PRESIDENCY UNIVERSITY

(Established under the Presidency University Act, 2013 of the Karnataka Act 41 of 2013)

Internet of Things

Lab Record

NAME: Sai Ram. K

ROLL NO: 20181CSE0621

SEC: 6-CSE-10

COURSE CODE: CSE 220

Contents

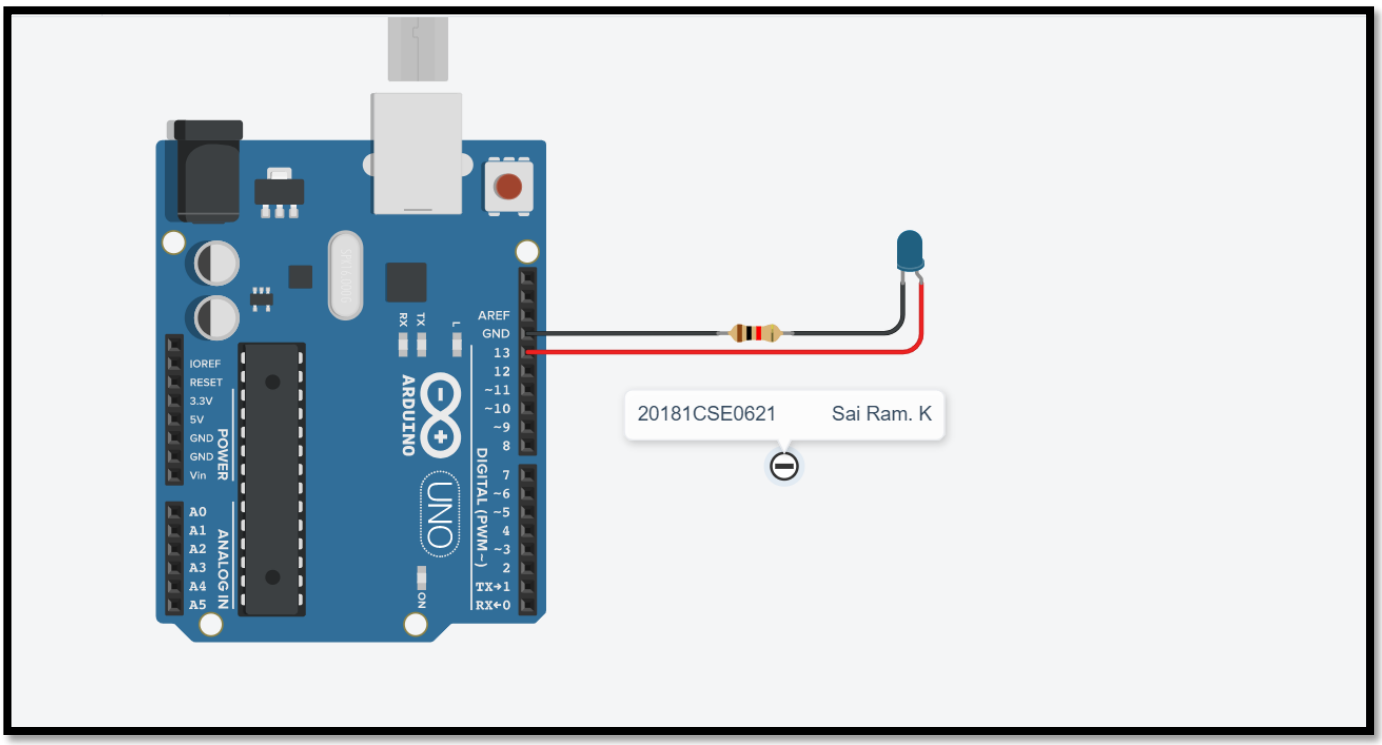
EXPERIMENT - 1

AIM: Write the Code to Blink an LED on ARDUINO UNO R3. Compile and verify the result on ARDUINO IDE.

Components Required:

Arduino, LED, Resistors, Breadboard

Initial Circuit Design:

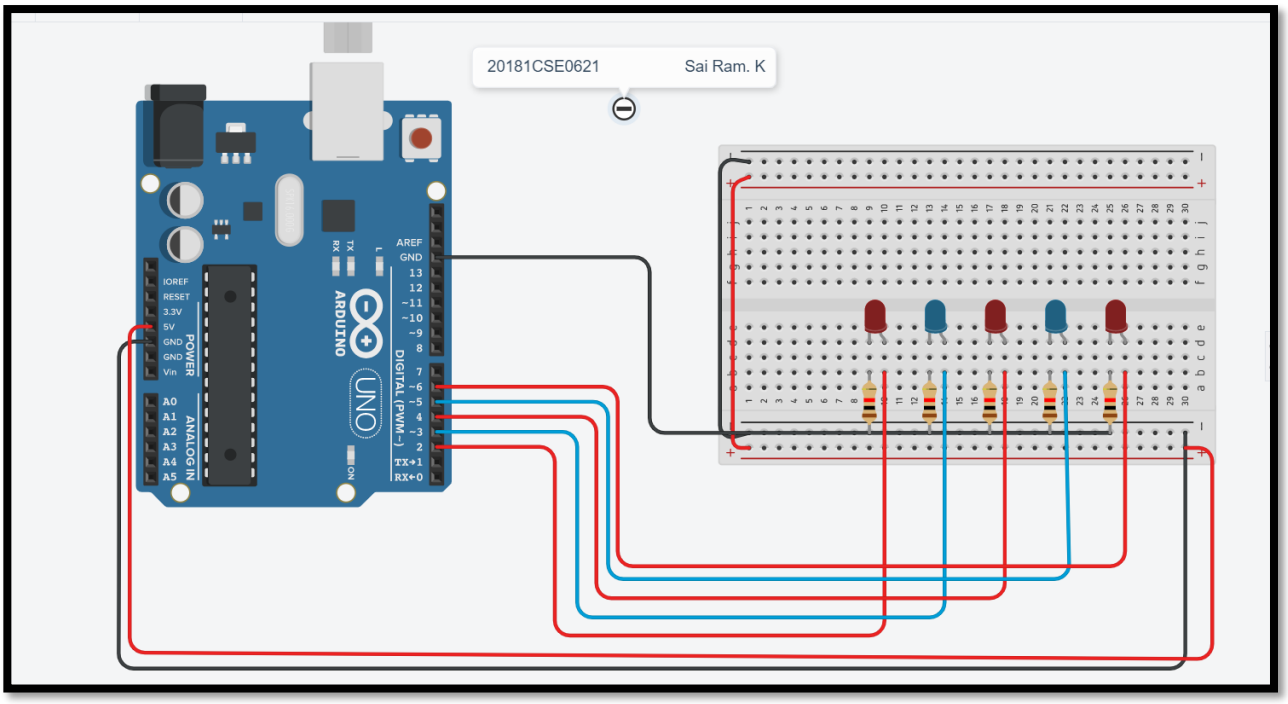


Arduino Sketch:

```
void setup()
{
  pinMode(13, OUTPUT);
}
void loop()
{
  digitalWrite(13, HIGH);
  delay(1000); // Wait for 1000
millisecond(s)
```

```
  digitalWrite(13, LOW);

  delay(1000); // Wait for 1000
millisecond(s)

}
```

Output Screenshots:



i)To blink two LED's alternatively

Aim: Blink LED's Alternately

Components: Arduino UNO

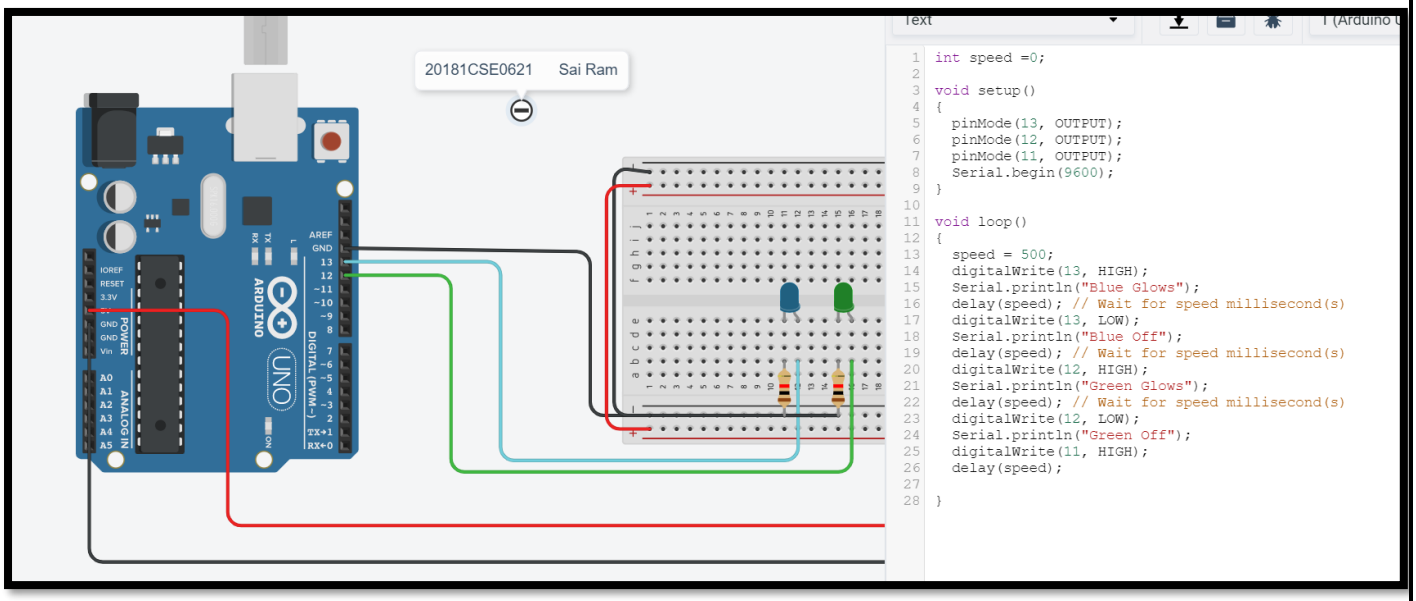
LED

Resistor

Tinker cad Simulator

Bread board

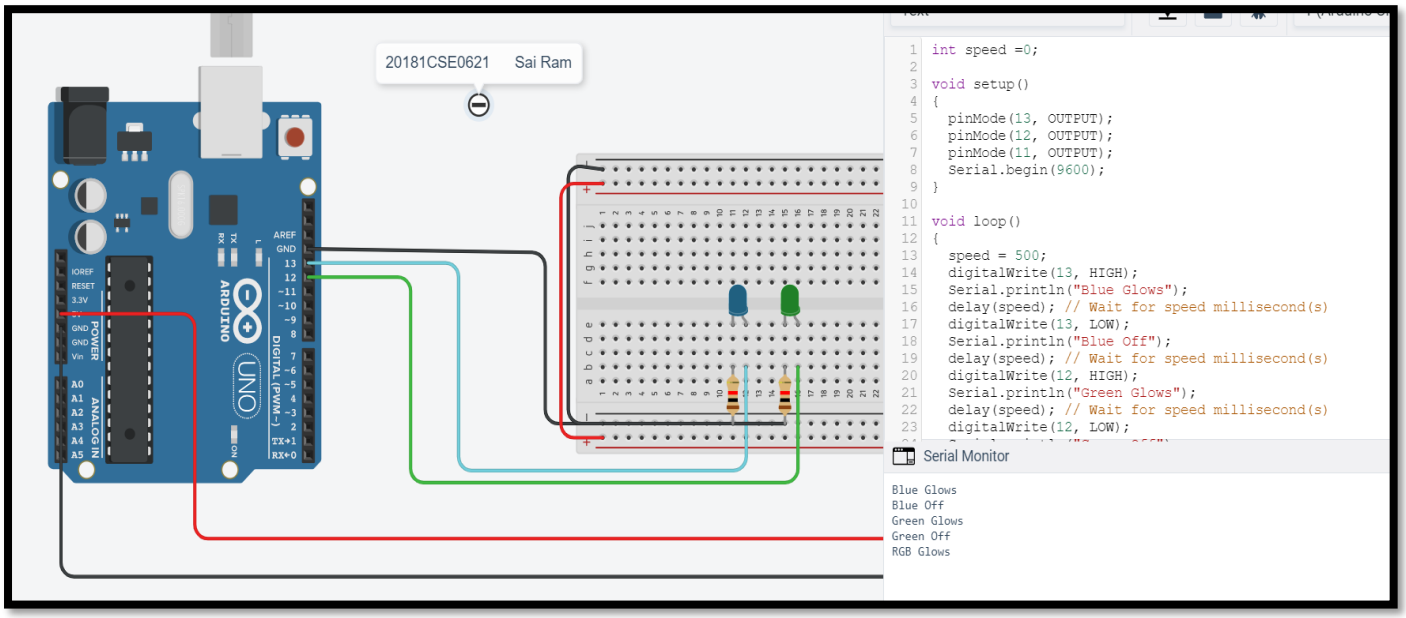
Circuit diagram:



Sketch:

```
void setup()
{
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  digitalWrite(13, HIGH);
  Serial.println("led1 is on");
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(12, HIGH);
  digitalWrite(13, LOW);
  Serial.println("led2 is on");
  Serial.println("led1 is off");
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(12, LOW);
  Serial.println("led2 is off");
}
```

Output Screenshot:

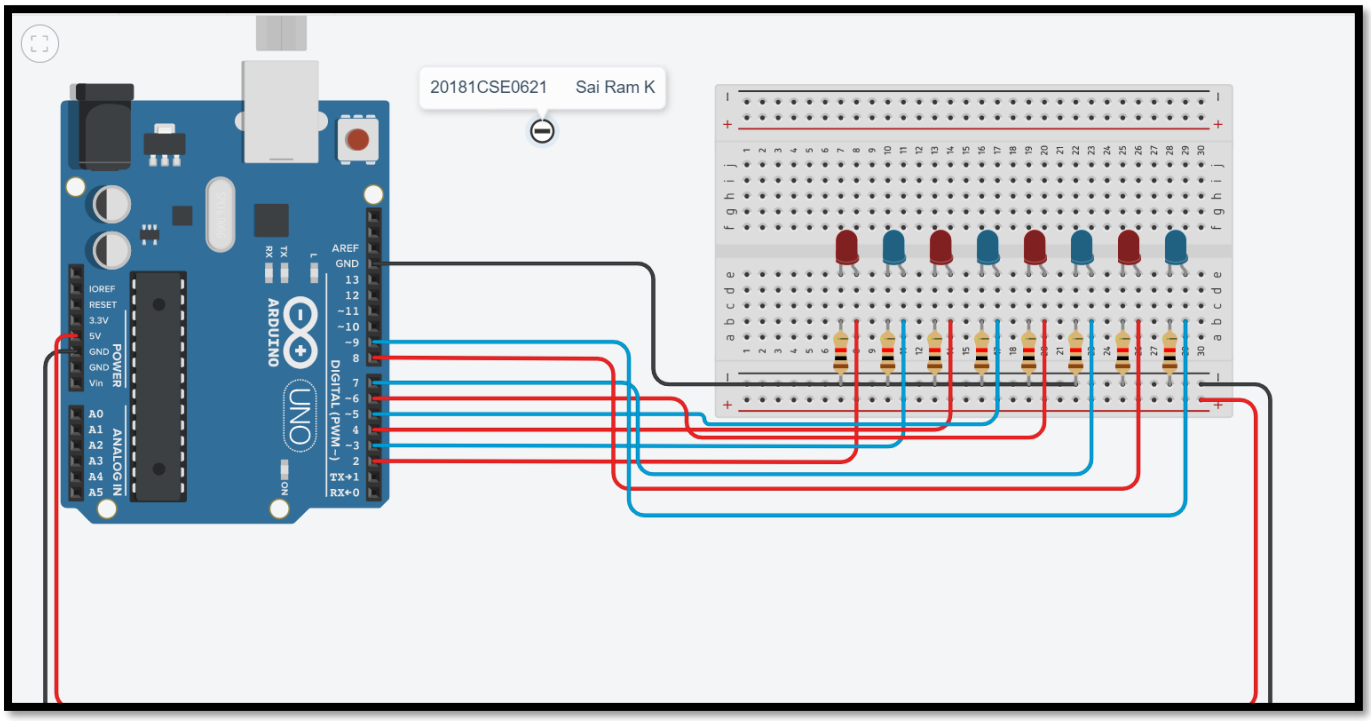


ii) To blink ODD and EVEN LED's

Aim: Blink led's ODD AND EVEN

Components: Arduino UNO
LED
Resistor
Tinker cad Simulator
Bread board

Circuit diagram:

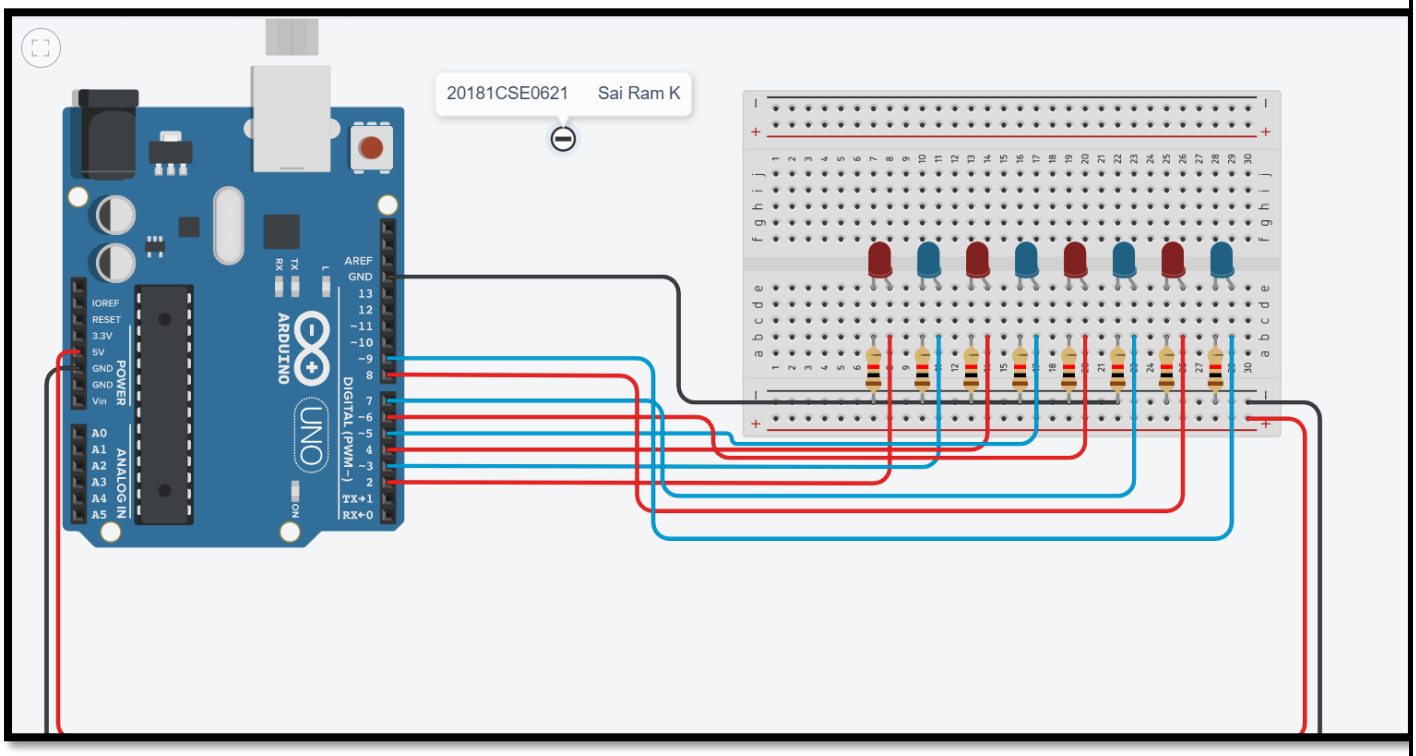


Sketch:

```
int ledPins[]={2,3,4,5,6,7,8,9};
int lightModulo=0;
void setup()
 {
  for(int i=0;i<8;i++)
   {
    pinMode(ledPins[i], OUTPUT);
   }
  Serial.begin(9600);
 }
void loop()
{
  int timer=1000;
  Serial.println(lightModulo);
  for(int i=0;i<8;i++)
  {
    if(i%2==lightModulo)
     {
      digitalWrite(ledPins[i],HIGH);
      }//end of if
    else
     {
      digitalWrite(ledPins[i],LOW);
      }//end of else
  }//end of for
  lightModulo--;
  lightModulo=abs(lightModulo);
```

```
 delay(timer);
}
```

Output Screenshots:

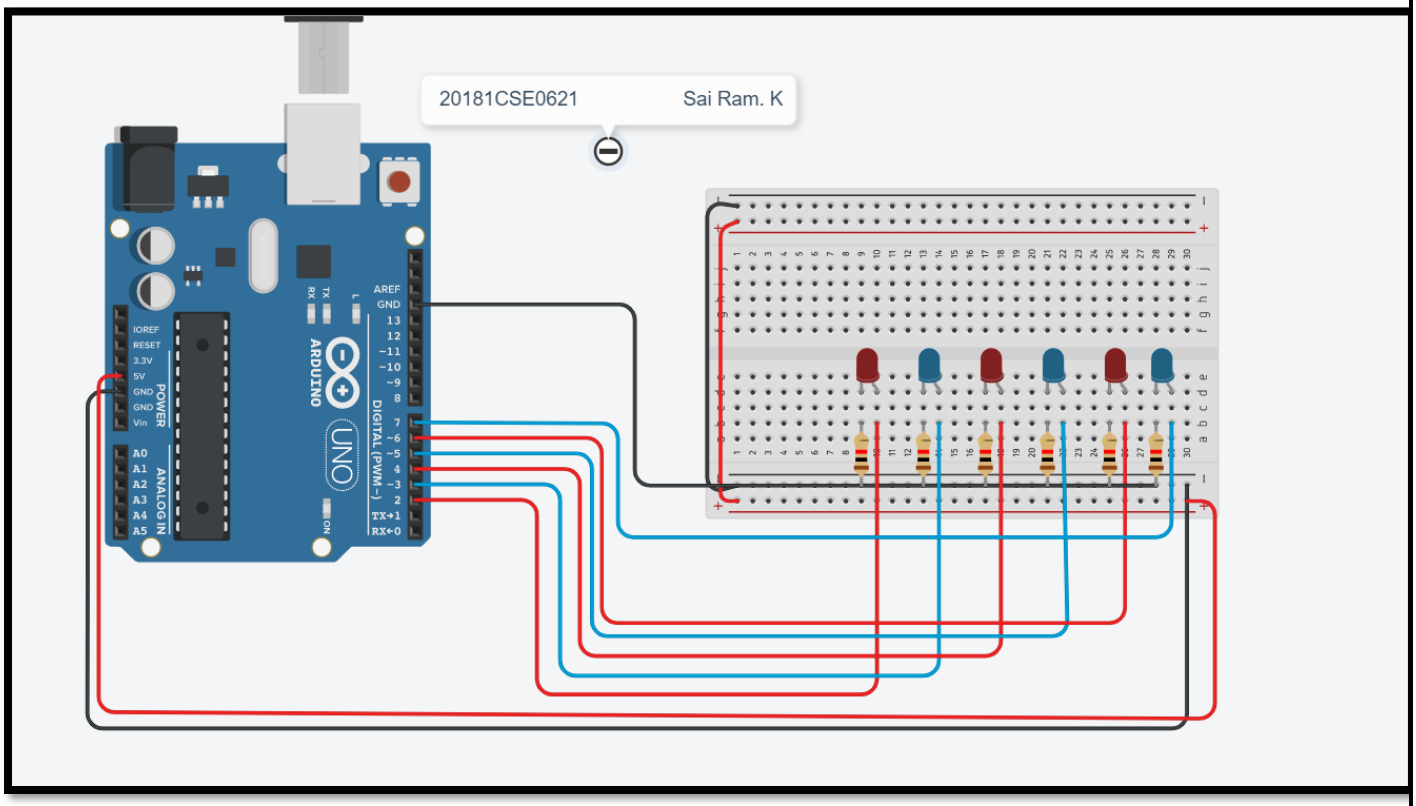


iii) Scroll LED's

Aim: To Scroll the LED's

COMPONENTS: Arduino UNO
LED
Resistor
Tinker cad Simulator
Bread board

Circuit diagram:

