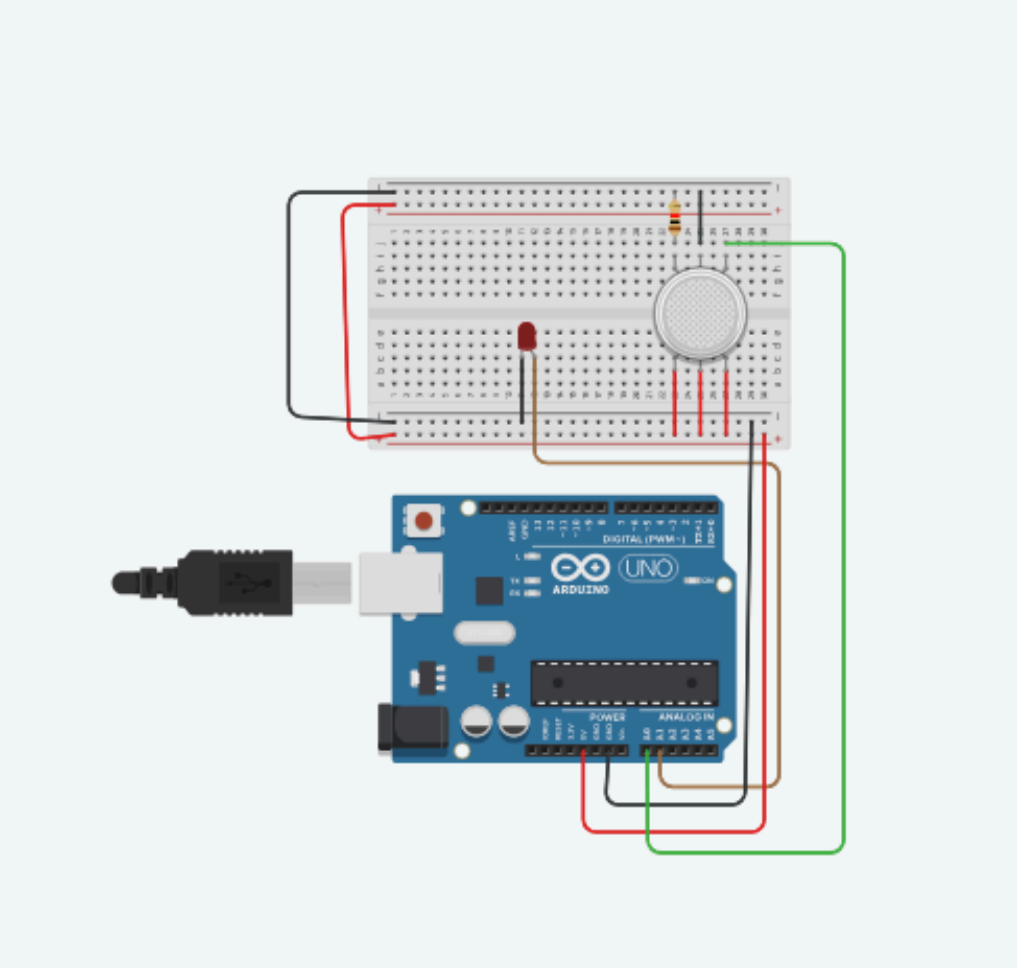
Name	Quantity	Component
U1	1	Arduino Uno R3
R1	1	1 kΩ Resistor
D1	1	Red LED
GAS1	1	Gas Sensor

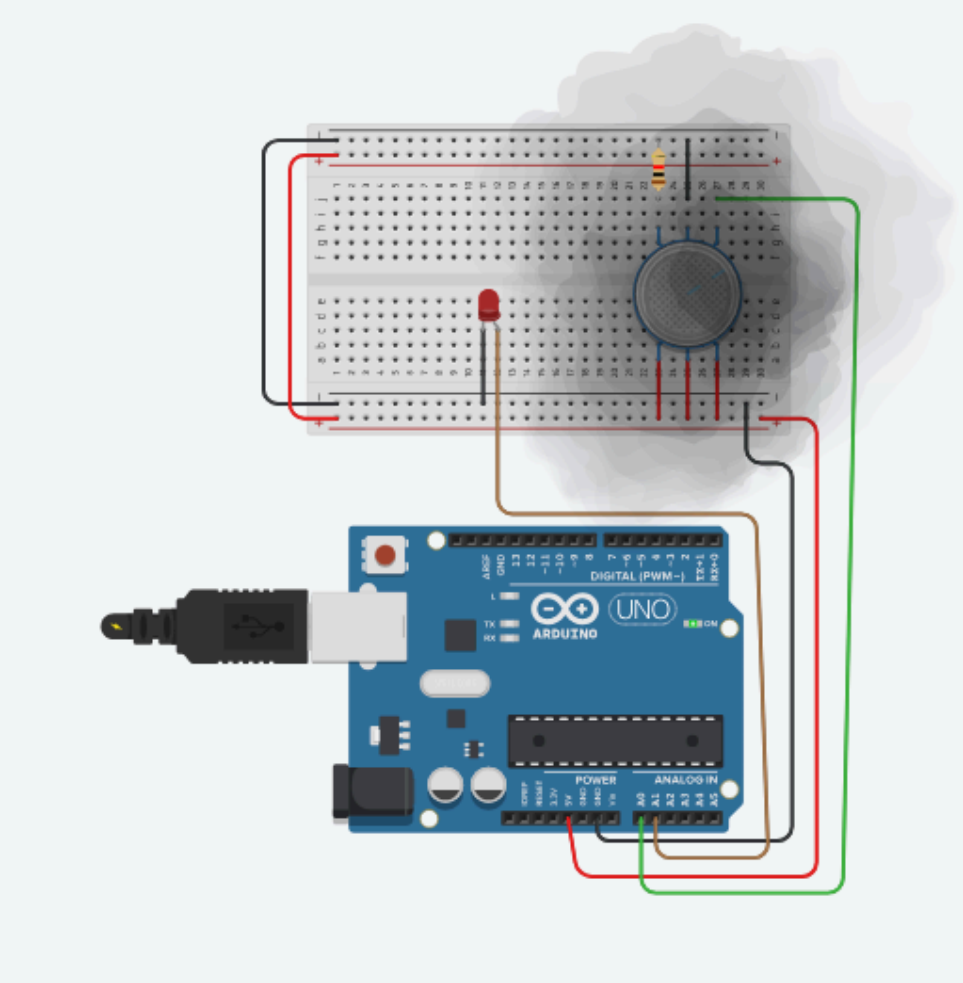
Code:

```
int LED =A1;
const int gas =0;
int MQ2pin = A0;
  void setup()
{
  Serial.begin(9600);
  }
void loop()
 {
float sensorValue,MQ2pin;
sensorValue =analogRead(MQ2pin);
  if(sensorValue<=170){
    digitalWrite(LED,LOW);
    Serial.print(sensorValue);
    Serial.println("SENSOR VALUE”);
```

```
  }
  else
{
    digitalWrite(LED,HIGH);
  Serial.println("SMOKE DETECTED);
  Serial.println(sensorValue);
  }
  delay(1000);
}
float getsensorValue(int pin)
{
  return(analogRead(pin));
}
```

Output :





Serial Monitor

```
Smoke Detected
304.00
Smoke Detected
304.00
Smoke Detected
285.00
143.00sensor value
85.00sensor value
```