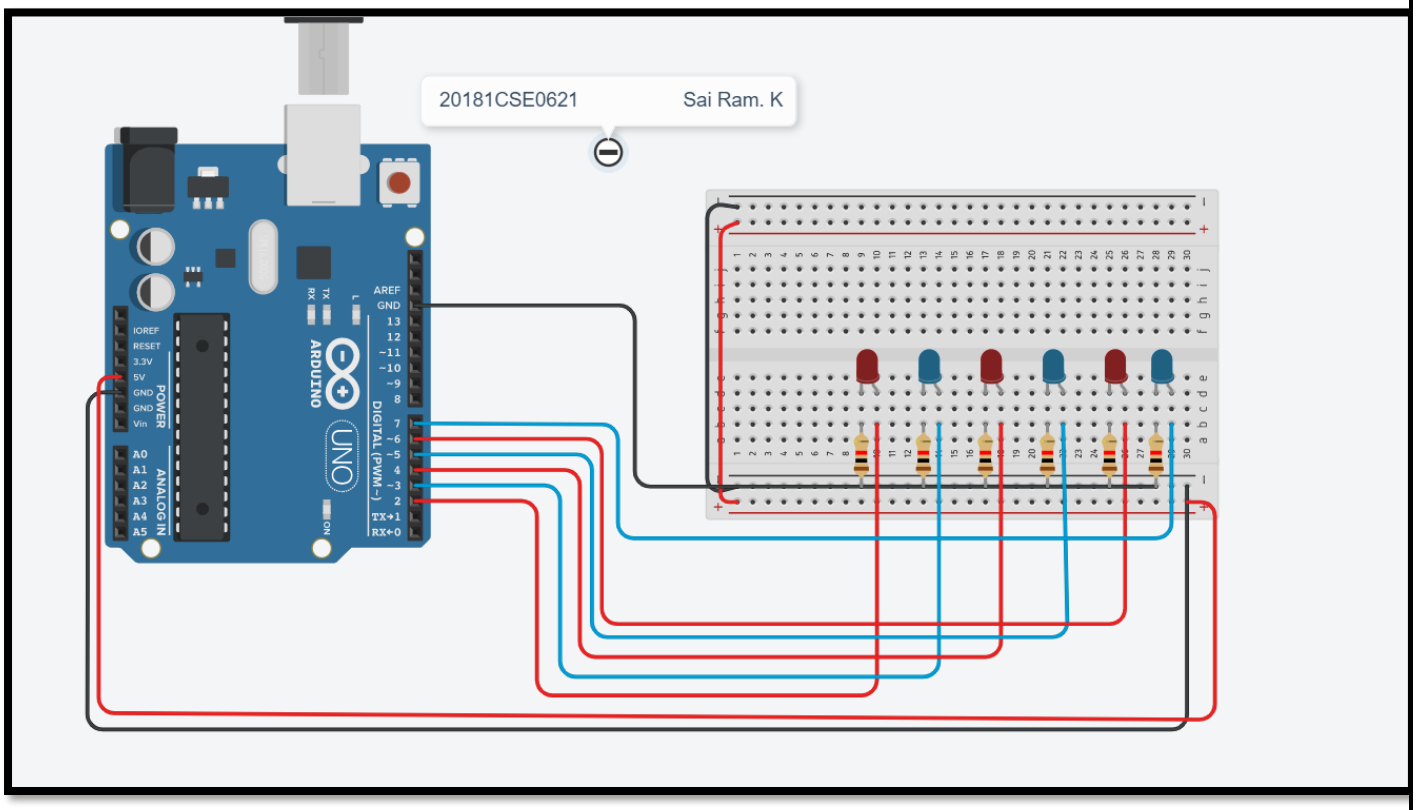


Sketch:

```
 int i;
void setup()
{

 pinMode(13, OUTPUT);
}

void loop()
{
 digitalWrite(8, HIGH);
 delay(1000); // Wait for 1000 millisecond(s)
 digitalWrite(8, LOW);
 for(i=9;i<=13;i++)
 {
   digitalWrite(i, HIGH);
```

```
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(i, LOW);
  }
  for(i=12;i>8;i--)
  {
   digitalWrite(i, HIGH);
   delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(i, LOW);
  }
}
```

Output screenshots:



Experiment – 2

Question : Interfacing of Arduino Uno with LED and switch. Write a program to control LED using Switch.
Additional Programs:
i)Single switch to control multiple LED's
ii)Multi switches to control multiple LED's

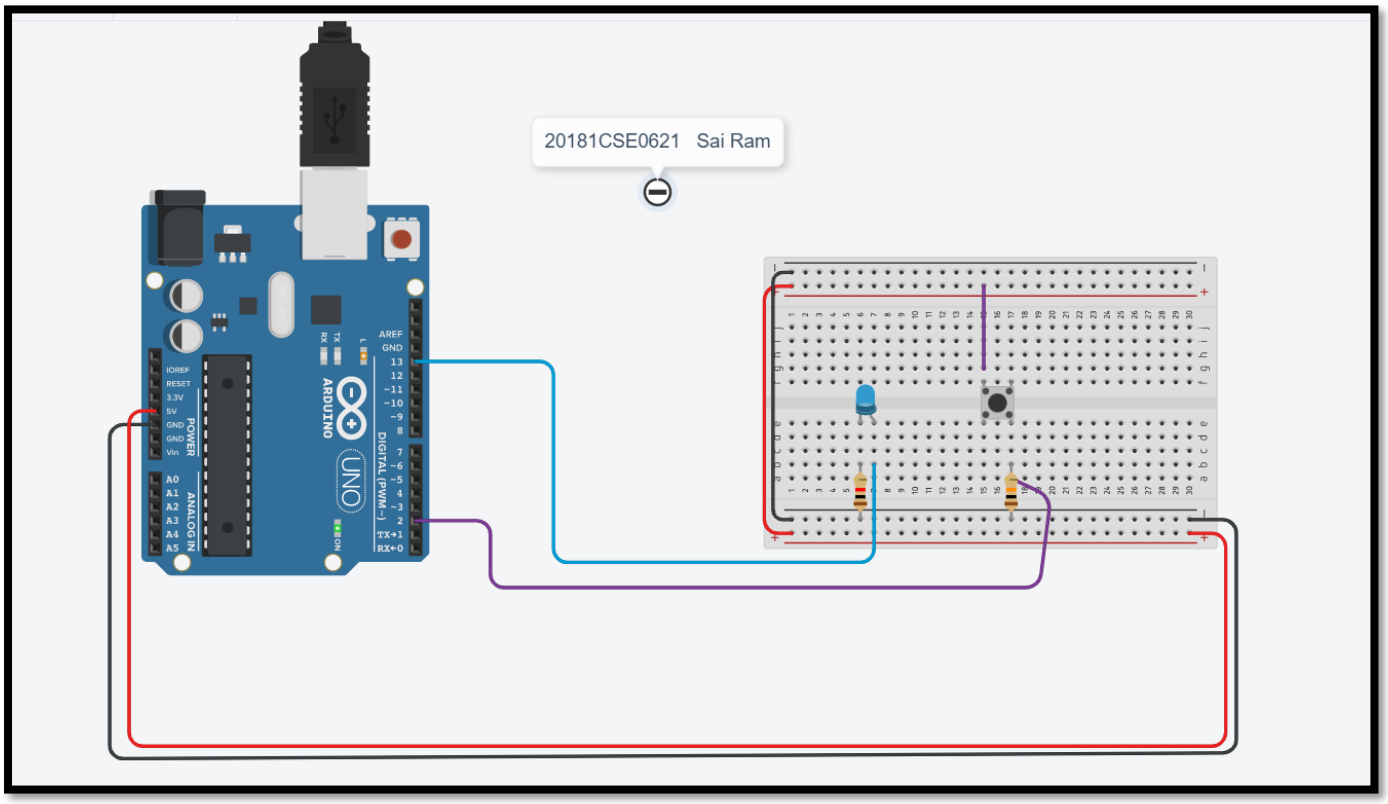
Aim : Connecting a single switch to control a single LED.

Components : Arduino, bread board, jumper wires, resistor, pushbutton.

Sketch [Code] :

```
int sbutton=0;

void setup()
{
        pinMode(13, OUTPUT);
        pinMode(2, INPUT);
        Serial.begin(9600);
}

void loop()
{
sbutton = digitalRead(2);
if (sbutton==HIGH)
{
        digitalWrite(13, HIGH);
        Serial.println('On');
}
else {
        digitalWrite(13, LOW);
        Serial.println('Off');
}
delay(10);
}
```

Output Screenshots :



■ Single switch to control multiple LED's :-

Aim : To connect a single switch to control multiple LEDs

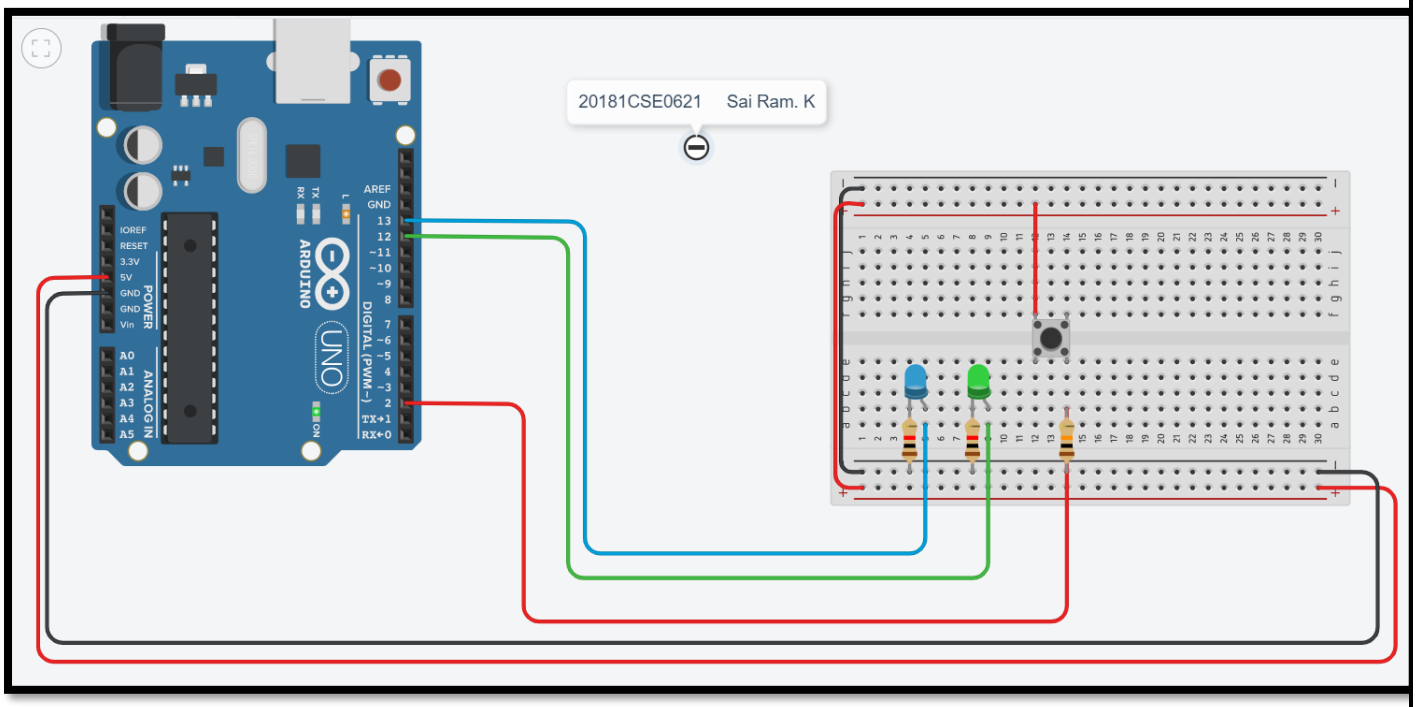
Components : Arduino, bread board, jumper wires, resistor, pushbutton.

Sketch [Code] :

```
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(2, INPUT);
 Serial.begin(9600);
}
```

```
void loop()

{

 if(digitalRead(2)==HIGH){

   digitalWrite(13,HIGH);

         digitalWrite(12,HIGH);

         delay(800);

 }

 else{

   digitalWrite(13,LOW); }

         digitalWrite(12,LOW);


}
```

Output Screenshots :



■ Multiple switches to control multiple LED's :-

Aim : To connect multiple switches to control multiple LEDs

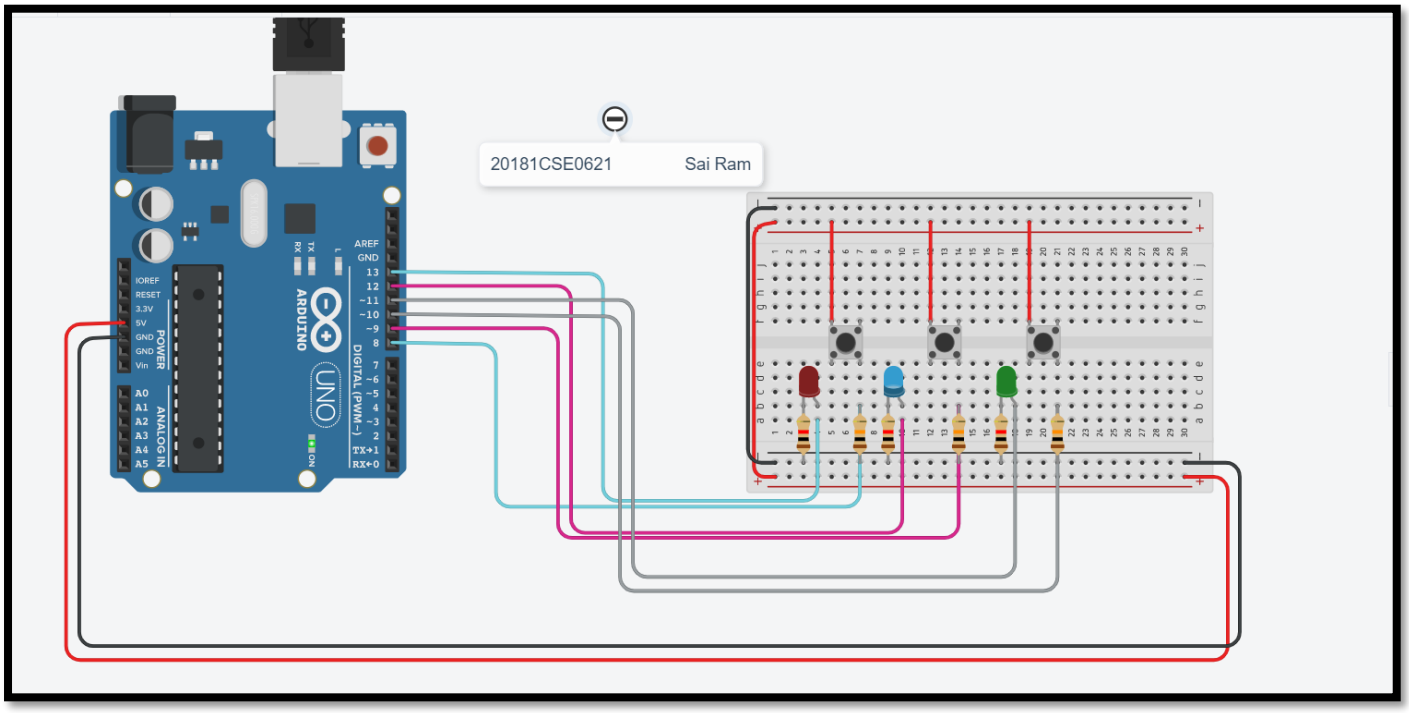
Components : Arduino, bread board, jumper wires, resistor, pushbutton.

Sketch [Code] :

```
int b1=0,b2=0,b3=0;
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(10, INPUT);
  pinMode(9, INPUT);
  pinMode(8, INPUT);
}

void loop()
{
  b1 = digitalRead(8);
  b2 = digitalRead(9);
  b3 = digitalRead(10);
if (b1==HIGH){
    digitalWrite(13, HIGH);     }
else {
  digitalWrite(13, LOW);    }
```

```
if (b2==HIGH){
   digitalWrite(12, HIGH);    }
else {
  digitalWrite(12, LOW);   }

if (b3==HIGH){
   digitalWrite(11, HIGH);    }
else {
  digitalWrite(11, LOW);   }
}
```

Output Screenshots :



Experiment – 3
Potentiometer

Question : Interfacing of Arduino Uno with potentiometer and LED. Write a program to vary the intensity of LED using a potentiometer.

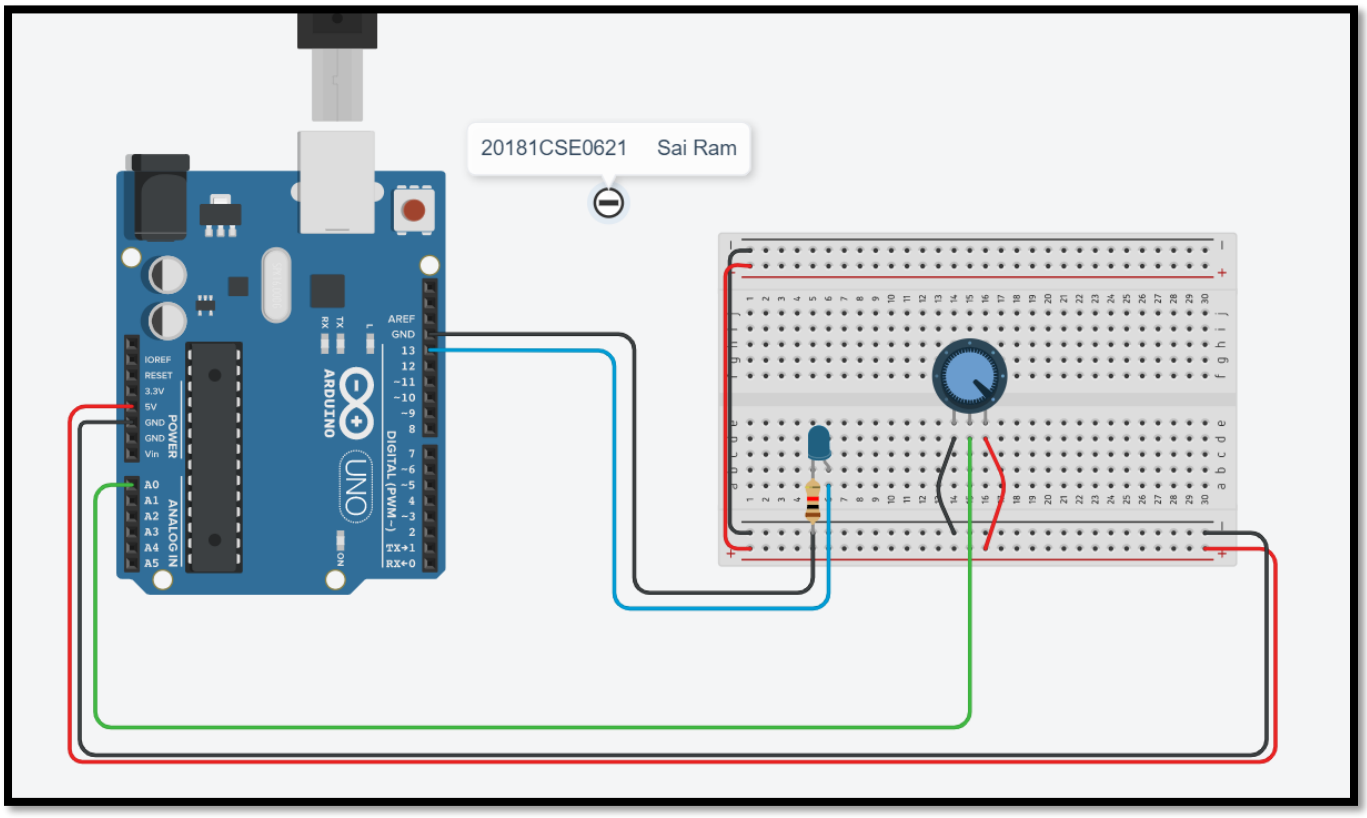
Additional Programs:

i)Adjust the brightness of LED without potentiometer.

Aim : Intensity of Led using Potentiometer ..

Components : Arduino UNO, Led, Potentiometer, Resistor, Tinckercad simulator .

Initial Circuit Design :

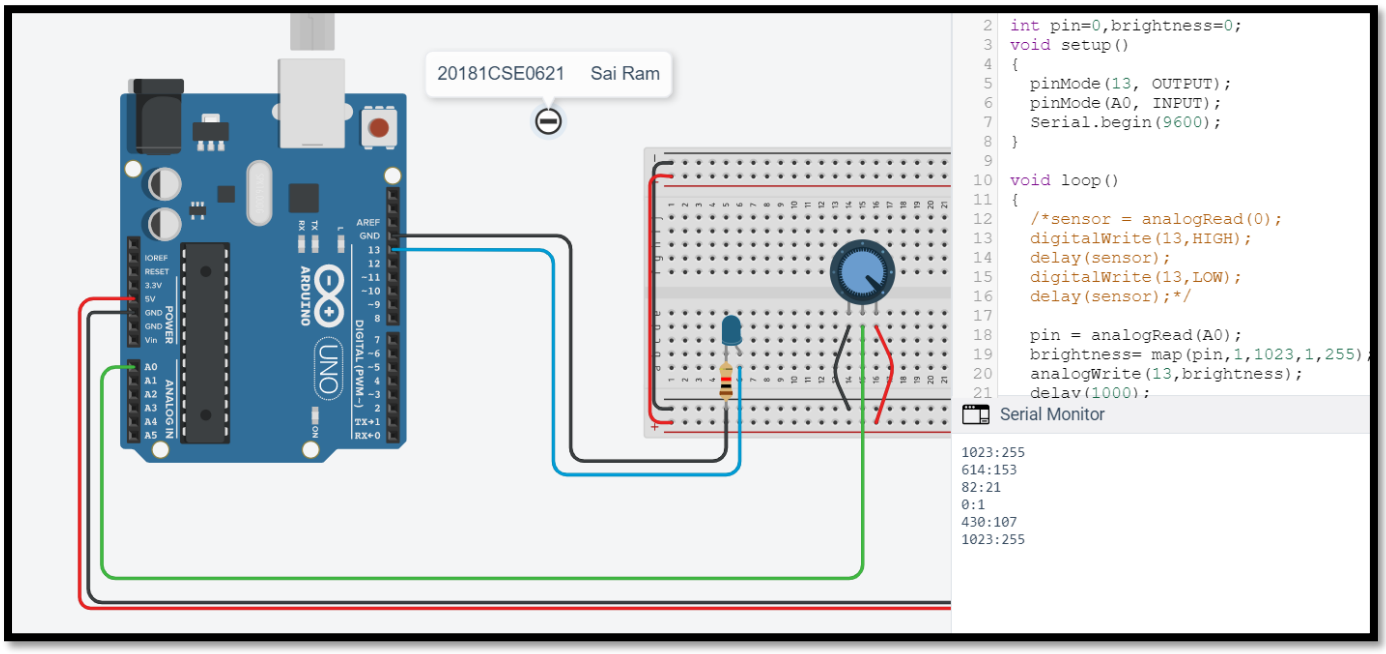


Sketch [Code] :

```
int sensor=0;

int pin=0,brightness=0;

void setup()

{
```

```
  pinMode(13, OUTPUT);

  pinMode(A0, INPUT);

  Serial.begin(9600);

}

void loop()

{

  /*sensor = analogRead(0);

  digitalWrite(13,HIGH);

  delay(sensor);

  digitalWrite(13,LOW);

  delay(sensor);*/

  pin = analogRead(A0);

  brightness= map(pin,1,1023,1,255);

  analogWrite(13,brightness);

  delay(1000);

  Serial.print(pin);

  Serial.print(":");

  Serial.println(brightness);

}
```

Output Screenshots :

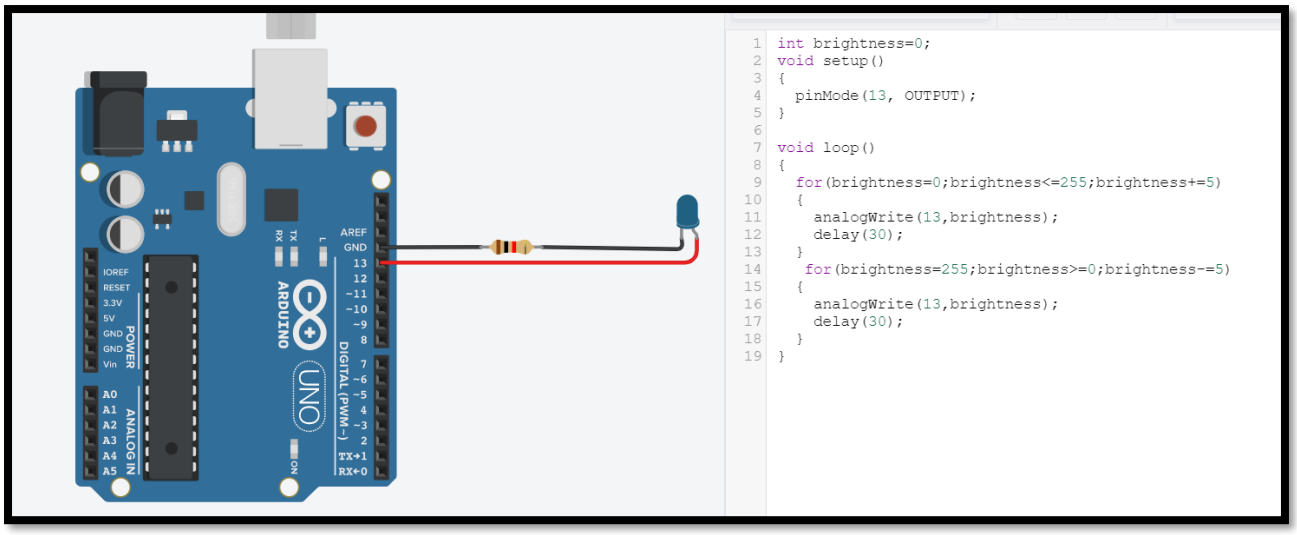


B. Adjusting the brightness without Potentiometer : -

Aim : Intensity of Led without using Potentiometer ..

Components : Arduino UNO, Led, Resistor, Tinckercad simulator .

Initial Circuit Design :

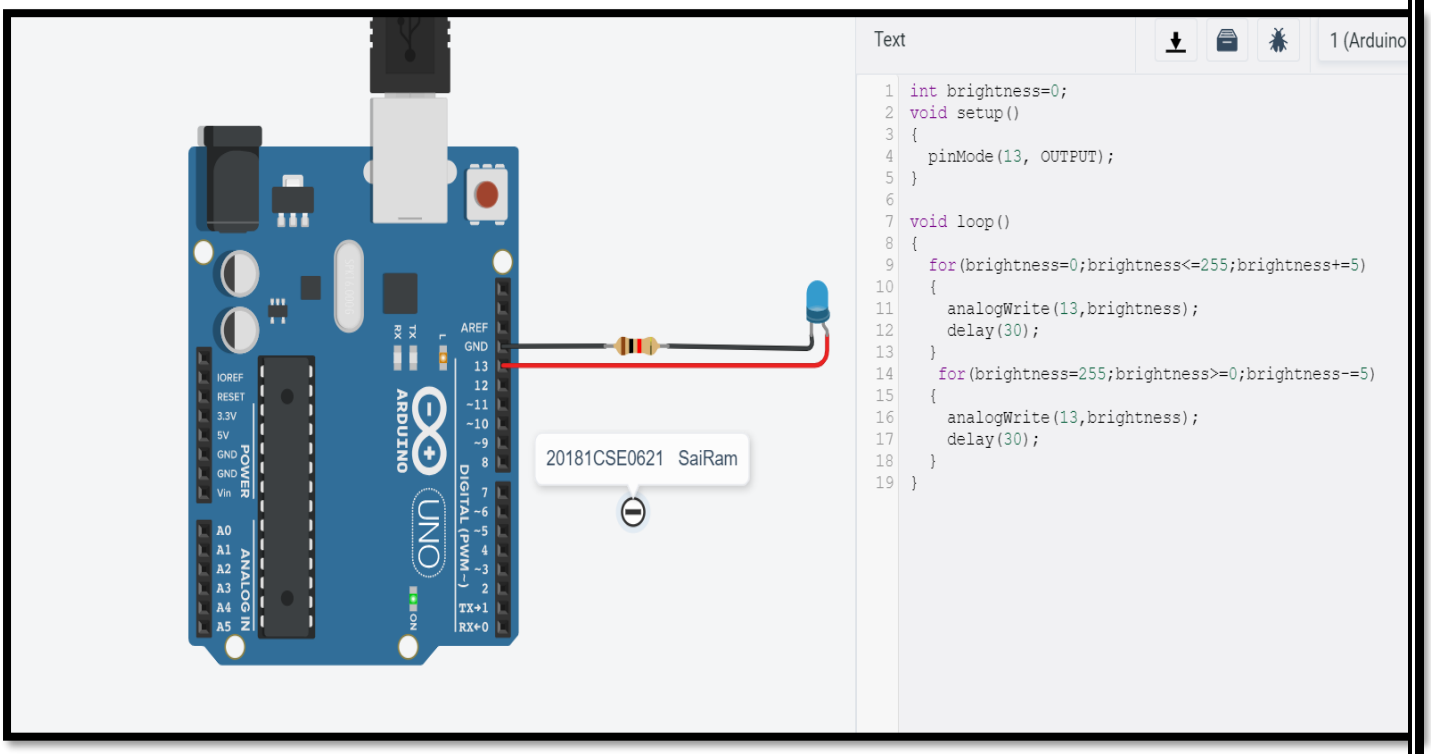


Sketch [Code] :

int brightness=0;

```
void setup()

{

  pinMode(13, OUTPUT);

}

void loop()

{

  for(brightness=0;brightness<=255;brightness+=5)

  {

    analogWrite(13,brightness);

    delay(30);

  }

   for(brightness=255;brightness>=0;brightness-=5)

  {

    analogWrite(13,brightness);

    delay(30);

  }

}
```

Output Screenshots :

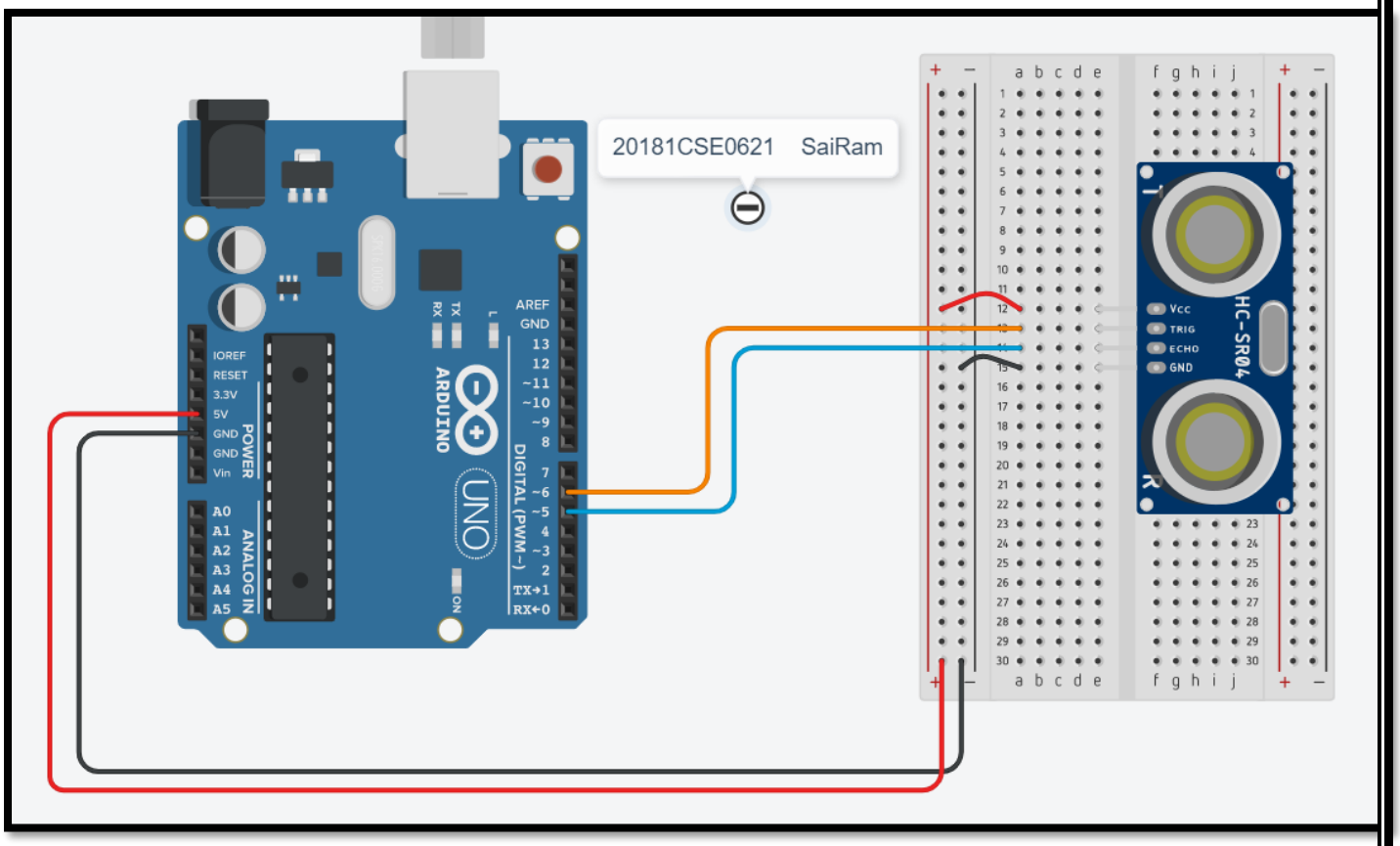

20181CSE0621 SaiRam
Text
1 (Arduino
int brightness=0;
void setup()
{
pinMode(13, OUTPUT);
}
void loop()
{
for(brightness=0;brightness<=255;brightness+=5)
{
analogWrite(13,brightness);
delay(30);
}
for(brightness=255;brightness>=0;brightness-=5)
{
analogWrite(13,brightness);
delay(30);
}
}

Experiment – 4
Ultrasonic Sensor

Aim : To find distance of an object using ultrasonic sensor.

Components : Arduino, bread board, jumper wires, resistor, ultrasonic sensor.

Initial Circuit Design :

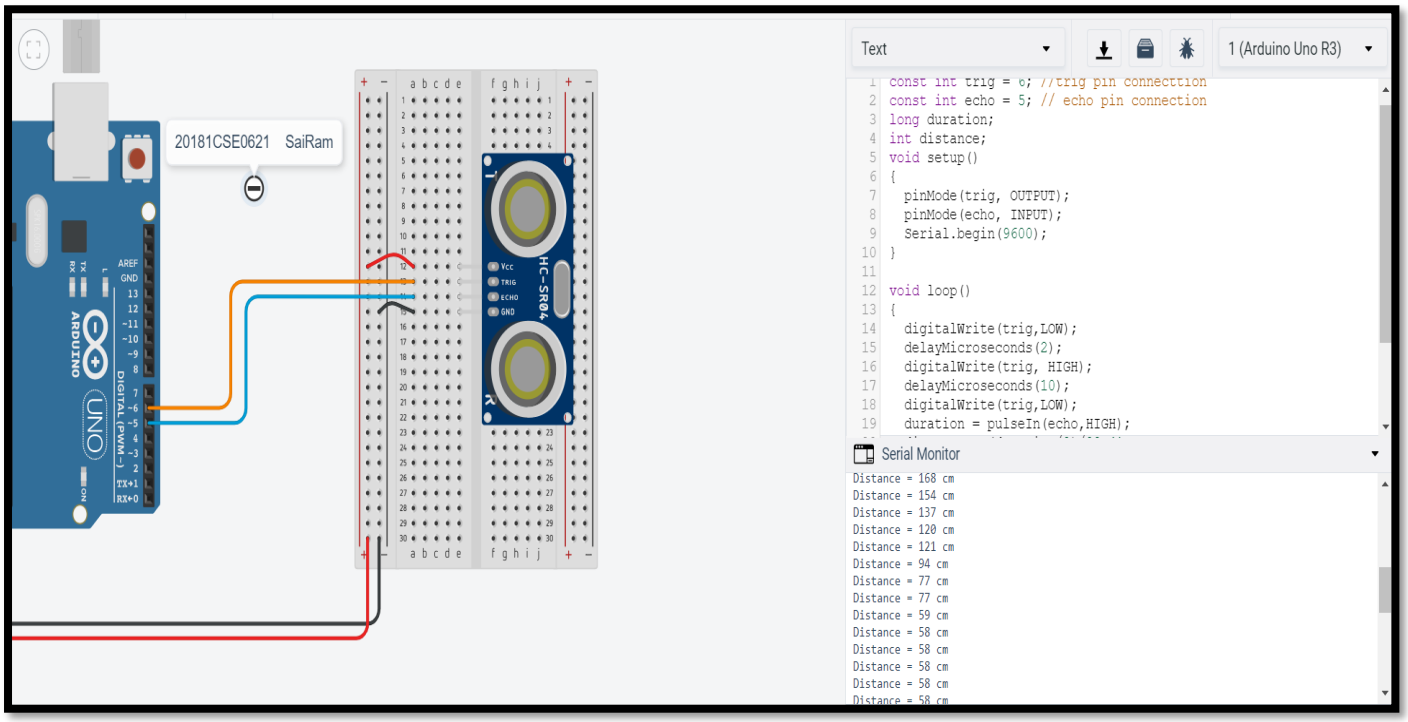


Sketch [Code] :

```
const int trig = 6; //trig pin connecttion

const int echo = 5; // echo pin connection

long duration;

int distance;

void setup()

{
```

```
  pinMode(trig, OUTPUT);

  pinMode(echo, INPUT);

  Serial.begin(9600);

}

void loop()

{

  digitalWrite(trig,LOW);

  delayMicroseconds(2);

  digitalWrite(trig, HIGH);

  delayMicroseconds(10);

  digitalWrite(trig,LOW);

  duration = pulseIn(echo,HIGH);

  distance = (duration/2)/29.41;

  Serial.print("Distance = ");

  Serial.print(distance);

  Serial.print(" cm");

  Serial.println("");

}
```

Output Screenshots :

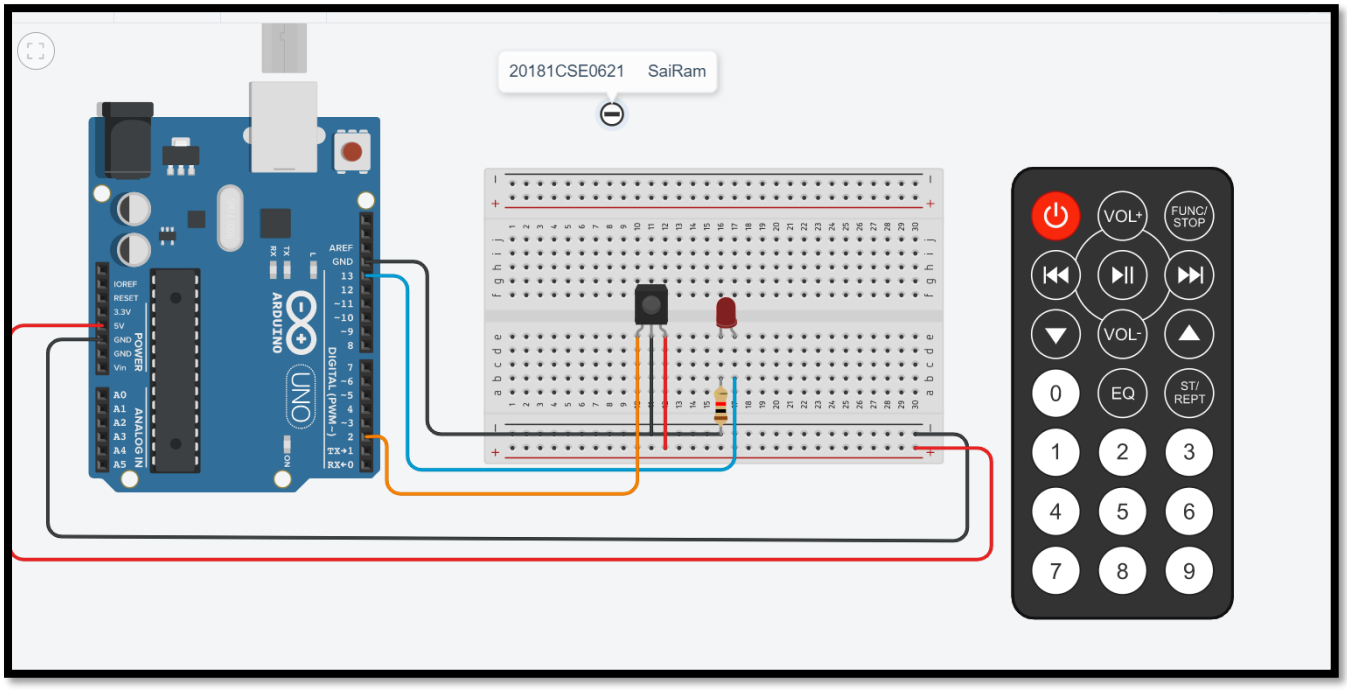


B.]

Aim : To find distance of an object using ultrasonic sensor.

Components : Arduino, bread board, jumper wires, resistor, ultrasonic sensor.

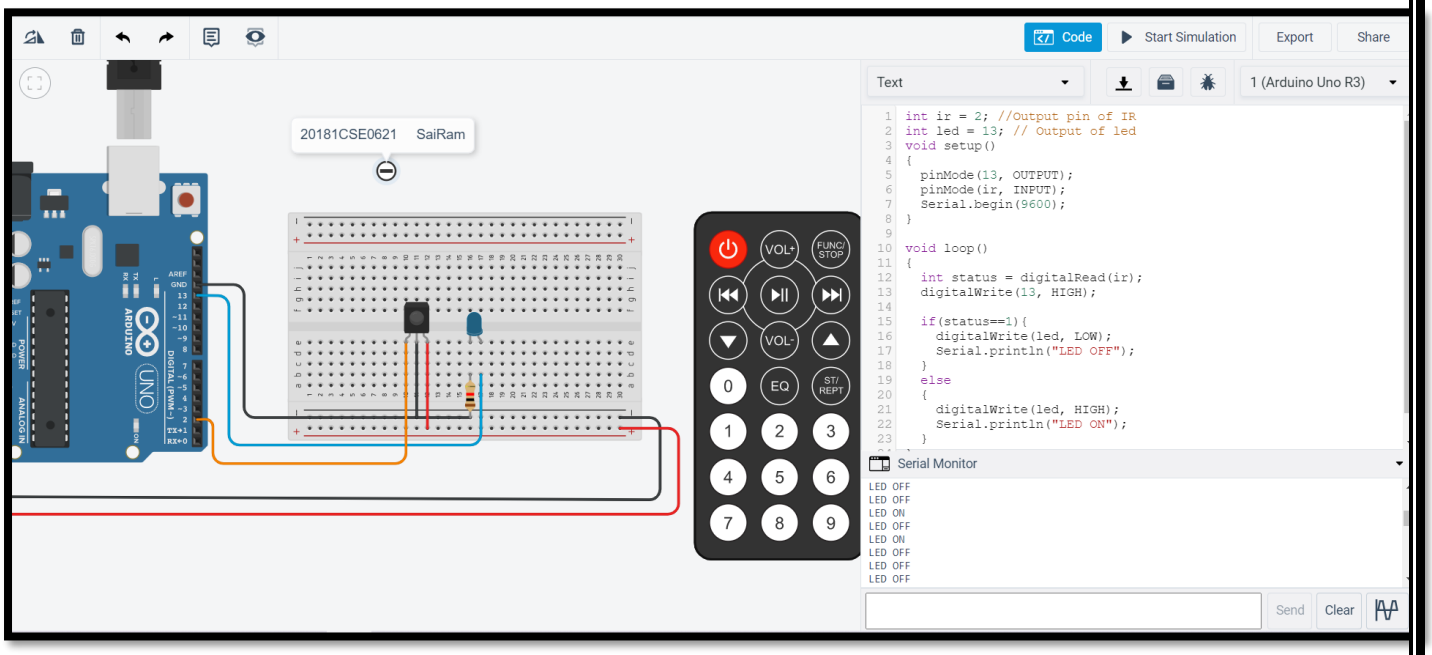
Initial Circuit Design :



Sketch [Code] :

```
int ir = 2; //Output pin of IR
int led = 13; // Output of led
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(ir, INPUT);
  Serial.begin(9600);
}
void loop()
{
  int status = digitalRead(ir);
  digitalWrite(13, HIGH);
  if(status==1){
   digitalWrite(led, LOW);
    Serial.println("LED OFF");
  }
  else
  {
    digitalWrite(led, HIGH);
    Serial.println("LED ON");
  }
}
```

Output Screenshots :

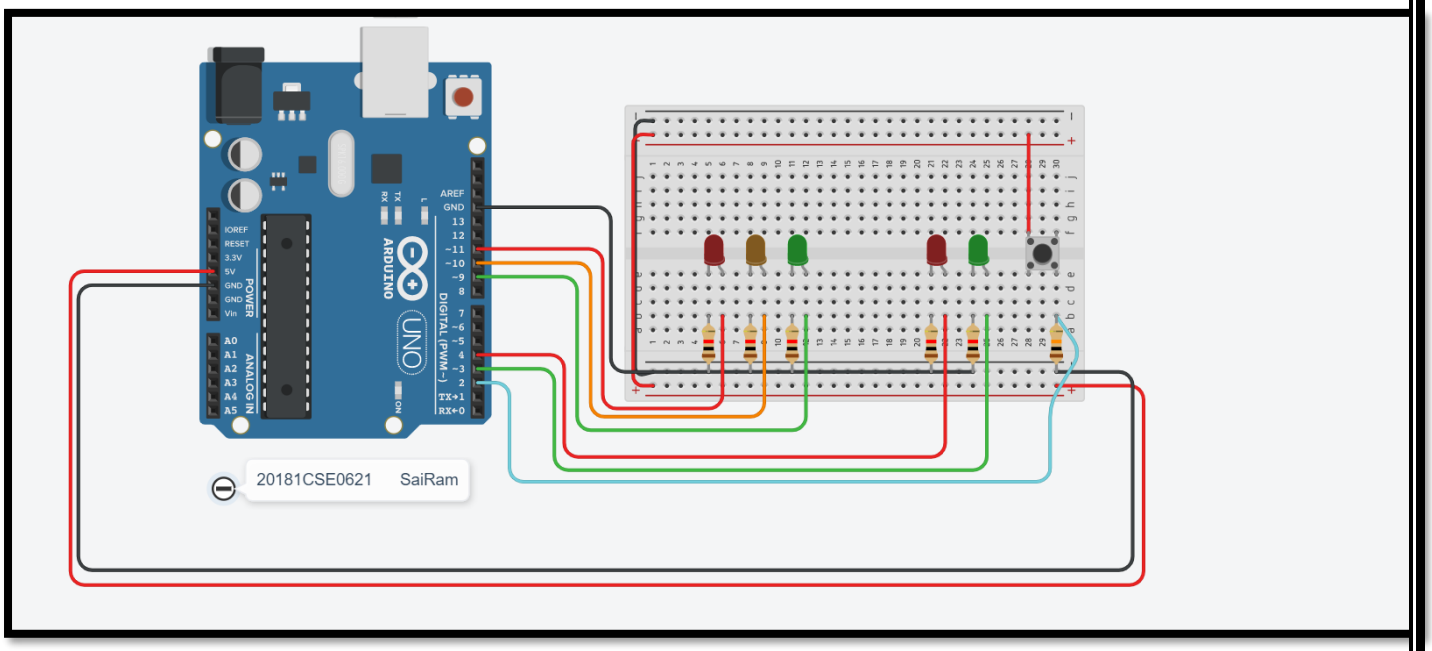


Experiment – 5
Traffic Control System

Aim : To control traffic lights on pedestrian and vehicle side.

Components : Arduino, bread board, jumper wires, resistor.

Initial Circuit Design :



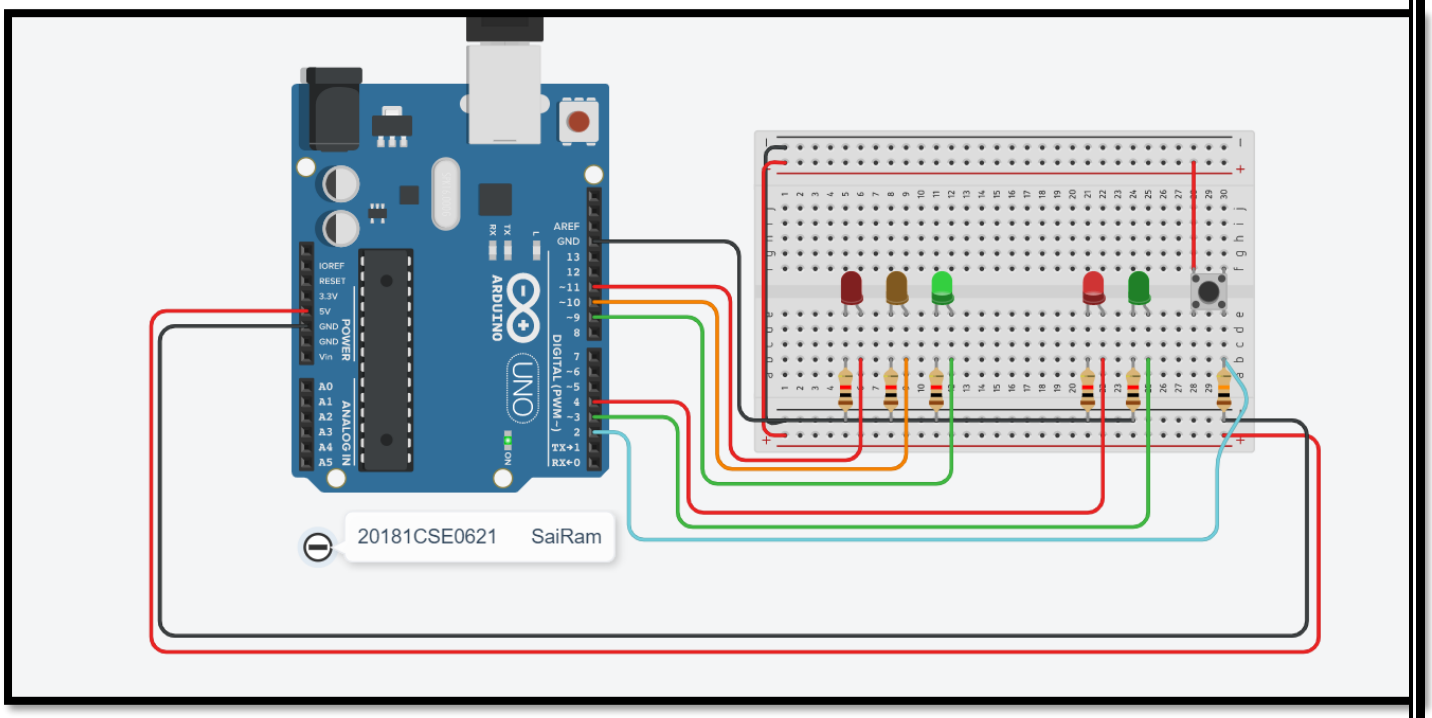
Sketch [Code] :

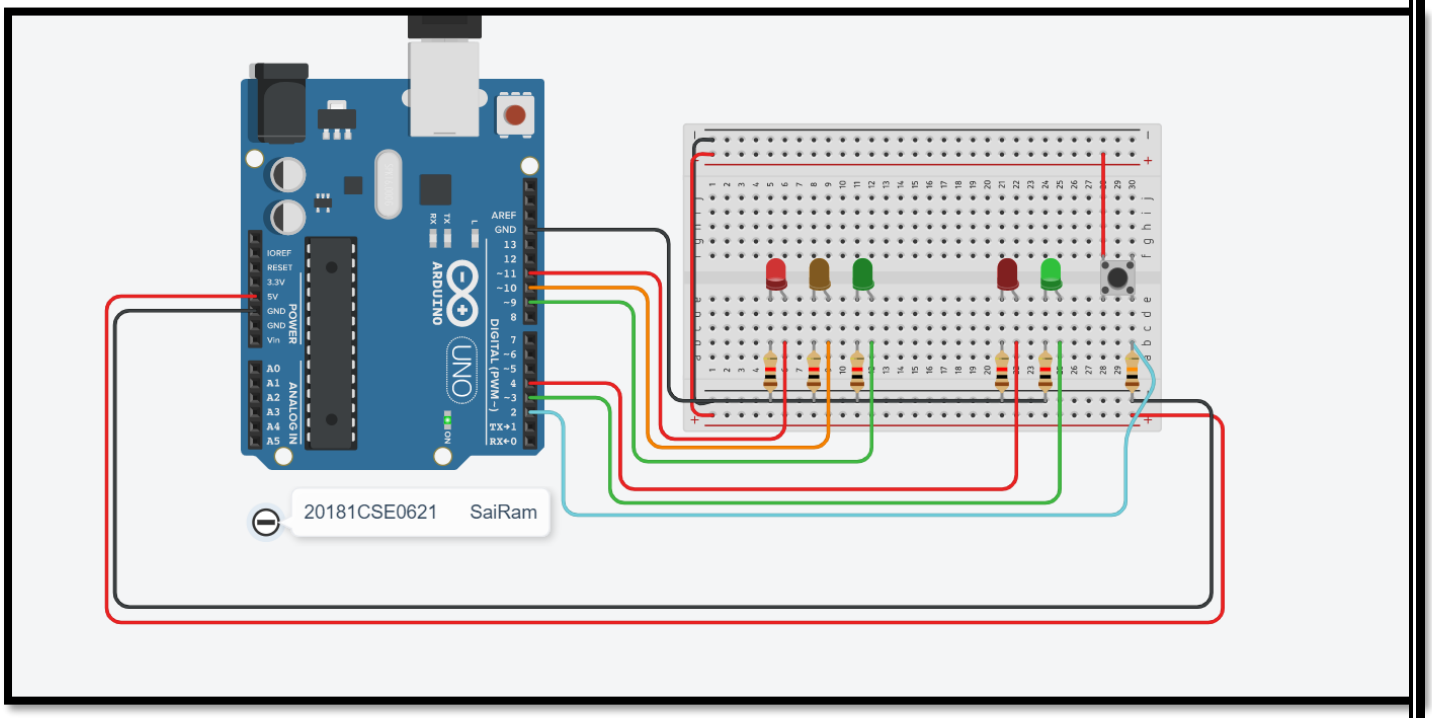
```
int carRed =11;

int carYellow=10;

int carGreen=9;

int pedRed=4;

int pedGreen=3;

int push=2;

int crossTime = 5000;

unsigned long changeTime;

void setup()

{
```

```
  pinMode(carRed, OUTPUT);
  pinMode(carYellow, OUTPUT);
  pinMode(carGreen, OUTPUT);
  pinMode(pedRed, OUTPUT);
  pinMode(pedGreen, OUTPUT);

  pinMode(push, INPUT);
  digitalWrite(carGreen,HIGH);
  digitalWrite(pedRed,HIGH);
}

void loop()
{
  int state = digitalRead(push);
  if(state==HIGH && (millis() - changeTime) > 5000)
  {
    changeLights();
  }

}
void changeLights()
{
  digitalWrite(carGreen, LOW);
  digitalWrite(carYellow, HIGH);
  delay(2000);
  digitalWrite(carYellow, LOW);
  digitalWrite(carRed, HIGH);
```

```
  delay(1000);

  digitalWrite(pedRed, LOW);

  digitalWrite(pedGreen, HIGH);



  delay(crossTime);

  for(int i=0;i<10;++i)

  {

    digitalWrite(pedGreen,HIGH);

    delay(1000);

    digitalWrite(pedGreen, LOW);

    delay(250);

  }

  digitalWrite(pedRed,HIGH);

  delay(500);

  digitalWrite(carYellow, HIGH);

  digitalWrite(carRed, 0);

  delay(1000);

  digitalWrite(carGreen, HIGH);

  digitalWrite(carYellow, LOW);

}
```

Output Screenshots :


20181CSE0621 SaiRam

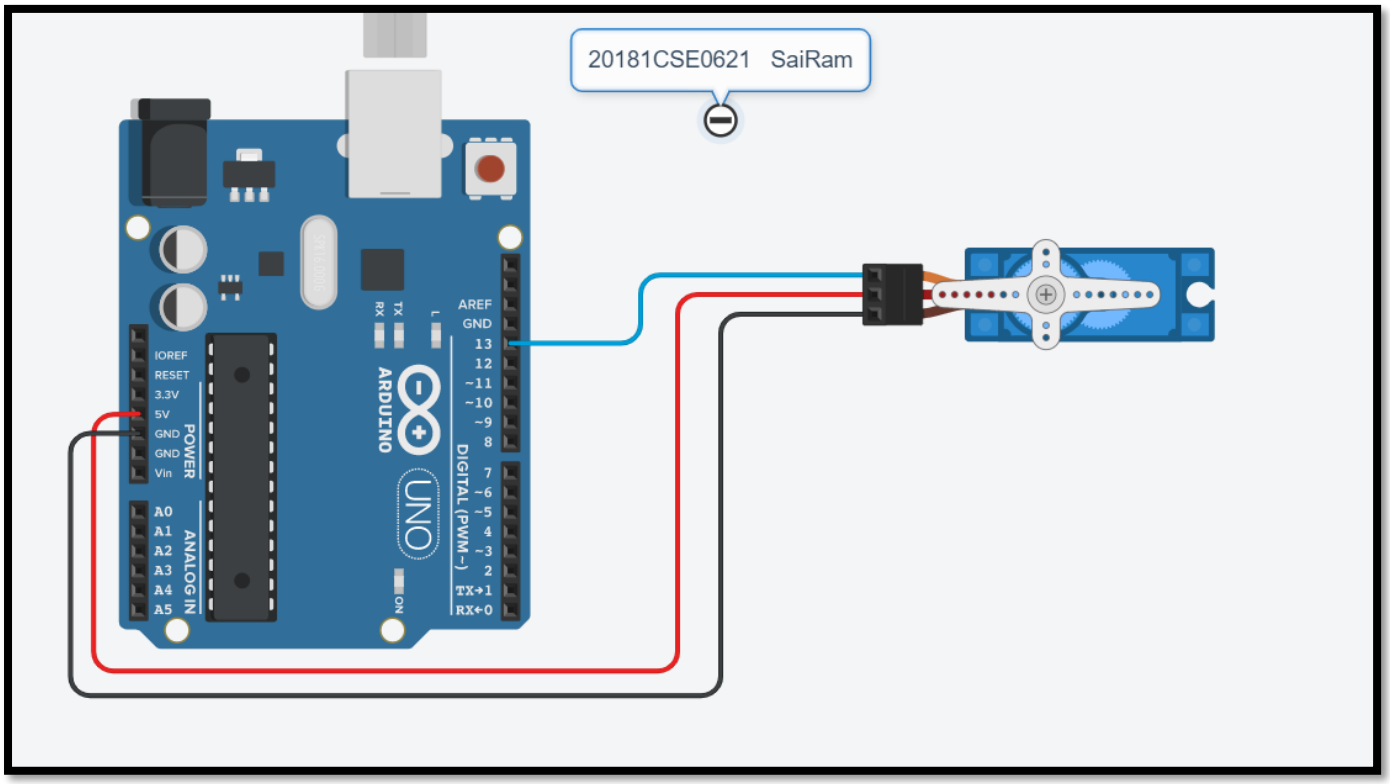

20181CSE0621 SaiRam

Experiment – 6
Servo Motor

Aim : To rotate the servo motor.

Components : Arduino, bread board, jumper wires, micro-servo.

Initial Circuit Design :

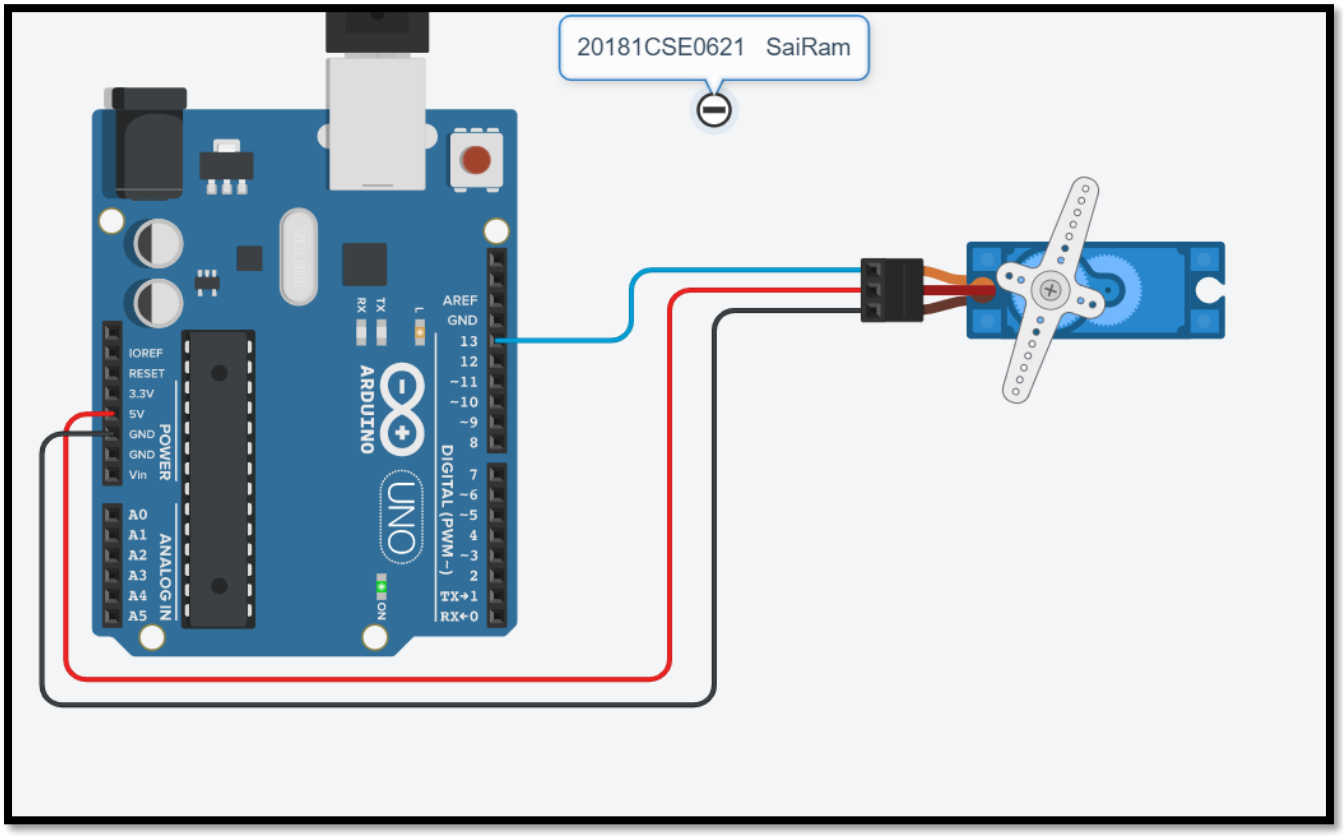


Sketch [Code] :

```
#include<Servo.h>

Servo servo; //object to access funcns in servo library

int pos=0;

void setup()

{

                                                    servo.attach(13);

}
```

```
void loop()

{

 for(pos=0;pos<=180;pos++)

 {

  servo.write(pos);

  delay(20);

 }

 for(pos=180;pos>=0;pos--)

 {

  servo.write(pos) ;

  delay(20);

 }

}
```

Output Screenshots :

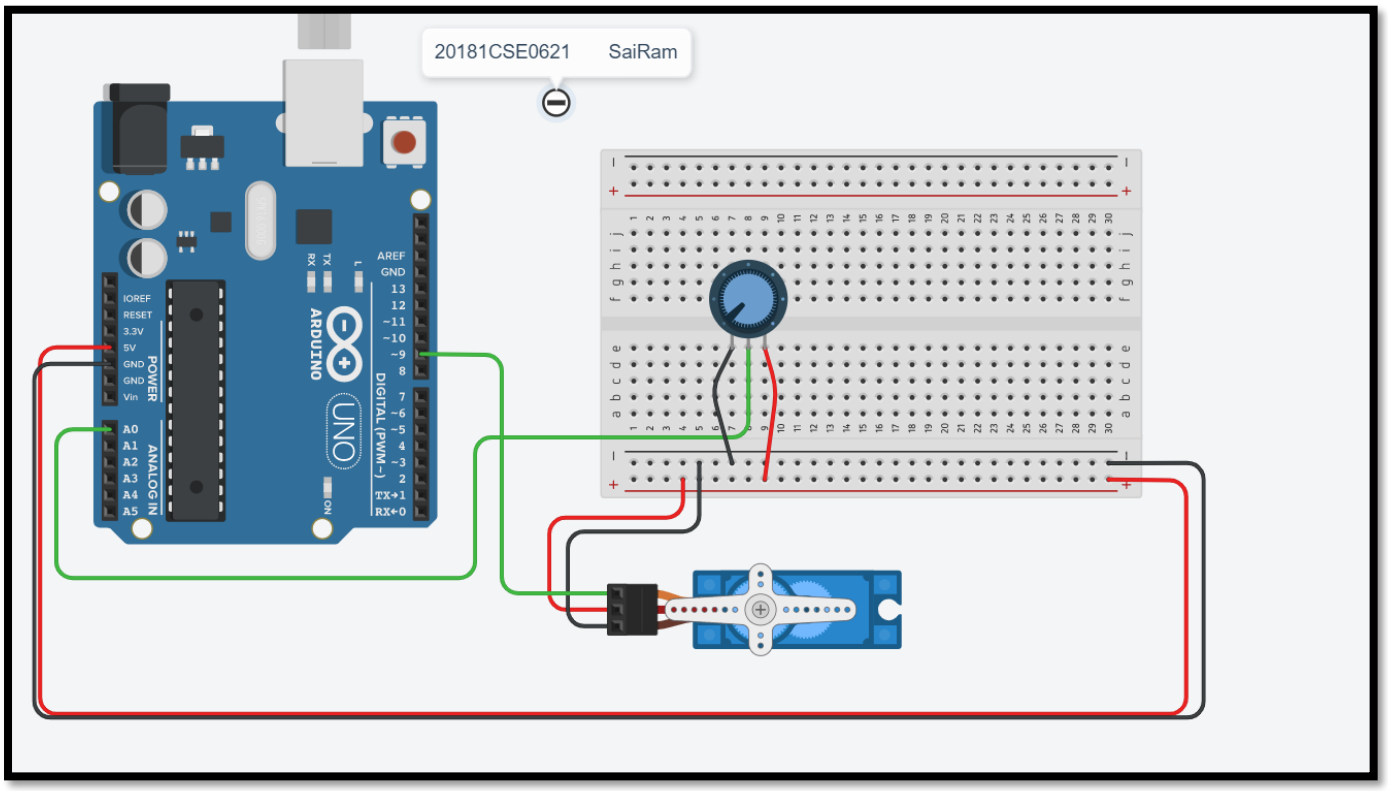


B. Servo with Potentiometer

Aim : To rotate the servo motor using **Potentiometer**.

Components : Arduino, bread board, jumper wires, micro-servo.

Initial Circuit Design :

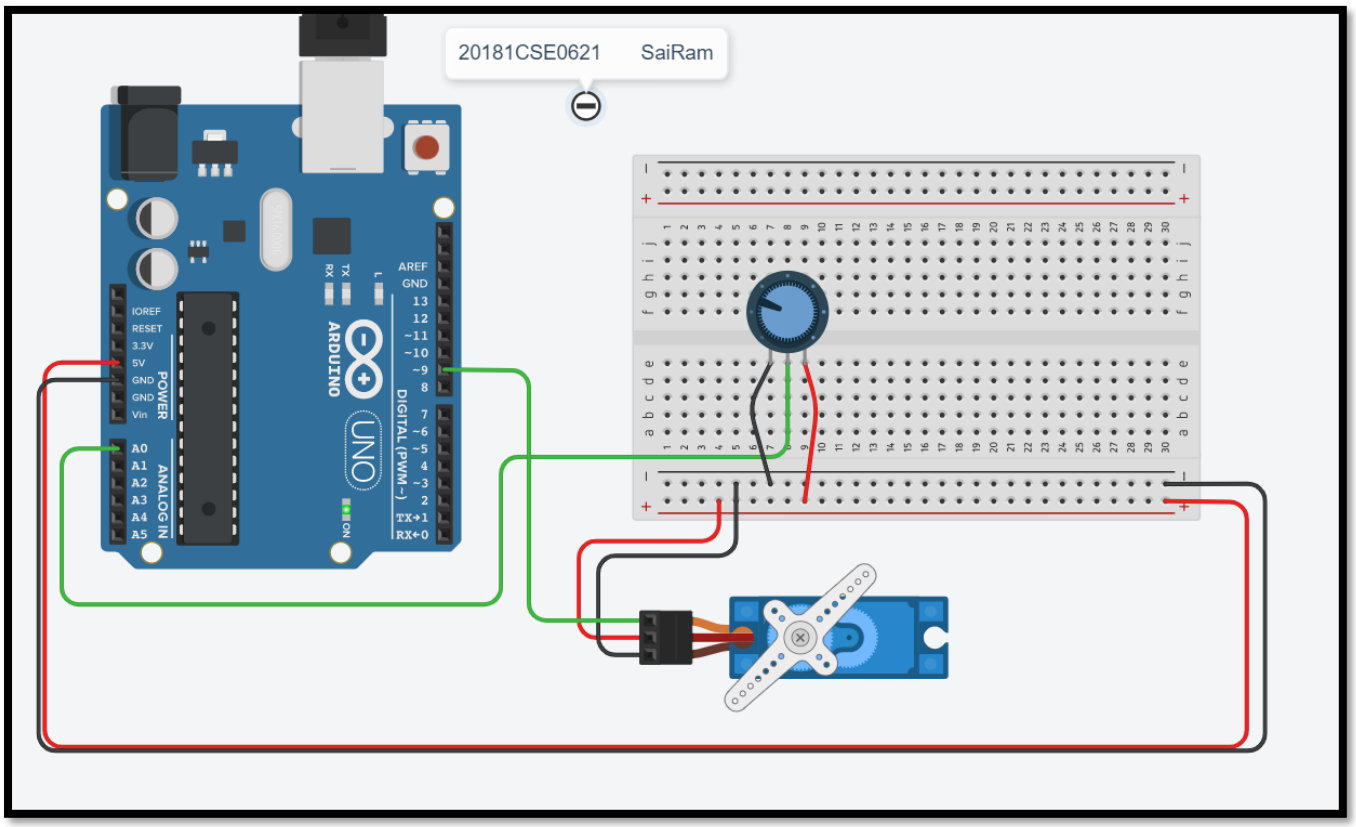


Sketch [Code] :

```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer

int val;    // variable to read the value from the analog pin

void setup() {

  myservo.attach(9);  // attaches the servo on pin 9 to the servo object

}

void loop() {
```

```
  val = analogRead(potpin);            // reads the value of the potentiometer (value between 0 and 1023)

  val = map(val, 0, 1023, 0, 180);     // scale it to use it with the servo (value between 0 and 180)

  myservo.write(val);                  // sets the servo position according to the scaled value

  delay(15);                           // waits for the servo to get there

}
```

Output Screenshots :

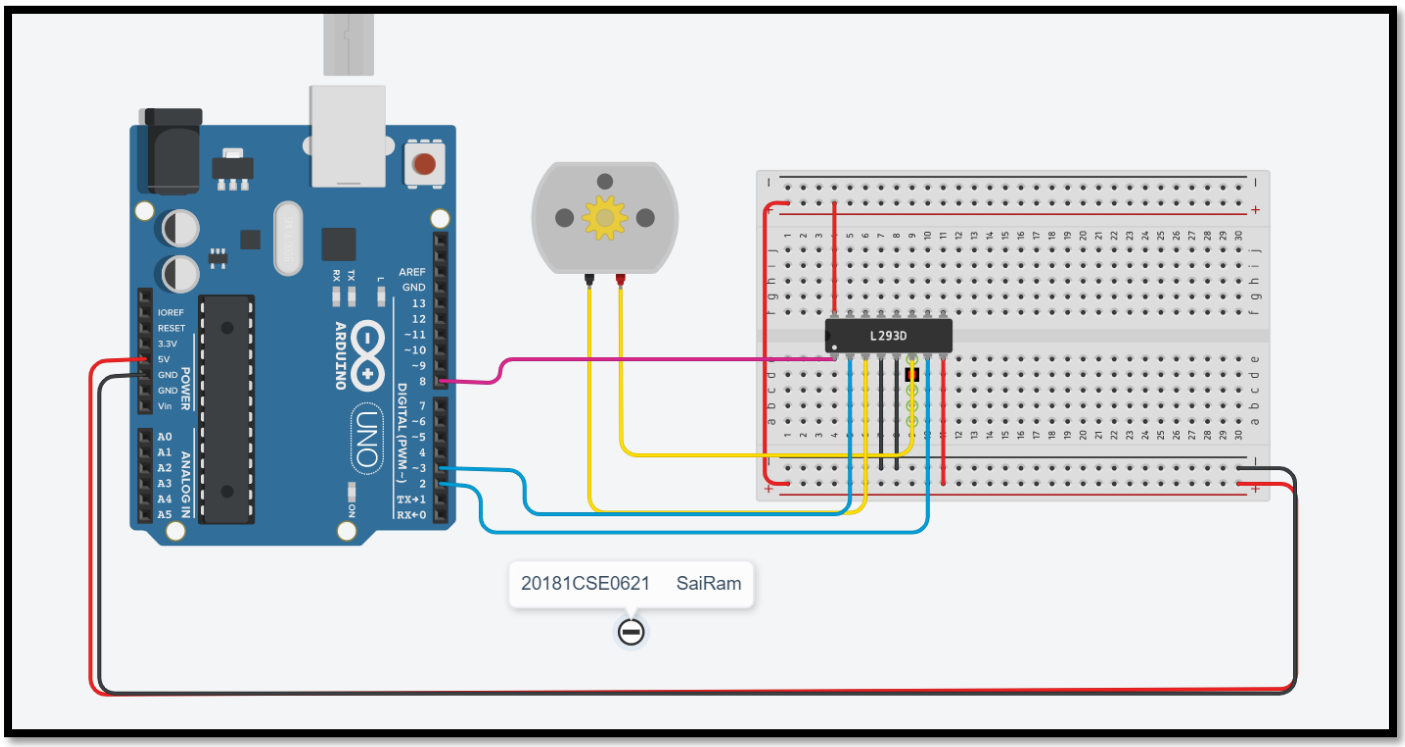


Experiment – 7
D.C Motor

Aim : To Interface a D.C motor with arduino

Components : Arduino, bread board, jumper wires, IC L293D.

Initial Circuit Design :

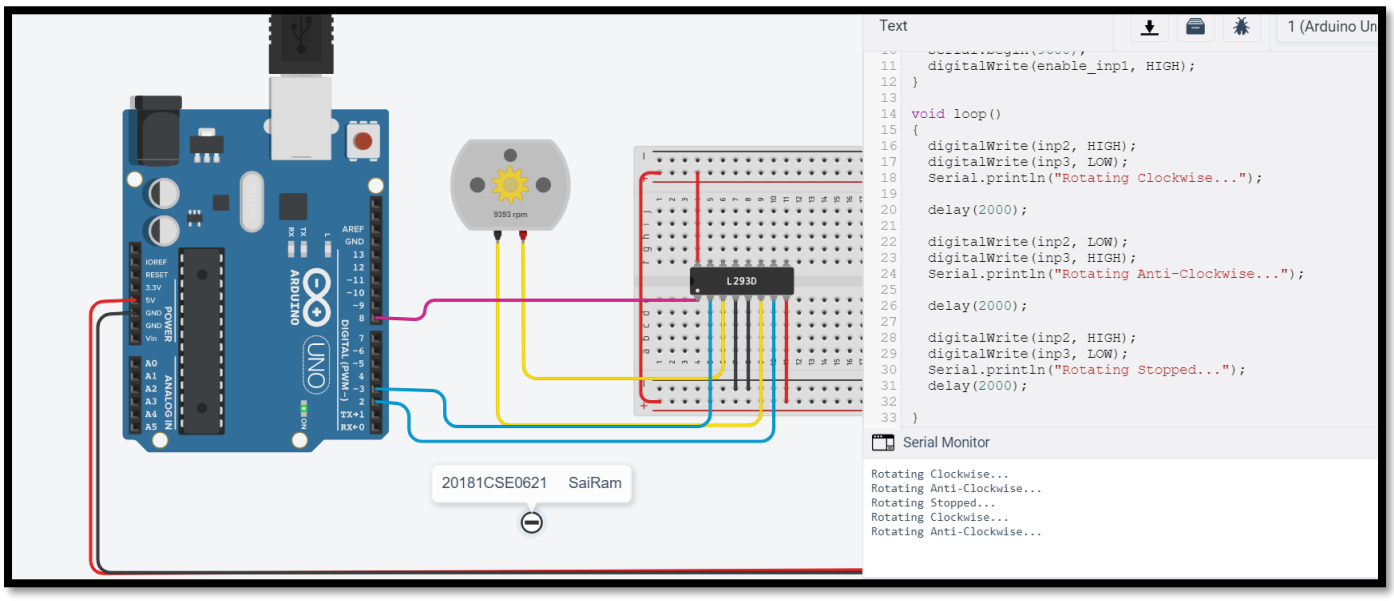


Sketch [Code] :

```
int enable_inp1 = 8,inp2 = 2;

int inp3 =  3;

void setup()

{

  pinMode(13, OUTPUT);

  pinMode(enable_inp1, OUTPUT);

  pinMode(inp2, OUTPUT);

  pinMode(inp3, OUTPUT);

  Serial.begin(9600);
```

```
  digitalWrite(enable_inp1, HIGH);

}

void loop()

{

  digitalWrite(inp2, HIGH);

  digitalWrite(inp3, LOW);

  Serial.println("Rotating Clockwise...");

  delay(2000);

  digitalWrite(inp2, LOW);

  digitalWrite(inp3, HIGH);

  Serial.println("Rotating Anti-Clockwise...");

  delay(2000);

  digitalWrite(inp2, HIGH);

  digitalWrite(inp3, LOW);

  Serial.println("Rotating Stopped...");

  delay(2000);

}
```

Output Screenshots

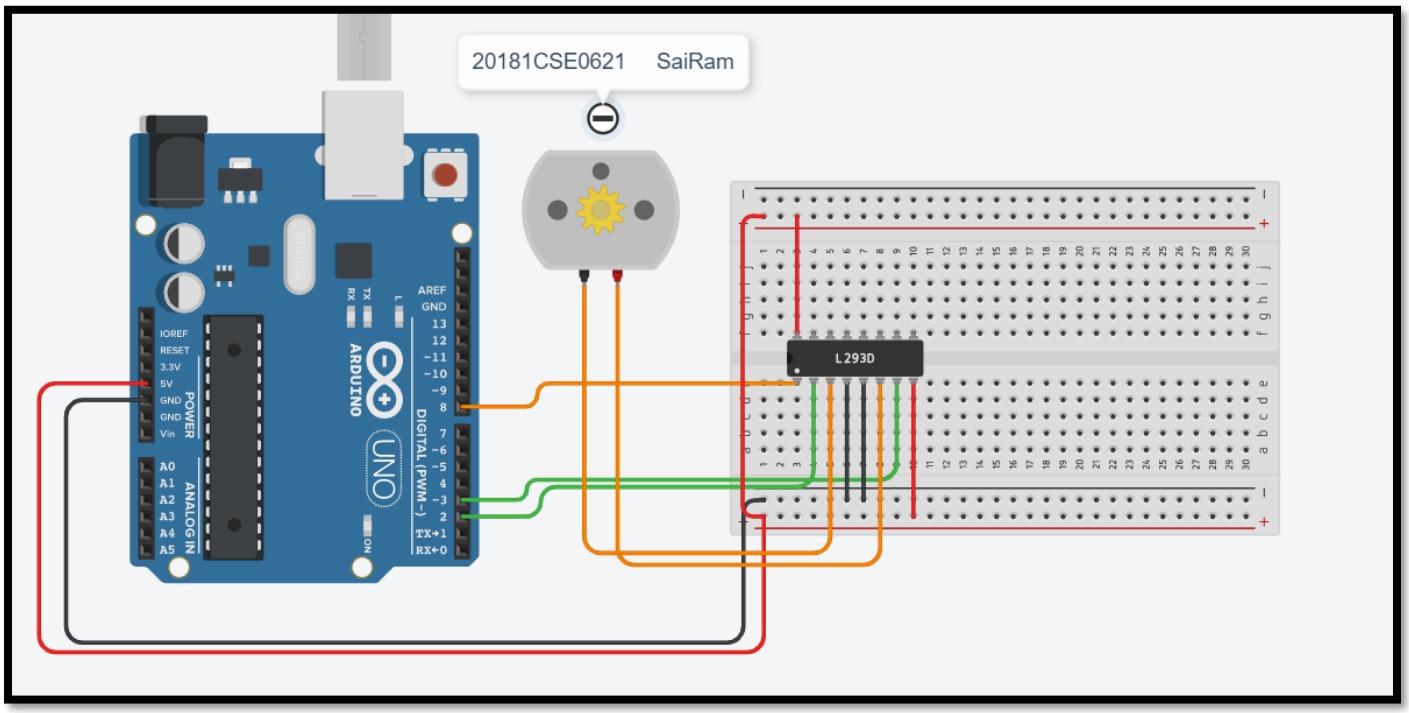


Experiment – 7B
D.C Motor

Aim : To Interface a D.C motor with arduino

Components : Arduino, bread board, jumper wires, IC L293D.

Initial Circuit Design :

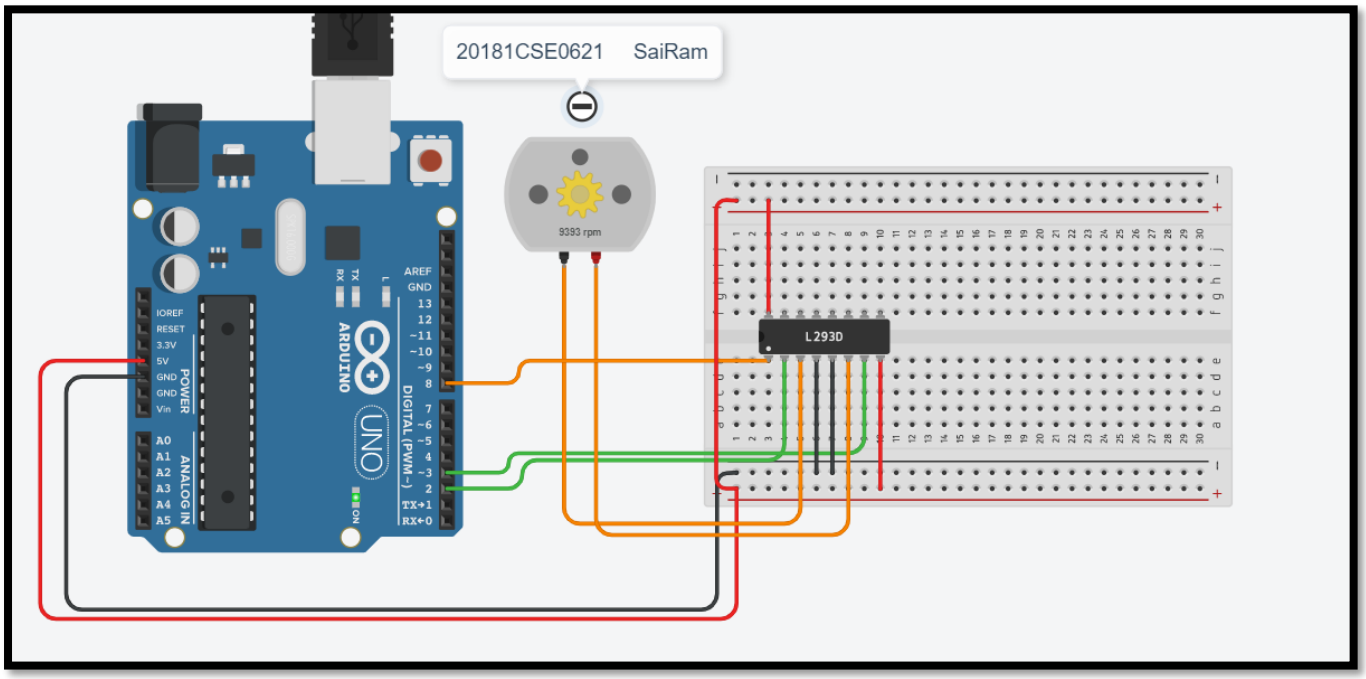


Sketch [Code] :

```
int enable=8,inp1=2,inp2=3;

int x=0,pwm=0;

void setup()

{

  digitalWrite(enable,HIGH);

  pinMode(enable,OUTPUT); //3-enable,9-inp1,11,inp2

  pinMode(inp1,OUTPUT);

  pinMode(inp2,OUTPUT);

  Serial.begin(9600);
```

```
}


void loop()

{

  digitalWrite(inp1,HIGH);

  digitalWrite(inp2,LOW);

  delay(2000);

  x=5;

  digitalWrite(inp2,HIGH);

  digitalWrite(inp1,LOW);

  delay(2000);

  pwm = map(x,0,5,0,255);

  analogWrite(3,pwm);

}
```

Output Screenshots



Alternate code for speed using input from serial monitor

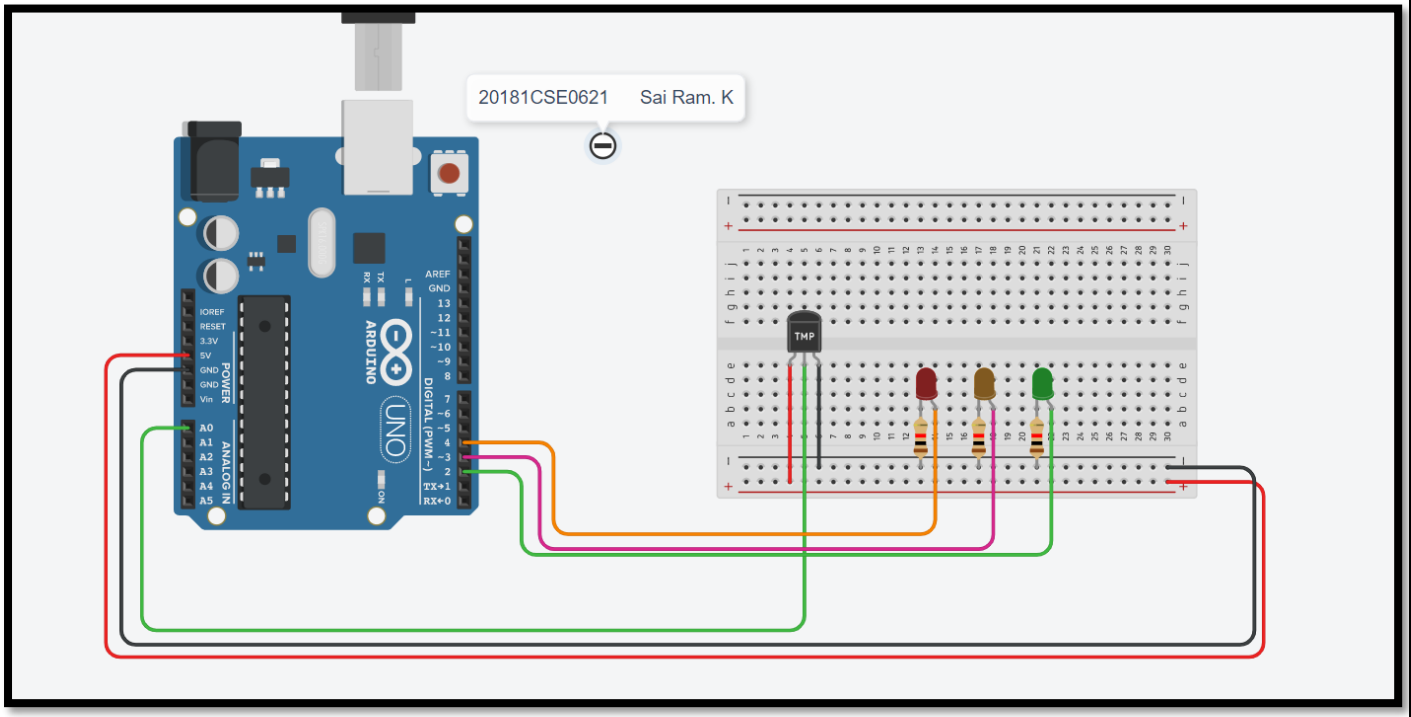
```
int enable=8,inp1=2,inp2=3;
int x=0,pwm=0;
void setup()
{
  digitalWrite(enable,HIGH);
  pinMode(enable,OUTPUT); //3-enable,9-inp1,11,inp2
  pinMode(inp1,OUTPUT);
  pinMode(inp2,OUTPUT);
  Serial.begin(9600);
}
void loop()
{
if(Serial.available())
  {
    int speed = Serial.parseInt();
    Serial.print(speed);
    analogWrite(enable, speed);
    digitalWrite(inp1,HIGH);
    digitalWrite(inp2,LOW);
  }
}
```

Experiment - 8
Temperature Sensor

Aim : To display temperature of room.

Components : Arduino Uno, Breadboard, Temperature sensor, Jumper wires.

Initial Circuit Design :

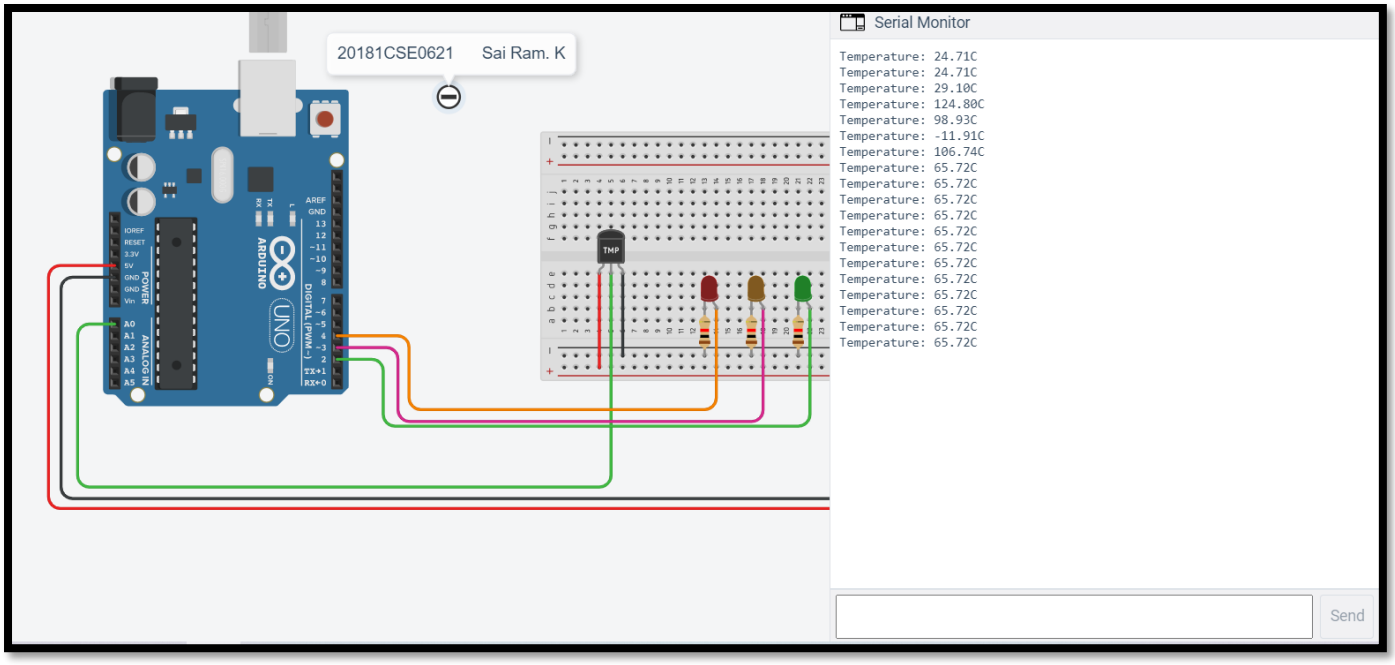


Sketch :

```
int pinTemp = A0;

double temp;

void setup()

{

  Serial.begin(9600);

}

void loop()
```

```
{
  int sinput = analogRead(A0);    //Read the analog pin
  temp = (double)sinput / 1024;
  temp=temp*5;
  temp=temp-0.5;
  temp=temp*100;

  Serial.print("Temperature: ");
  Serial.print(temp);
  Serial.println("C");  //print the temperature status
  delay(1000);
}
```

OUTPUT:

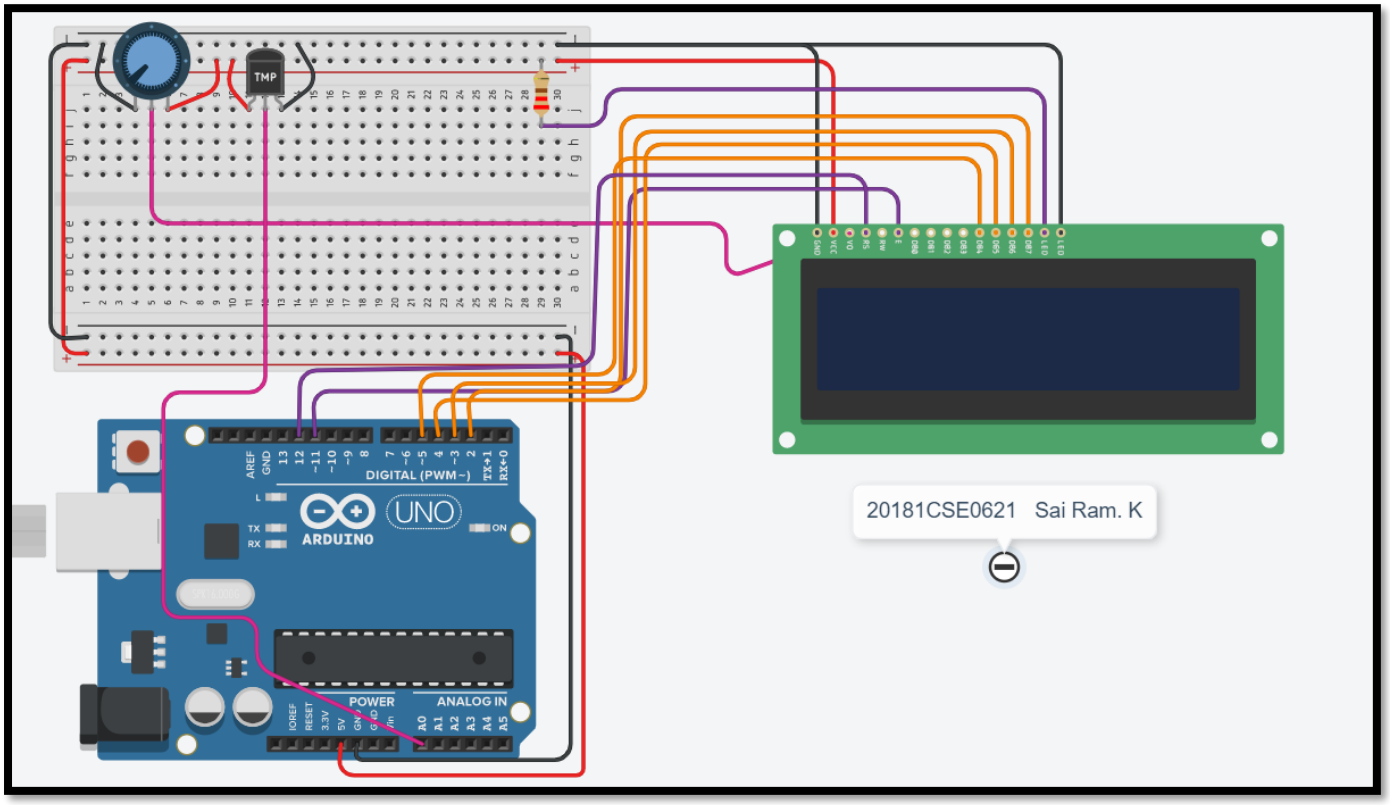


Experiment - 9
Interfacing of temperature sensor with Arduino uno.
Display room temperature on serial monitor and LCD.

Aim : To display temperature of room on LCD.

Components : Arduino Uno, Breadboard, Temperature sensor, Jumper wires, potentiometer, LCD.

Initial Circuit Design :

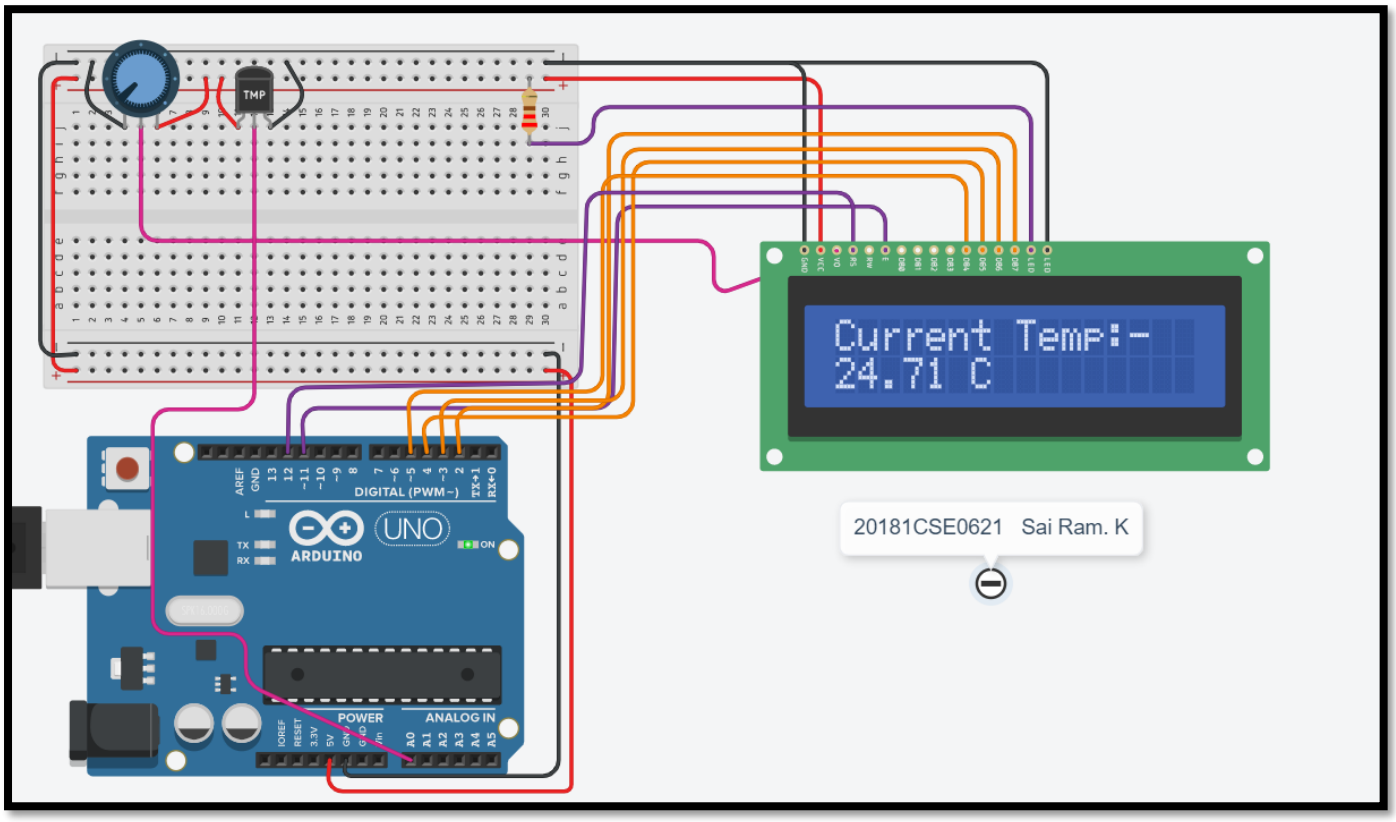


Sketch :

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12,11,5,4,3,2);

double temp;

void setup() {

  lcd.begin(16, 2);
```

```
  lcd.print("Current Temp:-");

  Serial.begin(9600);

}

void loop() {

  lcd.setCursor(0, 1);

  int temperature = analogRead(A0);  //Read the analog pin

  temp = (double)temperature / 1024;

  temp=temp*5;

  temp=temp-0.5;

  temp=temp*100;

  Serial.print("Temperature: ");

  Serial.print(temp);

  Serial.println("C");  //print the temperature status

  delay(1000);

  lcd.print(temp);

  lcd.print(" C");

}
```

OUTPUT:


TMP
Current Temp:-
24.71 C
20181CSE0621 Sai Ram. K
ARDUINO
UNO
DIGITAL (PWM~)
POWER
ANALOG IN

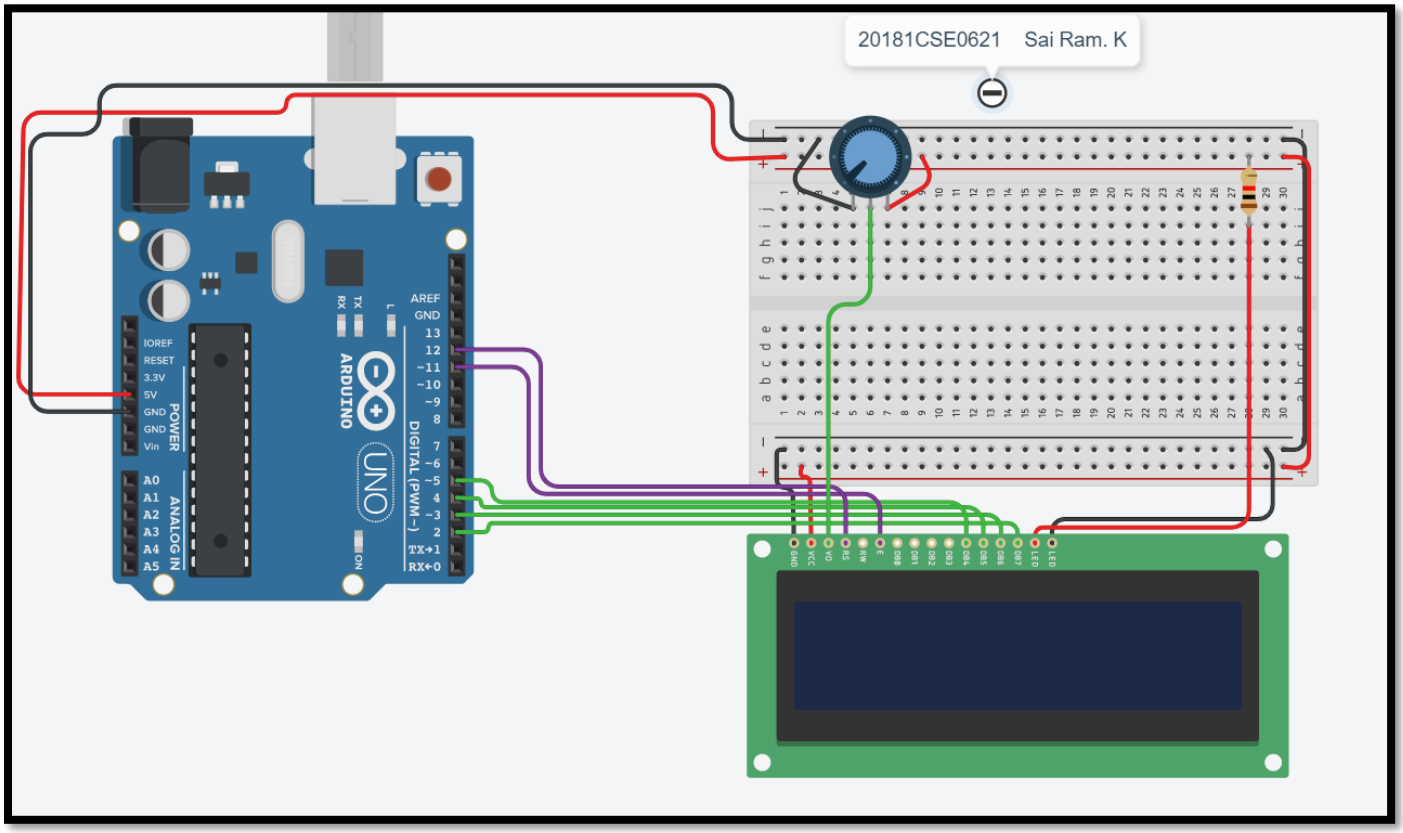
Experiment - 10

Interfacing of a display device i.e LCD with Arduino uno. WAP to display ‘ Hello IOT’ on LCD.

Aim : To display ‘Hello IOT’ on LCD.

Components : Arduino Uno, Breadboard, Jumper wires, potentiometer, LCD.

Initial Circuit Design :

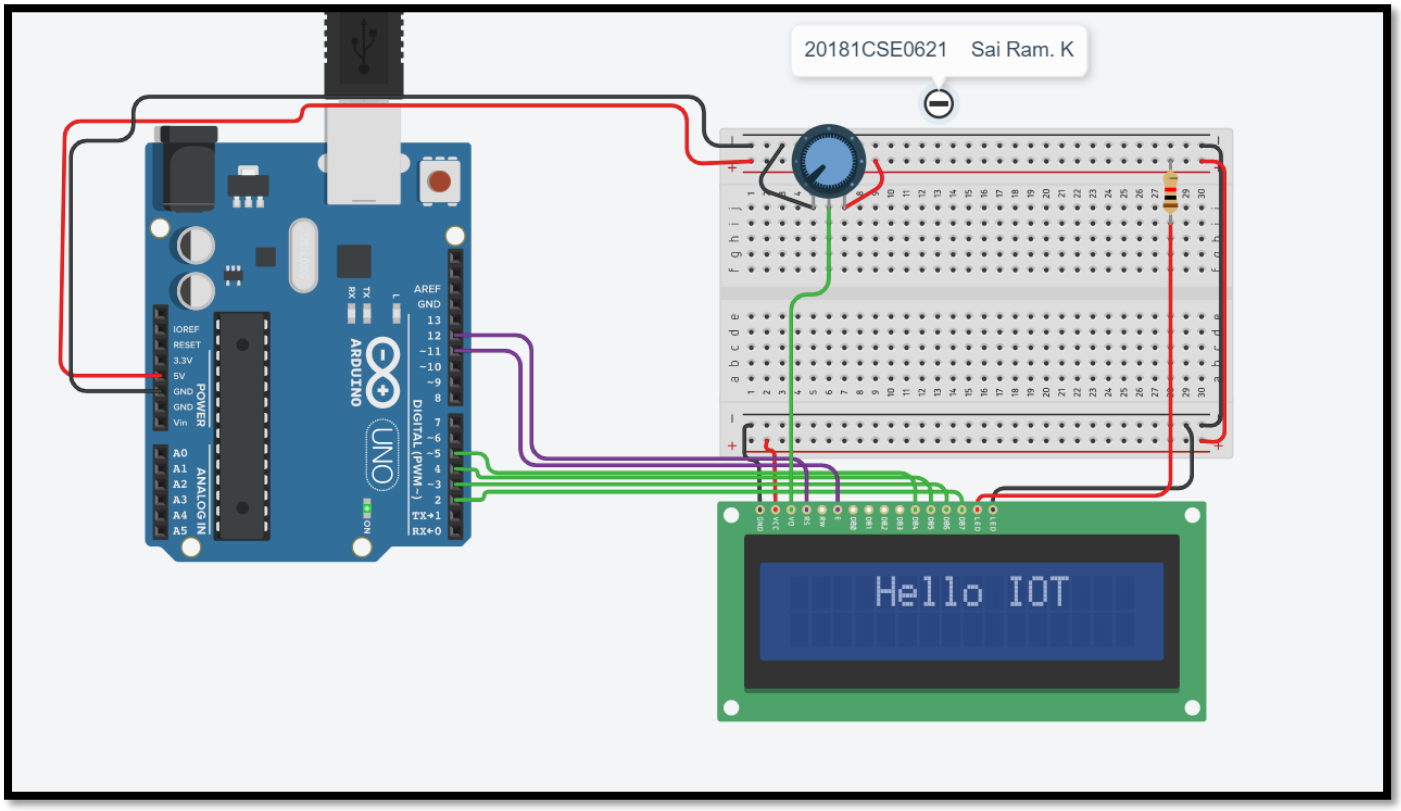


Sketch :

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12,11,5,4,3,2);

void setup() {

  lcd.begin(16, 2);

  Serial.begin(9600);

}
```

```
void loop() {
  lcd.setCursor(4,0);
  Serial.println("Hello IOT");
  lcd.print("Hello IOT");
  delay(1000);
}
```

Output :



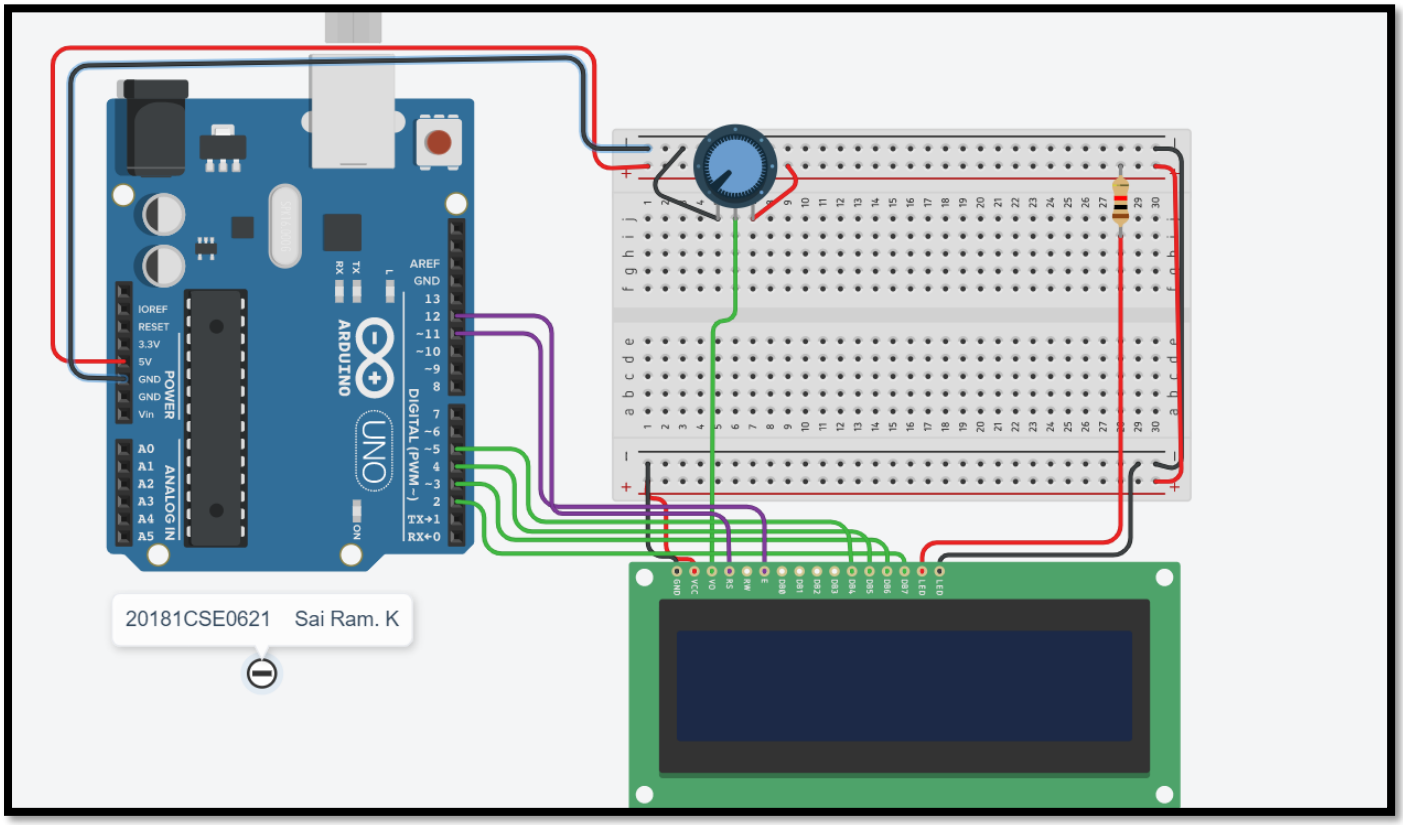
Experiment – 10B

Interfacing of a display device i.e LCD with Arduino uno. WAP to display ‘ Hello IOT’ on LCD.

Aim : To scroll text on LCD.

Components : Arduino Uno, Breadboard, Jumper wires, potentiometer, LCD.

Initial Circuit Design :



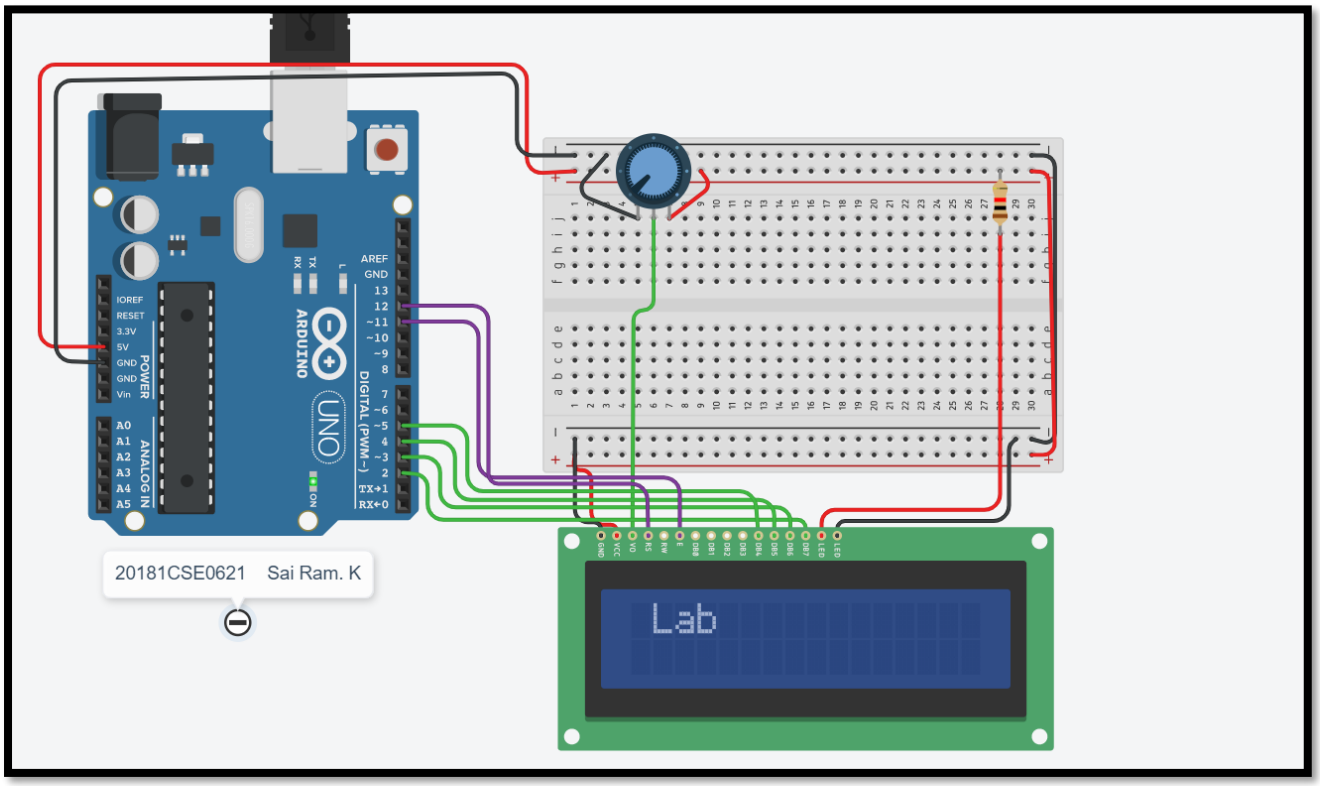
Code:

```
#include <LiquidCrystal.h>

const int rs=12, en=11,d4=5,d5=4,d6=3,d7=2;

LiquidCrystal lcd(rs,en,d4,d5,d6,d7);

void setup()

{

  lcd.begin(16, 2);
```

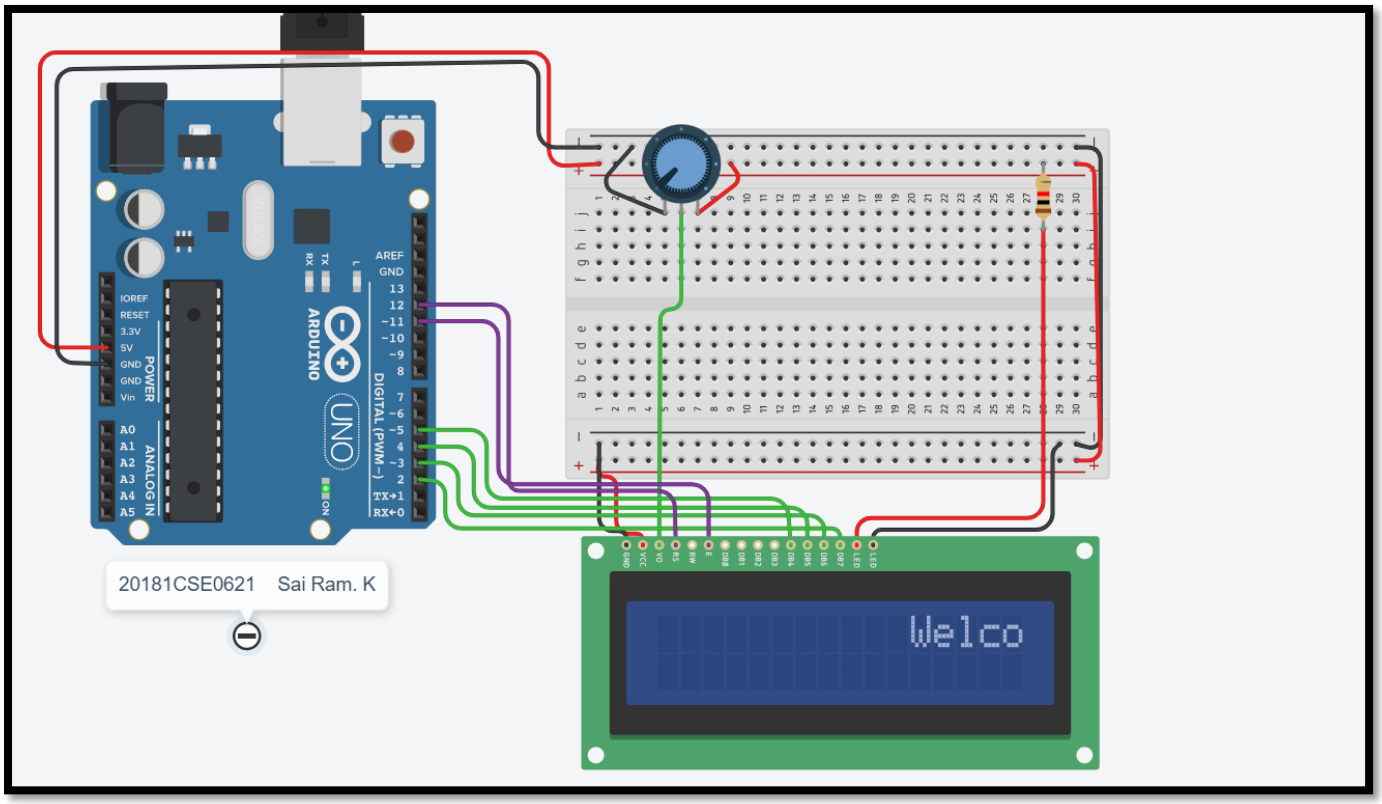
```
 lcd.print("Welcome to IOT Lab");

 delay(1000);

 Serial.begin(9600);

}

void loop()

{

 //Scroll Left

 for(int count=0;count<16;count++)

 {

  lcd.scrollDisplayLeft();

  delay(200);

 }

 //Sroll Right

  for(int count=0;count<16+10+15;count++)

 {

  lcd.scrollDisplayRight();

   delay(200);

 }

 lcd.clear();

 lcd.setCursor(0,0);

}
```

OUTPUT :

Scroll Left



Scroll Right



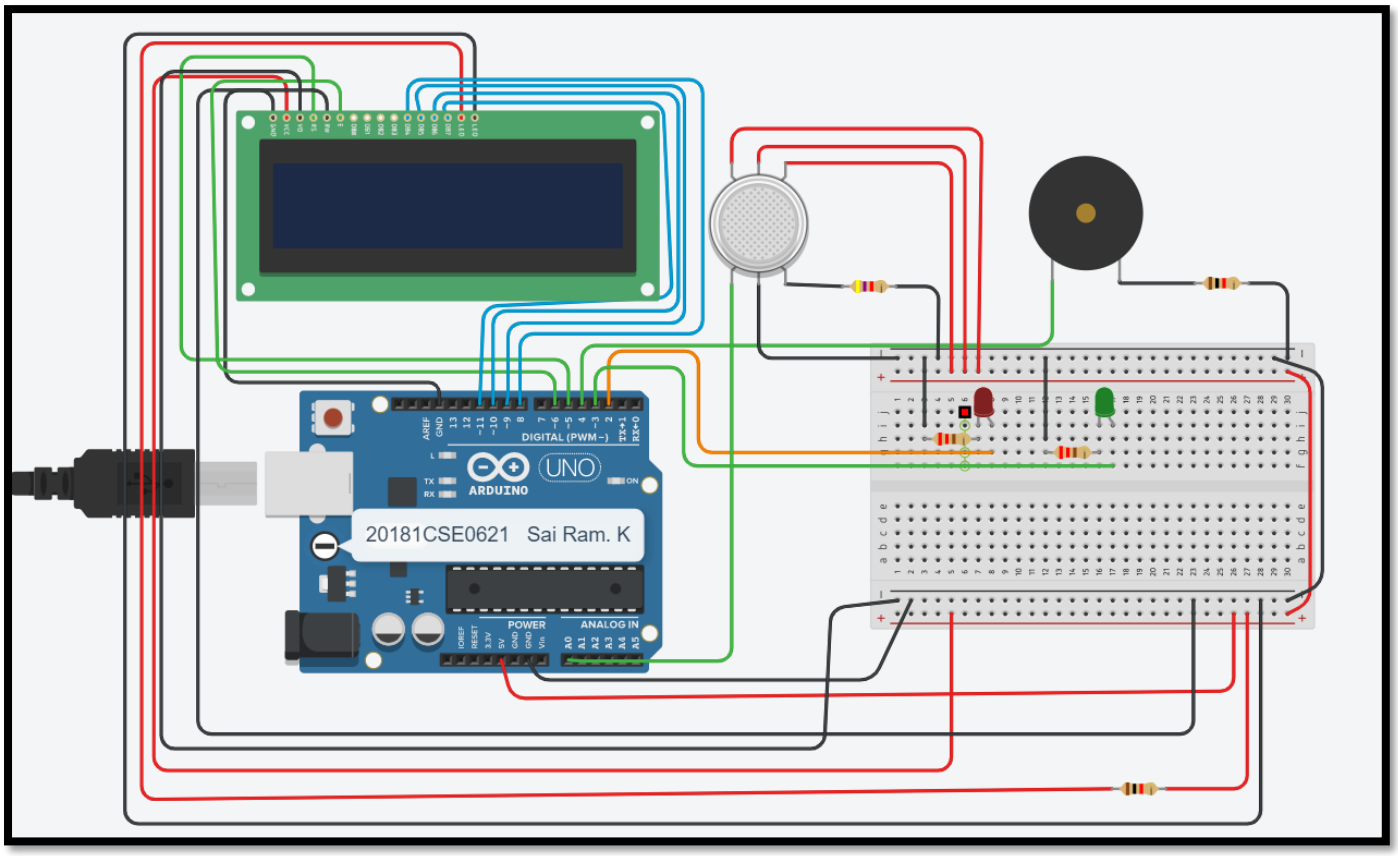
Experiment – 11

Design a Gas Leakage Detector using Gas sensor in TinkerCAD

Aim : To interface a gas leakage detector.

Components : Arduino Uno, Breadboard, Jumper wires, Gas Sensor, LCD, Piezo, Resistors.

Initial Circuit Design :



Code :

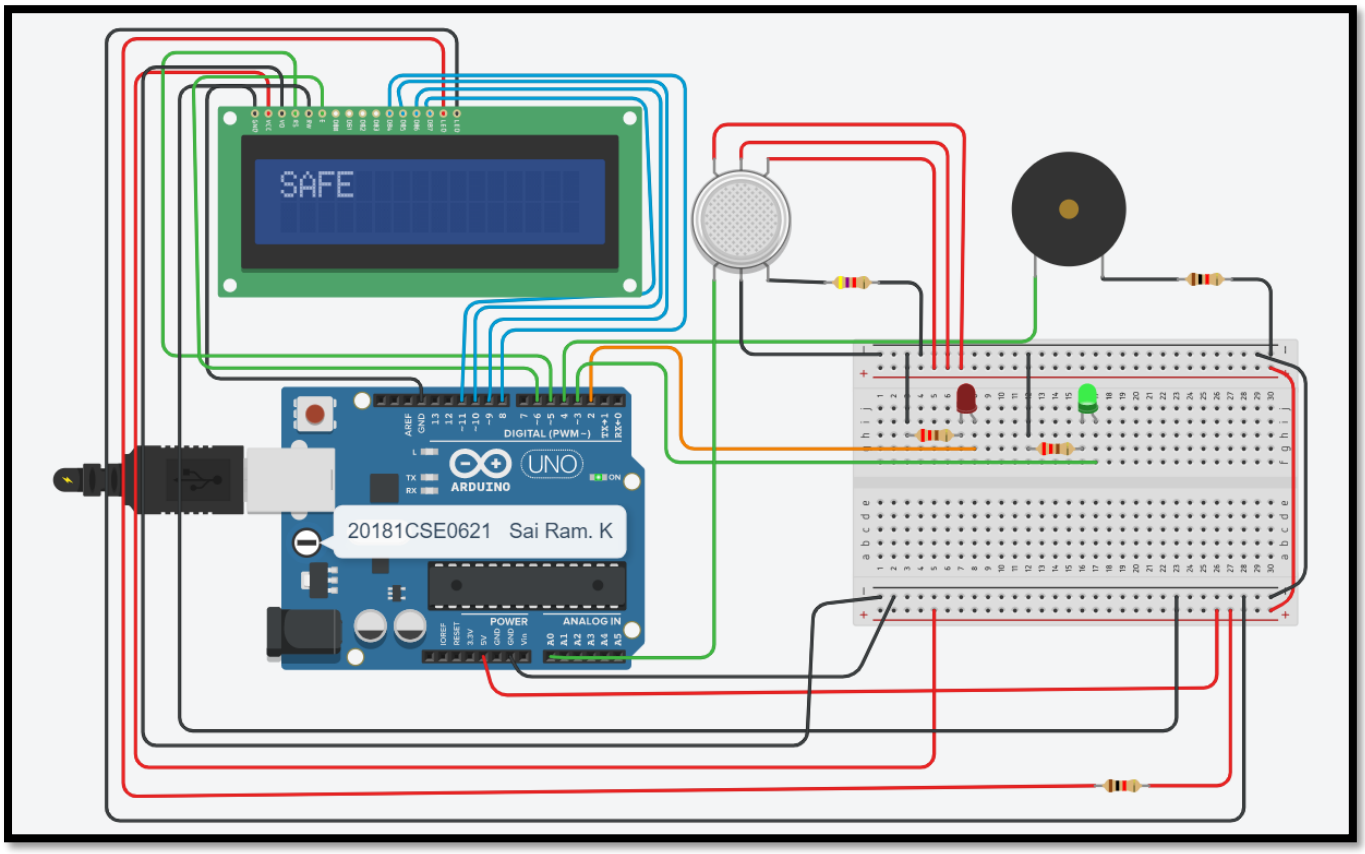
```
#include <LiquidCrystal.h>

LiquidCrystal lcd(5,6,8,9,10,11);

int redled = 2;

int greenled = 3;

int buzzer = 4;

int sensor = A0;

int sensorThresh = 400;
```

```
void setup()
{
pinMode(redled, OUTPUT);
pinMode(greenled,OUTPUT);
pinMode(buzzer,OUTPUT);
pinMode(sensor,INPUT);
Serial.begin(9600);
lcd.begin(16,2);
}
void loop()
{
 int analogValue = analogRead(sensor);
 Serial.print(analogValue);
 if(analogValue>sensorThresh)
 {
  digitalWrite(redled,HIGH);
  digitalWrite(greenled,LOW);
  tone(buzzer,1000,10000);
  lcd.clear();
  lcd.setCursor(0,1);
  lcd.print("ALERT");
  delay(1000);
  lcd.clear();
  lcd.setCursor(0,1);
  lcd.print("EVACUATE");
  delay(1000);
```

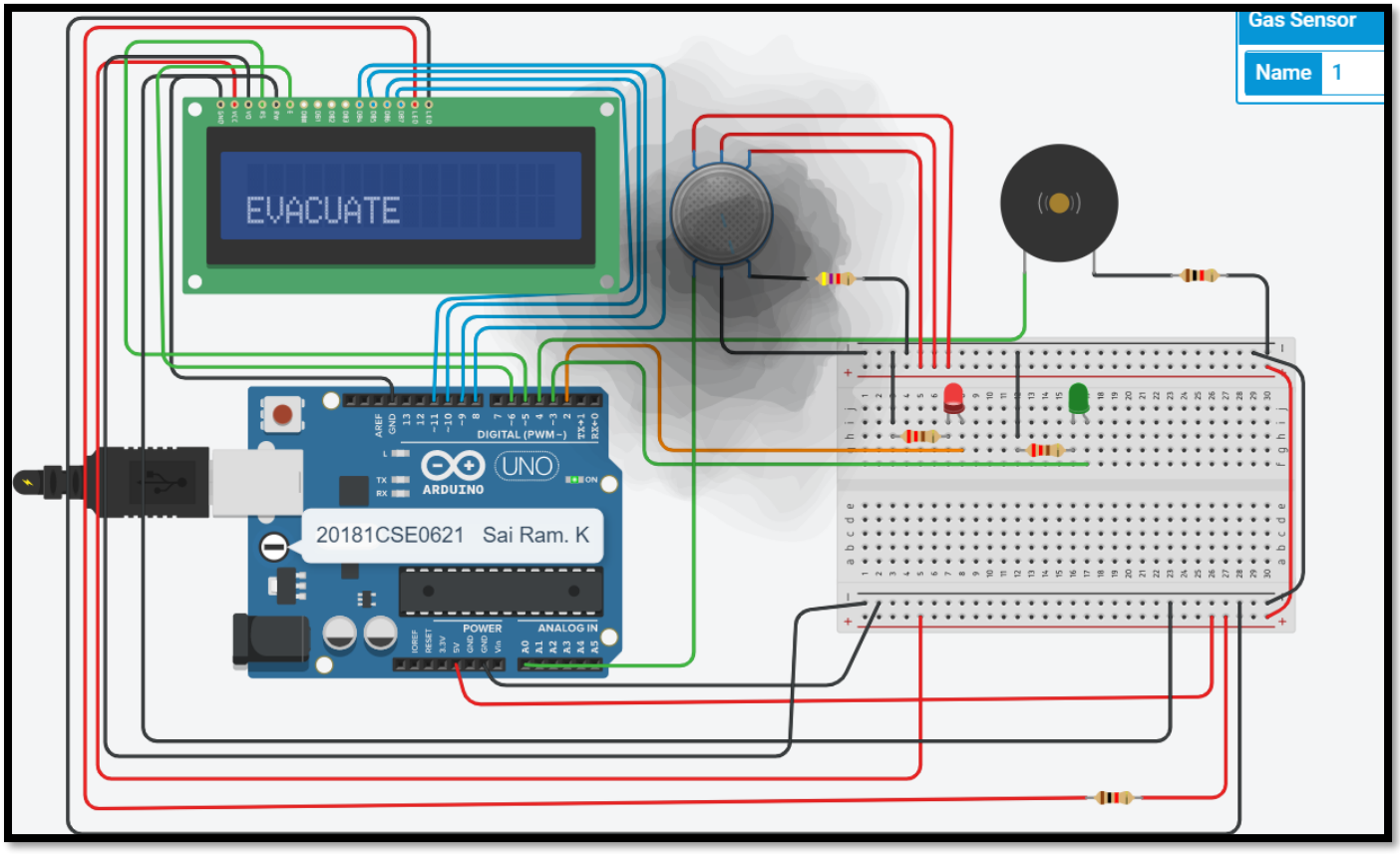
```
 }

 else

 {

  digitalWrite(greenled,HIGH);

  digitalWrite(redled,LOW);

  noTone(buzzer);

  lcd.clear();

  lcd.setCursor(0,0);

  lcd.print("SAFE");

  delay(1000);

  lcd.clear();

  lcd.setCursor(0,1);

  lcd.print("ALL CLEAR");

  delay(1000);

 }

}
```

OUTPUT :

No Gas Leakage :


SAFE
20181CSE0621 Sai Ram. K
UNO
ARDUINO
DIGITAL (PWM~)
POWER
ANALOG IN

Leakage :


Gas Sensor
Name 1
EVACUATE
20181CSE0621 Sai Ram. K
UNO
ARDUINO
DIGITAL (PWM~)
POWER
ANALOG IN

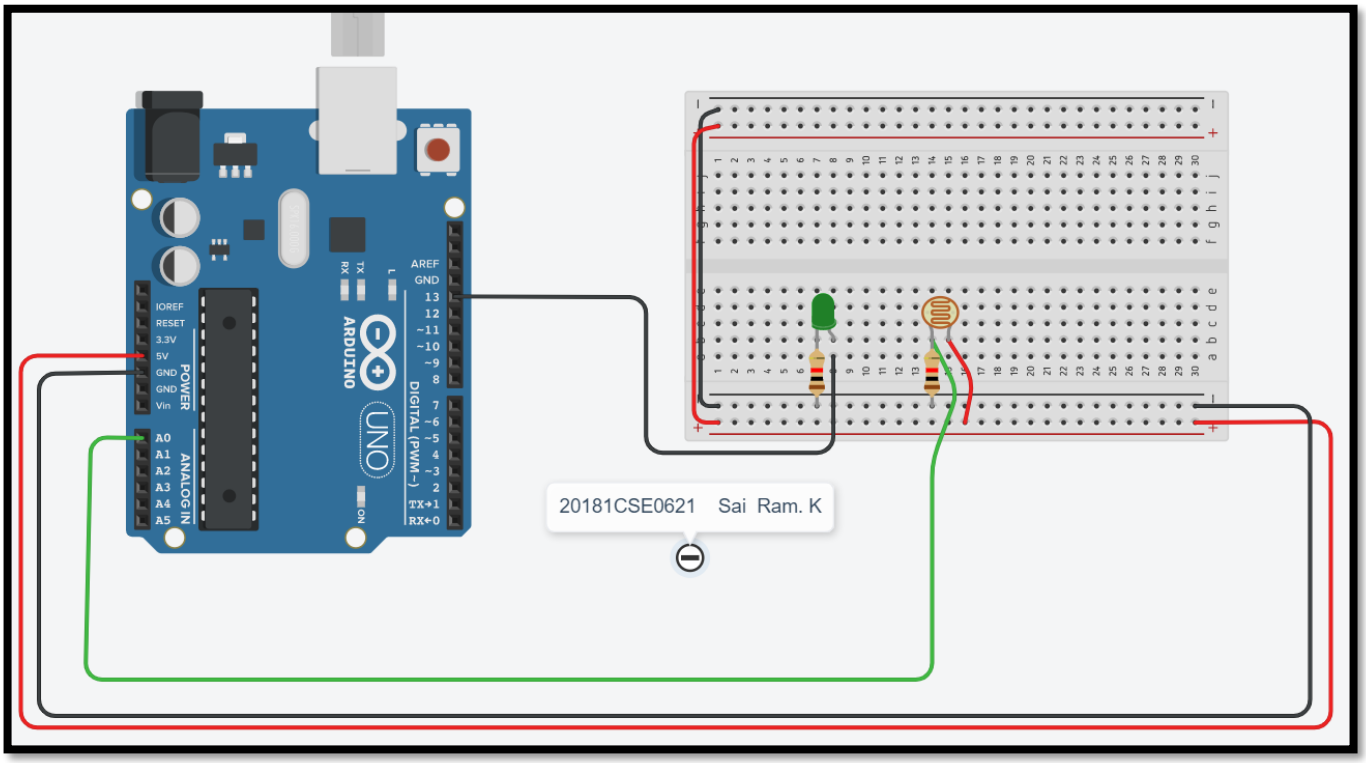
Experiment – 12

Interfacing of LDR using Arudino. Write a program to control intensity of LED using LDR in TinkerCAD

Aim : To control intensity of LED using LDR.

Components : Arduino Uno, Breadboard, Jumper wires, LDR, LED, Resistors.

Initial Circuit Design :

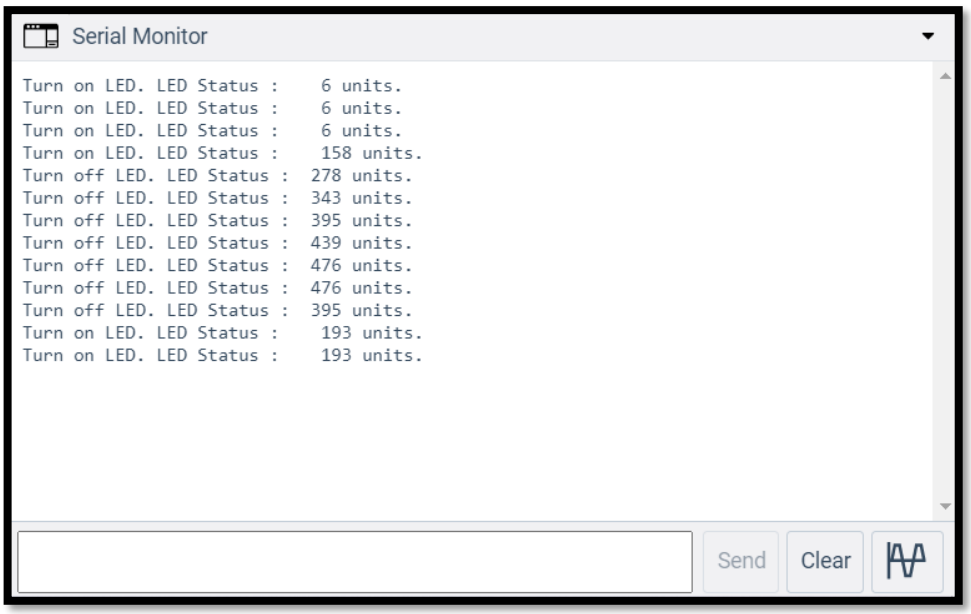


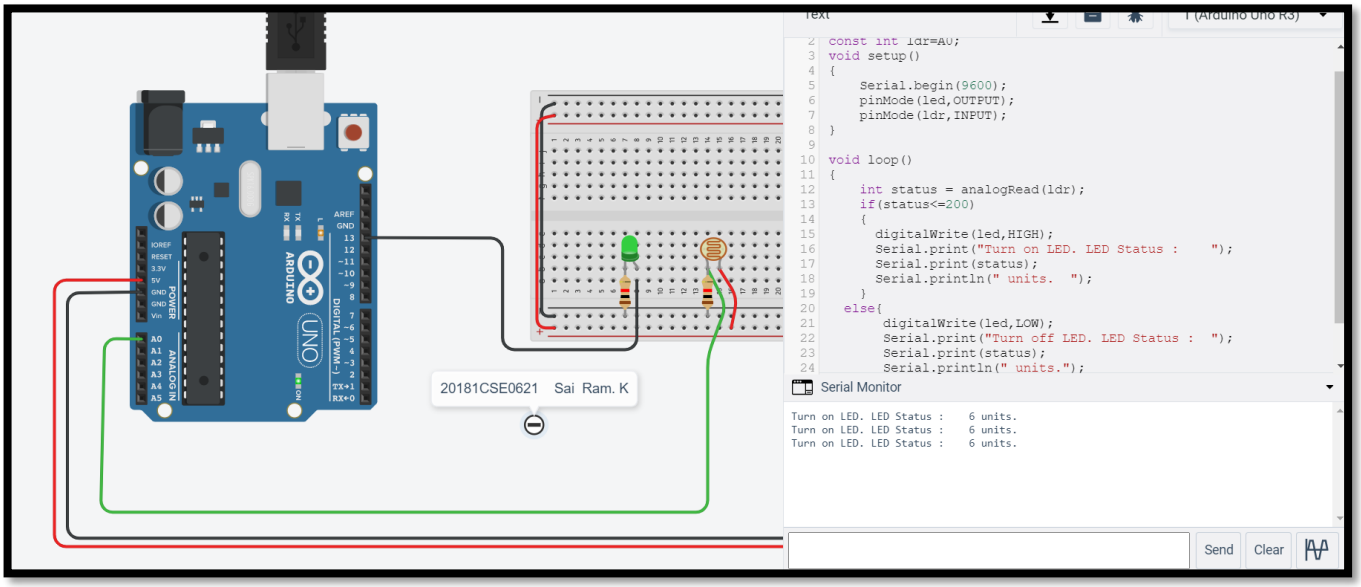
Sketch :

```
const int led=13;

const int ldr=A0;

void setup()

{

Serial.begin(9600);

pinMode(led,OUTPUT);

pinMode(ldr,INPUT);
```

```
}

void loop()

{

int status = analogRead(ldr);

if(status<=200)

{

digitalWrite(led,HIGH);

Serial.print("Turn on LED  ");

Serial.println(status);

}

else{

digitalWrite(led,LOW);

Serial.print("Turn off LED  ");

Serial.println(status);

}

delay(1000);

}
```

OUTPUT :



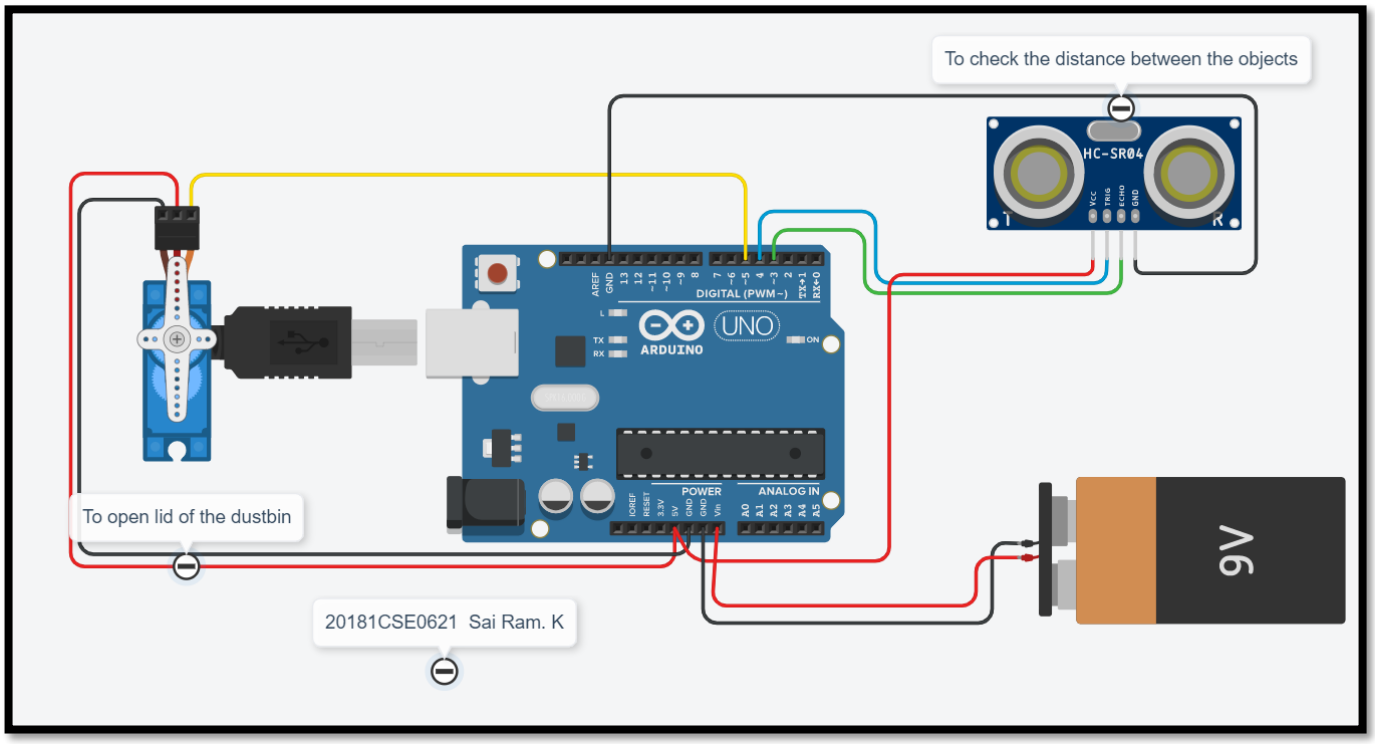

20181CSE0621 Sai Ram. K
Serial Monitor
Turn on LED. LED Status : 6 units.
Turn on LED. LED Status : 6 units.
Turn on LED. LED Status : 6 units.
Send
Clear

Experiment – 15
Smart Dustbin

Aim : To implement a smart dustbin.

Components : Arduino Uno, Breadboard, Jumper wires, micro servo, 9V battery, Ultrasonic Distance sensor

Initial Circuit Design :



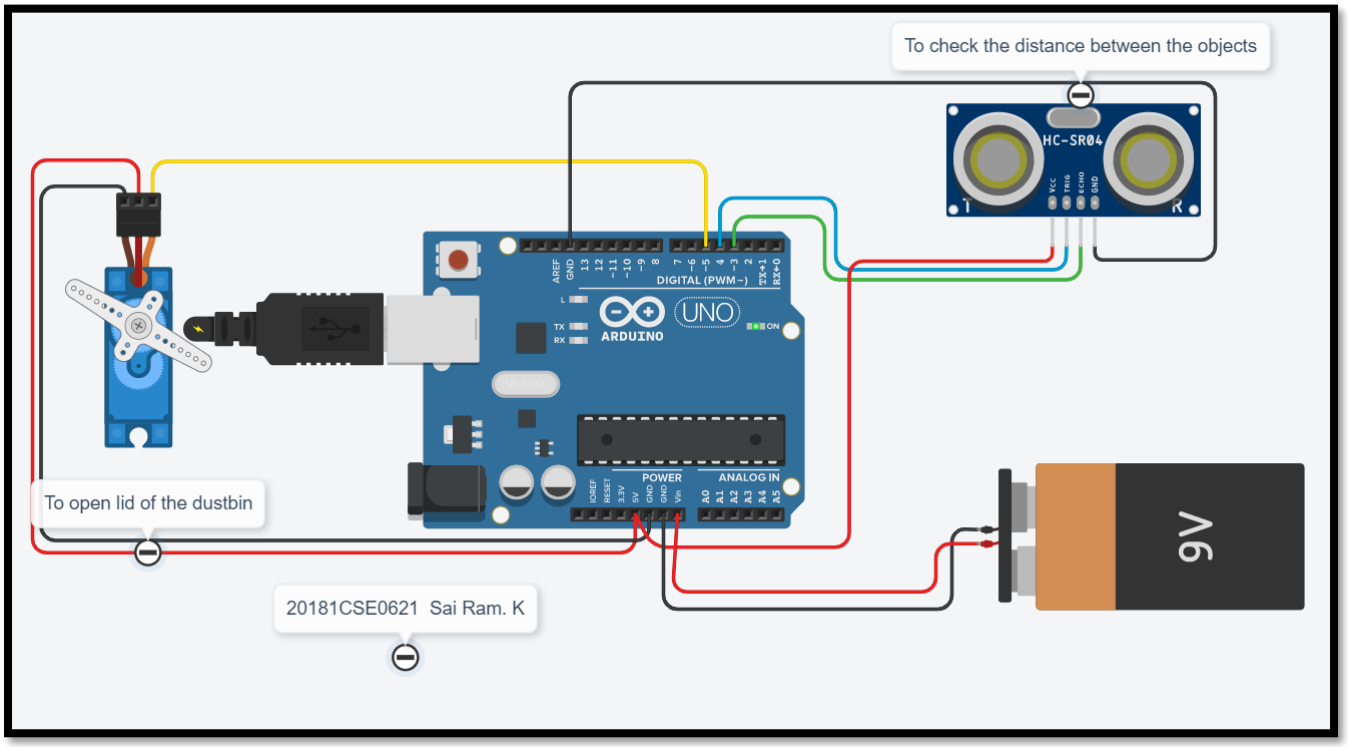
Code:

```
#include <Servo.h>

Servo servoMain;

int trig=4;

int echo=3;

int distance;

float duration;
```

```
float cm;


void setup()
{
servoMain.attach(5);
pinMode(trig,OUTPUT);
pinMode(echo, INPUT);
}
void loop()
{
digitalWrite(trig,LOW);
delay(2);
digitalWrite(trig,1);
delayMicroseconds(10);
digitalWrite(trig,0);
duration=pulseIn(echo,1);
cm = duration/58.82;
distance = cm;
if(distance < 50)
{
servoMain.write(180);
delay(3000);
}
```

```
else

{

servoMain.write(0);

delay(50);

}

}
```

OUTPUT:

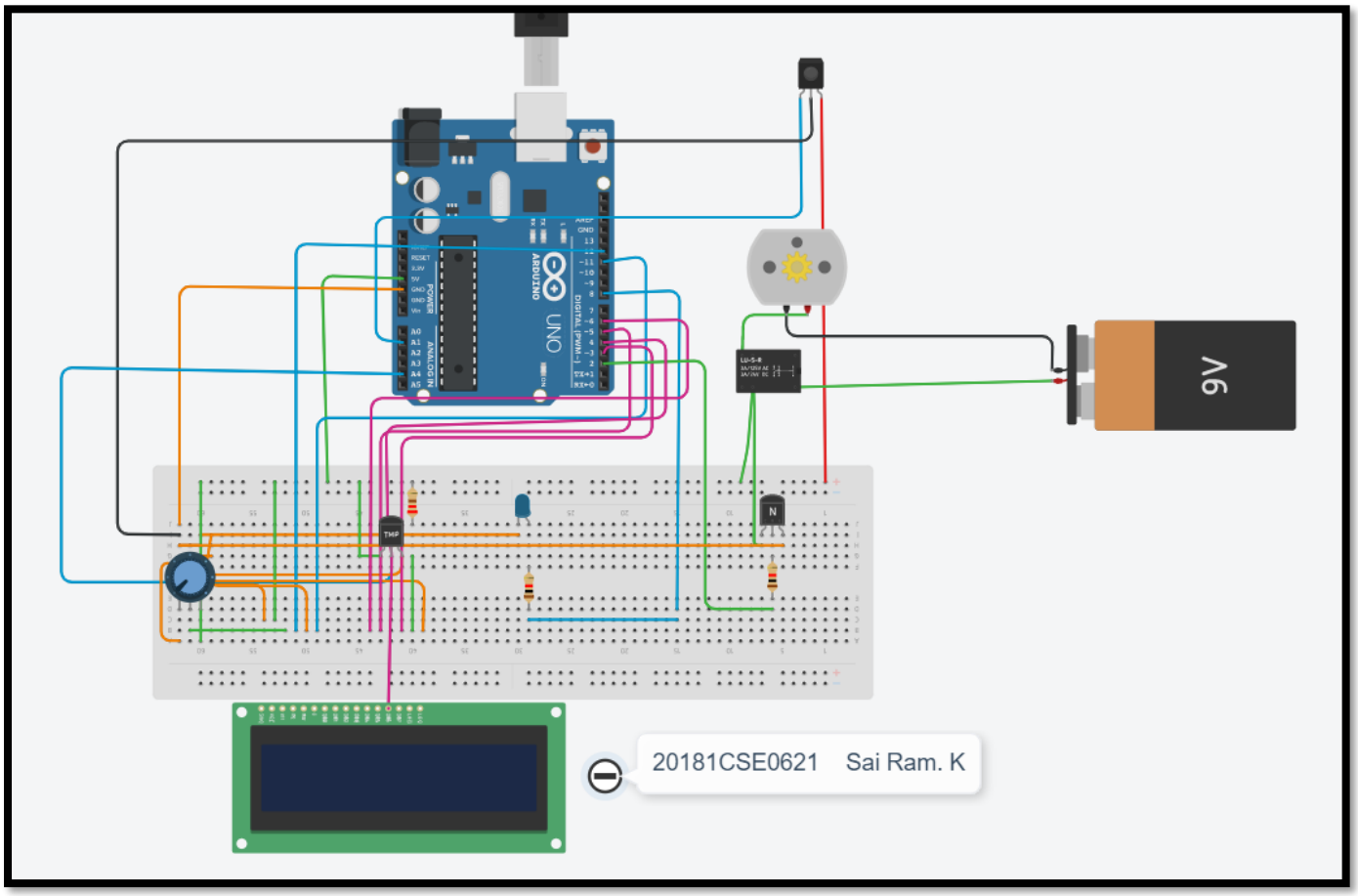


Experiment – 16
Smart Irrigation

Aim : To implement a smart dustbin.

Components : Arduino Uno, Breadboard, Jumper wires, micro servo, 9V battery, relay spdt, potentiometer, led, resistors, lcd, ir sensor, temperature sensor, npn transistors.

Initial Circuit Design :



Code:

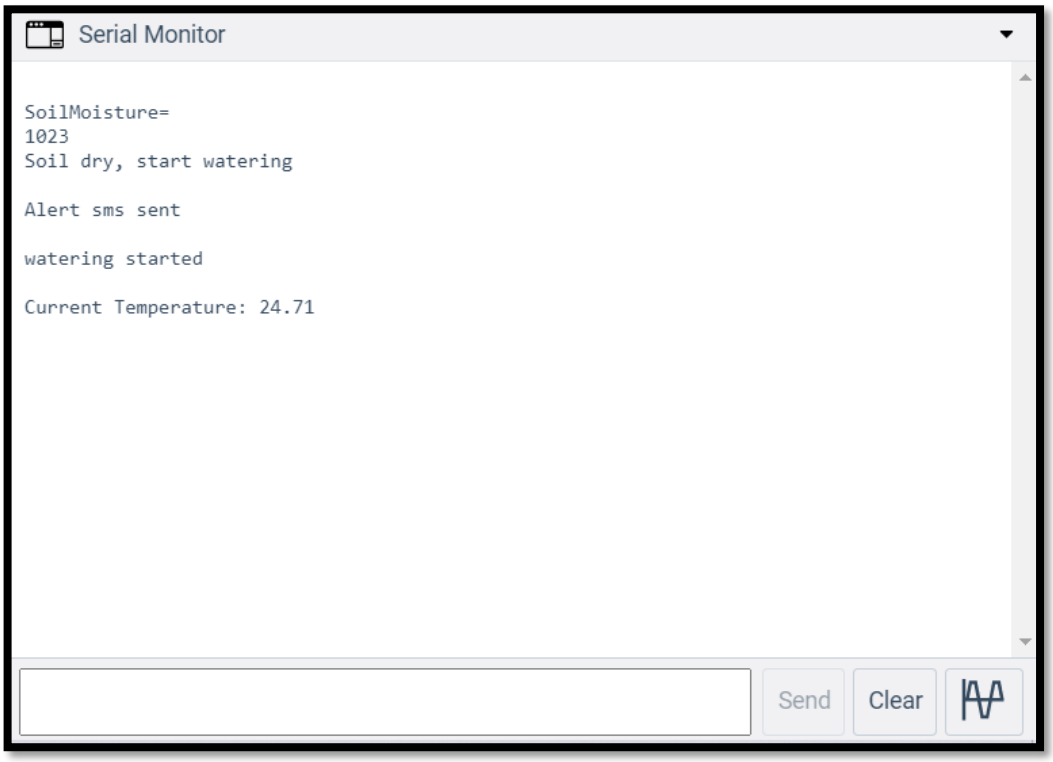
```
#include <SoftwareSerial.h>

#include <LiquidCrystal.h>

#define WATERPIN 2

#define SENSOR A1

#define MAXDRYNESS 500

#define WATERPOSTDELAY 1000

int ch=0;

double temp;

int sensorInput;

LiquidCrystal lcd(12, 11, 6, 5, 4, 3);

void setup()

{

  digitalWrite(WATERPIN, LOW);

  pinMode(WATERPIN, OUTPUT);

  Serial.begin(9600);

  lcd.begin(16,2);

}


void loop() {


 Soil_Moisture();

  sensorInput = analogRead(A4);
```

```
 temp = (double)sensorInput / 1024;

 temp = temp * 5;

 temp = temp - 0.5;

 temp = temp * 100;

 Serial.print("Current Temperature: ");

 lcd.clear();

 Serial.println(temp);

 lcd.print("surr temp ");

 lcd.print(temp);

 delay(2000);

 lcd.setCursor(0,1);

 lcd.clear();

}


void Soil_Moisture()

 {

   int SensorValue = analogRead(SENSOR);

   Serial.print("\nSoilMoisture=\n");

   Serial.print(SensorValue);

  Serial.print("\n");


 if(ch%2==0) //SensorValue >= MAXDRYNESS

 {
```

```
   Serial.println("Soil dry, start watering\n");
   lcd.clear();
   lcd.print("Soil dry");
   lcd.setCursor(0,1);
   lcd.print("alert sms sent");
   Serial.println("Alert sms sent\n");
   delay(2000);
   lcd.clear();
   lcd.print("Motor is On");
   lcd.setCursor(0,1);
   lcd.print("watering started");
   Serial.println("watering started\n");
   digitalWrite(8, HIGH);
  digitalWrite(2, HIGH);
      delay(8000);
    digitalWrite(WATERPIN, LOW);
   ch++;
    lcd.clear();

 }
  else //SensorValue < MAXDRYNESS
  {
    Serial.println("\nSoil is good\n");
```

```
    Serial.println("Motor is Off\n");
    lcd.print("Soil is good");
    delay(3000);
    lcd.clear();
    lcd.print("Motor is Off");
    delay(3000);
    lcd.clear();
    digitalWrite(8,LOW);
    delay(WATERPOSTDELAY);
    ch++;
  }
   delay(2000);
 }
```

OUTPUT:



Experiment – 17

Raspberry Pi Installation

Aim: To install Rasbian OS on local device

Equipment's Needed

Raspberry Pi, VGA cable, Power Cable

Implementation:

Introduction: The Raspberry Pi is a fully-fledged minicomputer, capable of doing whatever you might do with a computer. It comes with 4x USB, HDMI, LAN, built-in Bluetooth/Wi-Fi support, 1GB RAM, 1.2GHz quad-core ARM CPU, 40 GPIO (General Purpose Input Output) pins, audio and composite video output, and more.

One can use Raspberry Pis as home security cameras, server monitoring devices, cheap headless machines (basically running low-weight scripts 24/7 with a low cost-to-me) ... others have used them for media centres and even for voice-enabled IoT devices. The possibilities are endless, but first we need to get acquainted!

Make sure that, if you do get a case, it has openings for the GPIO pins to be connected, you will also need a 1000mA+ mini usb power supply and at least an 8GB micro-SD card, but suggested is 16 GB micro-SD card or greater.

You will also want to have a spare monitor (HDMI), keyboard, and mouse handy to make things easier when first setting up. You won't will eventually be able to control your Pi remotely, so you won't always need a separate keyboard, mouse, and monitor. If you don't have a monitor with HDMI input, you can buy something like an HDMI to DVI converter.

If you're using an older version board, please see what you might need to change, for example, the older Raspberry Pis take a full-sized SD card, but the latest model requires a micro SD card. Also, the Raspberry Pi 3 Model B has built-in wifi, where the older models will require a wifi dongle.

A typical Raspberry Pi shopping list, assuming you have a mouse, keyboard, and HDMI monitor that you can use temporarily while setting up is:

1.Raspberry Pi -

2.1000mA+ mini usb power supply -

3.16 GB micro SD card –

Additionally, if you plan to join us on the initial GPIO (General Purpose Input Output pins) tutorials, you will also want to pick up:

1.10 x Male-to-Female jumper wires (you should consider just buying a bunch of these so you have plenty in the future).

2.1 x Breadboard (You may also want multiples of these)

3.3 x LED light (...more wouldn't hurt)

4.~6 x Resistors (between 300 and 1K Ohm). You will need at least 1K and 2K ohms for the distance sensor, then ~300-1K resistance per LED bulb. You probably should just buy a kit, they're super cheap.

5.1 x HC-SR04 Ultrasonic Distance Sensor (...you know what I'm going to say...think about maybe a few.)

6.1 x Raspberry Pi camera module. You only need one of these!

For the jumpers, breadboard, and leds, you could also just buy a kit, something like: this GPIO starter kit.

There are also a few ways to install and use an operating system on the Raspberry Pi. The most user-friendly method is to use the NOOBS (New Out of Box Software) installer. If you're comfortable enough, you can just simply download the operating system ISO, format the SD card, mount the ISO, and boot the Pi. If that sounds like gibberish to you, then follow along with the NOOBS installation option.

While we're working with the SD card, let's go ahead and Download NOOBS, which is just over 1GB.

First, we must format the SD card. If you are on Windows, you can use SD Formatter. Mac users can also use SD Formatter, but they have a built in formatter, and Linux users can use GParted. Whatever the case, you need to format the SD card, do not do a "quick format" and do make sure you have the "resize" option on. Using SDFormatter on Windows, and chosing options:

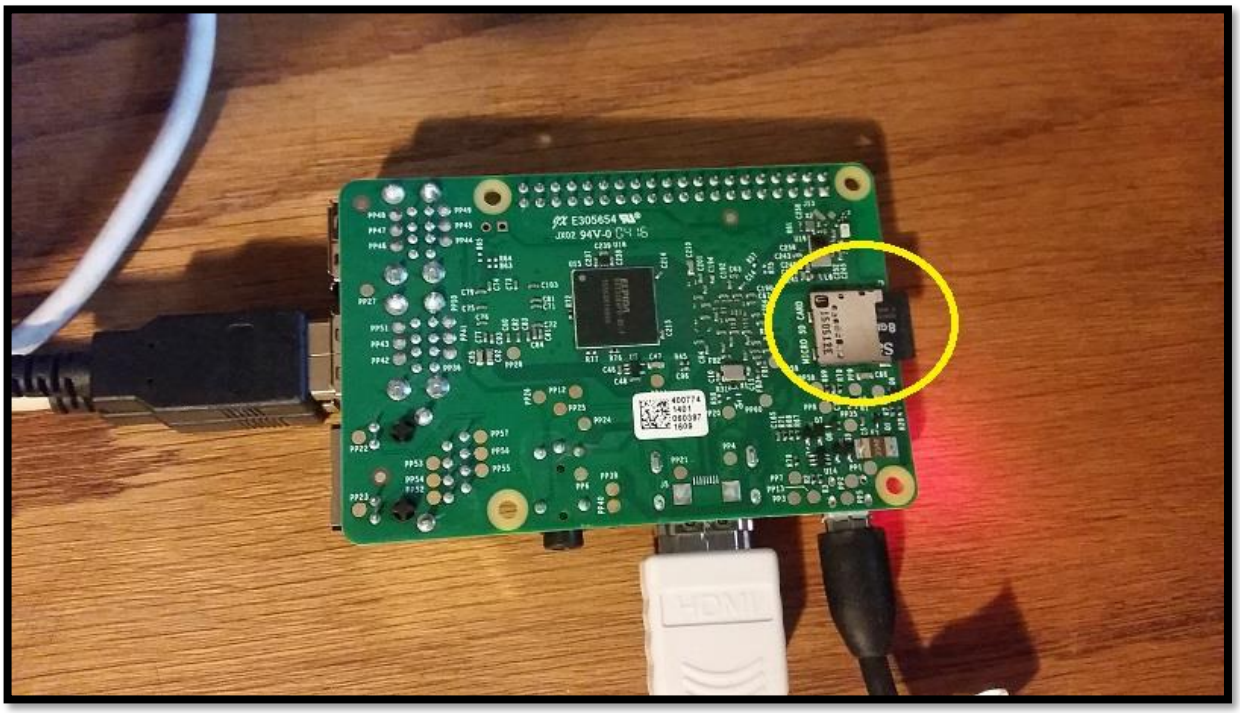
This should go without saying, but do make sure you're formatting the right drive. This will format any flash drive, in alphabetical order. If you had something plugged in already, like your favorite USB drive, and forgot about it, that'd likely be the default choice to format, and then you'd spend all afternoon trying to recover your data rather than enjoying playing with your Raspberry Pi.

Now, assuming you've downloaded the NOOBS package, let's go ahead and extract that. Now, we want to copy all these NOOBS contents to our SD Card. Do not drag the directory, but rather the contents:

While that's transferring, let's talk about a few things on the actual Raspberry Pi board:

The GPIO (General Purpose Input/Output) pins are underlined in blue. We can use these to control peripheral devices like motors, servos, and more. Circled in red is the micro usb power input for the board. In orange, the HDMI output port. The yellow is where you can plug in the Raspberry Pi camera module. The grey circle has the USB ports. This is obviously not everything, but these are the main things to note.

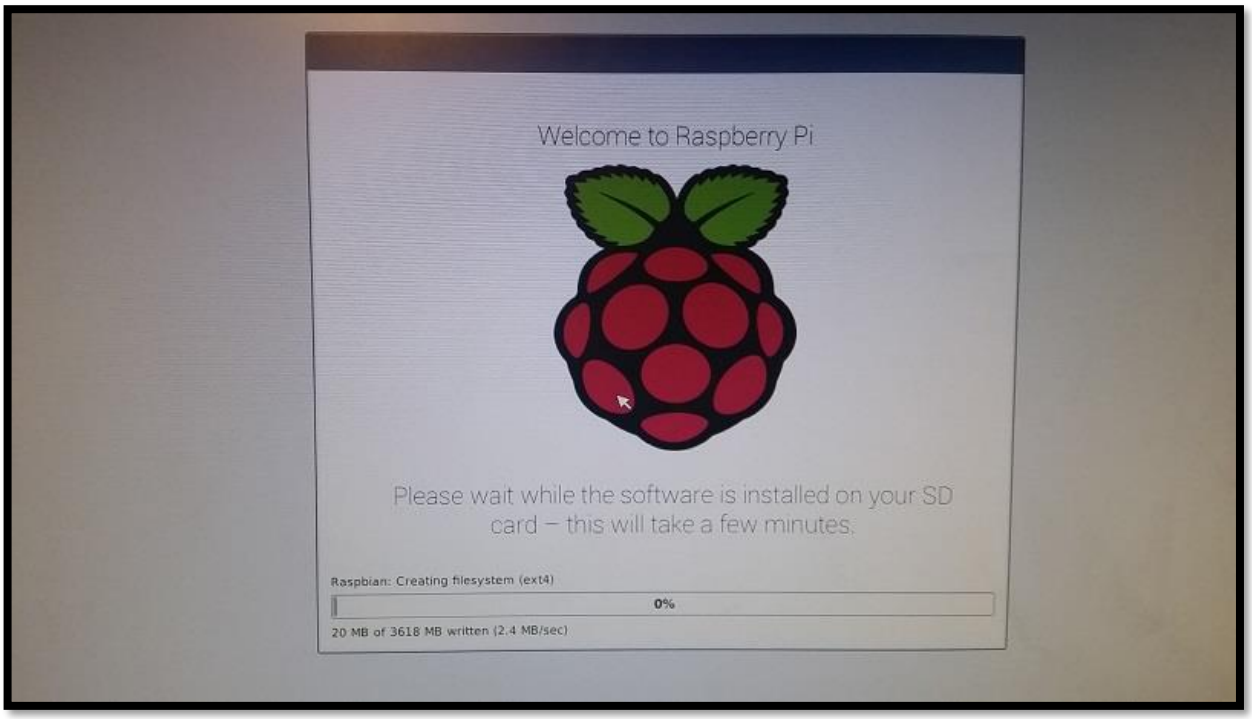
Once everything is transferred to the micro SD card, you can put it in the Raspberry Pi. The slot is on the bottom side of the board, circled in J yellow here:



~

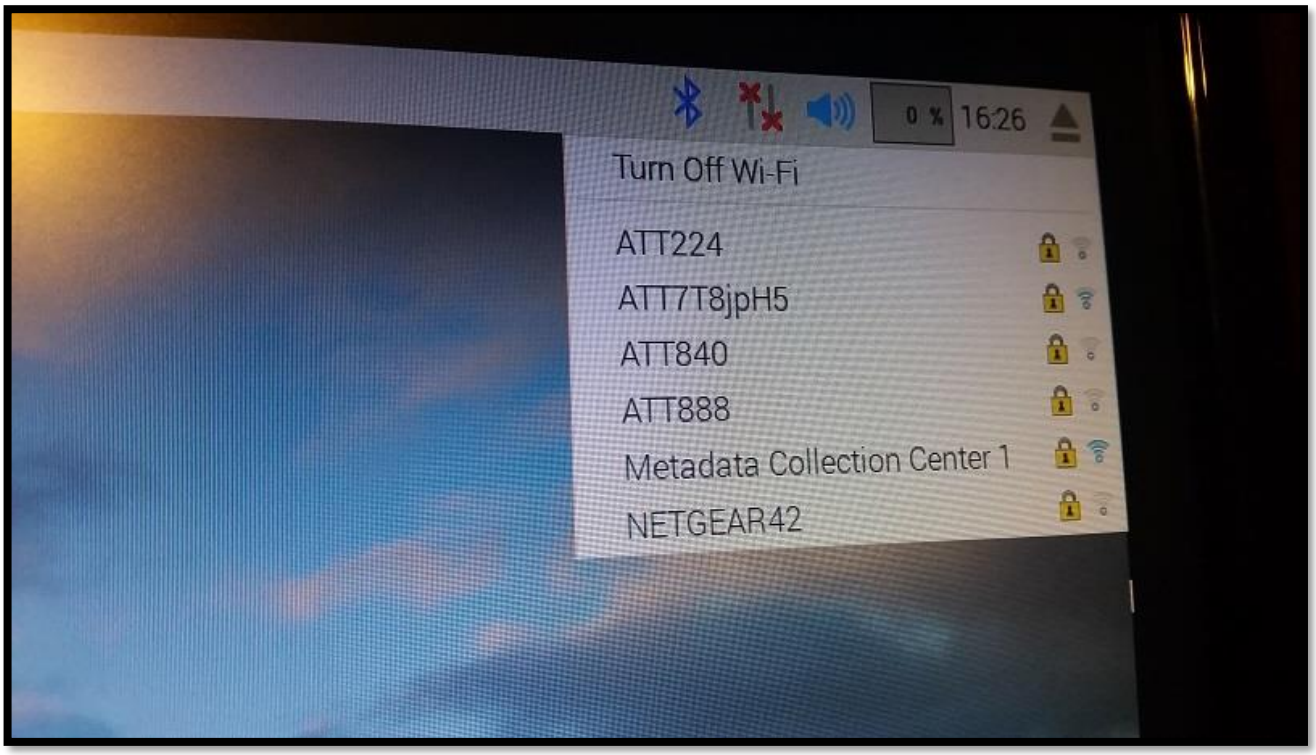
~

Let the process go, this will take a while, something like 20-30 minutes or so.



sa

Once that's done, hit okay and the device should reboot to desktop. While on the desktop, wait for a moment for wifi to start up and find available connections. Connect to your wifi network if possible. You can also plug directly in with an ethernet cable if you don't have wifi. You can also just continue interacting directly with the Raspberry Pi with the mouse and keyboard connected to it if you like, but I prefer to access it remotely.



Once we've connected to our network, we'd like to actually interface with the Pi. First, we want to update. Open a terminal by either right clicking on the desktop and opening terminal that way, or by doing control+alt+t. Now, in the terminal, do:

$sudo apt-get update

and then

$ sudo apt-get upgrade

You do not type the $ sign, it's there to denote when you're typing something in the command line. The upgrade might take a minute. While we wait, your Raspberry Pi's default credentials are: username: pi password: raspberry. For some reason, the apt-get upgrade for me was taking absurdly long. You need to be connected to your network, and have internet access, so make sure you have those things first before doing this, but still was having trouble. One can solve this by doing:

$ sudo nano /etc/apt/sources.list

Then replace everything here with:

deb http://archive.raspbian.org/raspbian jessie main contrib non-free

deb-src http://archive.raspbian.org/raspbian jessie main contrib non-free

control+x, y, enter

$ sudo apt-get dist-upgrade

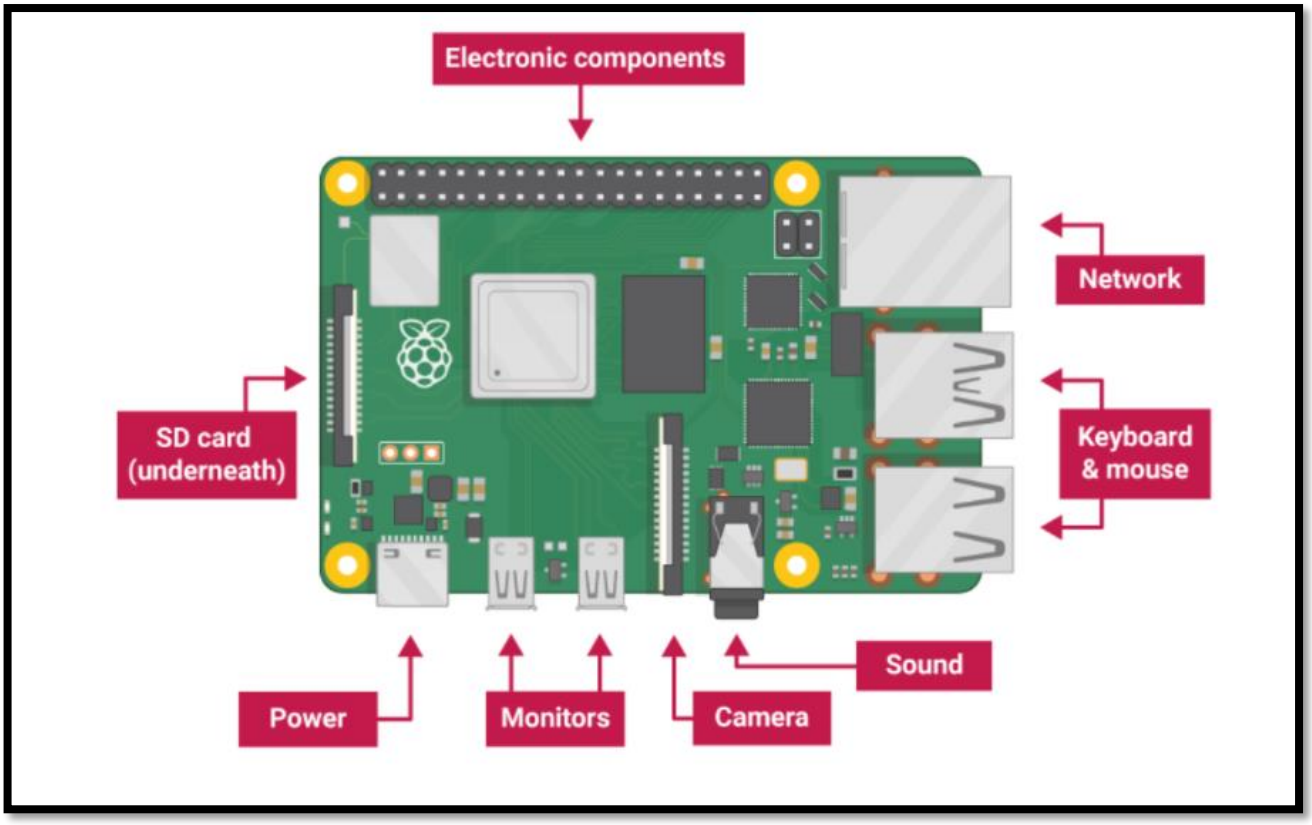
$ sudo apt-get update

$ sudo apt-get upgrade

To save some space you can also do: $ sudo apt-get purge wolfram-engine and then $ sudo apt-get autoremove. This alone freed up almost 700mb of space for me.

This is going to conclude the first part of this tutorial series. In the next tutorial, we're going to cover how we can remotely access our Raspberry Pi.

Experiment – 18
Raspberry Pi Commands



The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games.

Features of Raspberry PI

· 512 MB SDRAM memory

· Broadcom BCM2835 SoC full high definition multimedia processor

· Dual Core Video Core IV Multimedia coprocessor

· Single 2.0 USB connector

· HDMI (rev 1.3 and 1.4) Composite RCA (PAL & NTSC) Video Out

· 3.5 MM Jack, HDMI Audio Out

· MMC, SD, SDIO Card slot on board storage

· Linux Operating system

· Dimensions are 8.6cm*5.4cm*1.7cm

· On board 10/100 Ethernet RJ45 jack

Basic Commands

Cd – change directory
Cat – shows all the file contents
Ls – list of all files and folders
Locate – search for a file
Lsusb – list of usb devices
Pwd – print working directory
Mkdir – create new directory
Mv – move the file
Rm – removing the file
Reboot – reboot the device
Shutdown – shutdown the device
Grep – print the lines matching a patern
Df – disk usage space
Ifconfig – config the network interface
Netstat – print all network connections

Commands used in rasbian OS

1. PWD (Print Working Directory) This tells you your current working directory.

2. CD (Change Directory) This commands is used to move to other directory. If you are on home and you want to move to pictures folders which is in Desktop folder, then you should do as shown below. If you want to move back a directory, then write "cd .." and to go to the home directory, just write "cd"

3. LS (list) This command lists the content of the current working directory. This command when used with "-l" will list the files and folders in the current directory along with the file size, date modified and permissions. Try "ls –help" and have a look at the other options.

4. MKDIR (Make Directory) This command is used to make a new directory.

5. RMDIR (Remove Directory) This command is used to remove an empty directory.

6. RM (Remove) If you want to remove a file (Both empty and not empty), then use this command. If used with –r will remove the file or even folder permanently.

7. Touch If you want to create a new file, then use this command.

8. NANO If you want to edit the file, then use this command.

9. CP (Copy) This command is used to copy a file or a folder from one location to the other.

If you want to copy the whole folder, then use it with "-r".

10. MV (Move) This command will cut the file or folder from one location and will paste it to the other location. To cut and paste a file, follow the below image. If you want to cut and paste the folder. The folder must be empty.

Networking Commands

1. Ping This command is used to check that whether communication can be made with another host or not.

2. Nmap This command is used for network exploration and scanning. It can return port and OS information about a host or a range of hosts. Type "nmap" to check out what it can do.

3. Hostname This command tells you your hostname

4. Ifconfig This command will tell you the network configuration details such as your IP address.

5. Iwconfig This command is used to check which network you are using. It will tell the network name and its details.

6. Iwlist wlan0 scan This command will print the currently available wireless networks.

General Commands

1. Sudo This command allows you to run a command as a superuser or another user. Simply type "sudo" to check what it can do.

2. Apt-get This command is used to install, upgrade and remove the packages. Simply type "apt-get" to check what it can do.

3. Find This command is used to find the files or directories.

4. Raspi-config This command will open the configuration settings menu.

5. Startx This command will open the Graphical User Interface.

6. Reboot This command will reboot the system immediately.

7. Shutdown To shutdown at specific time

Experiment – 19
Raspberry pi-camera module

Aim : To implement the camera module in Raspberry Pi.

Components:

Raspberry Pi, Raspberry Pi Camera Module, VGA cable, Power Cable.

Implementation:

Step 1:Open the notch “Connect camera” module to raspberry pi Blue should be facing towards Ethernet port.

Step 2:

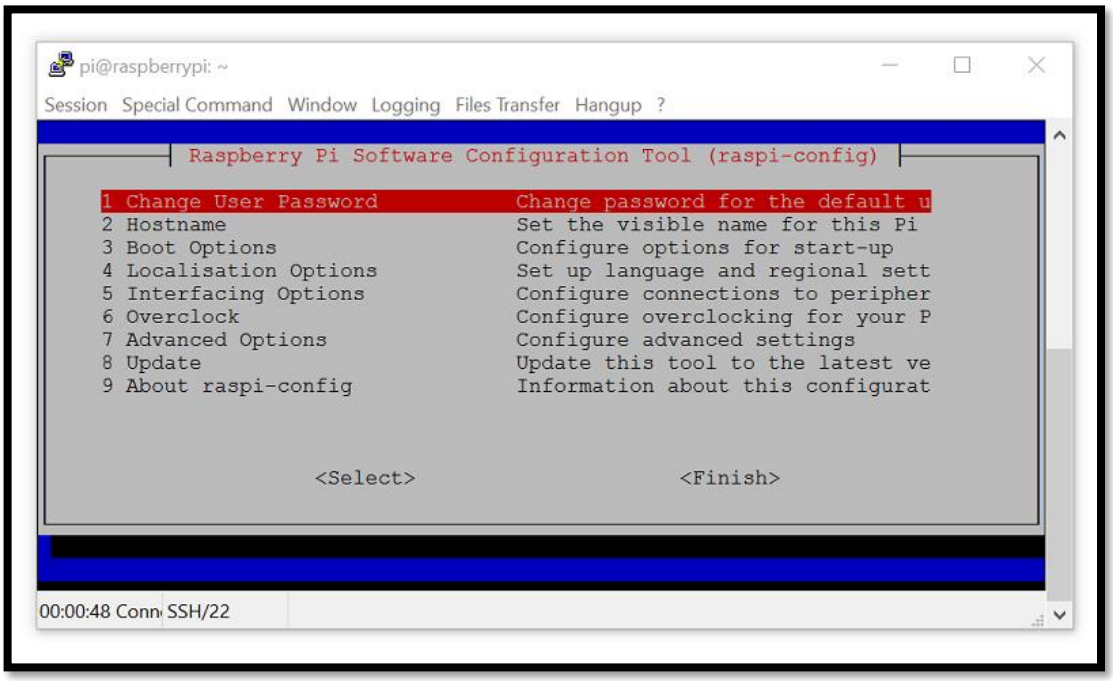
pi@raspberrypi:~ $cd Desktop/

Step 3:

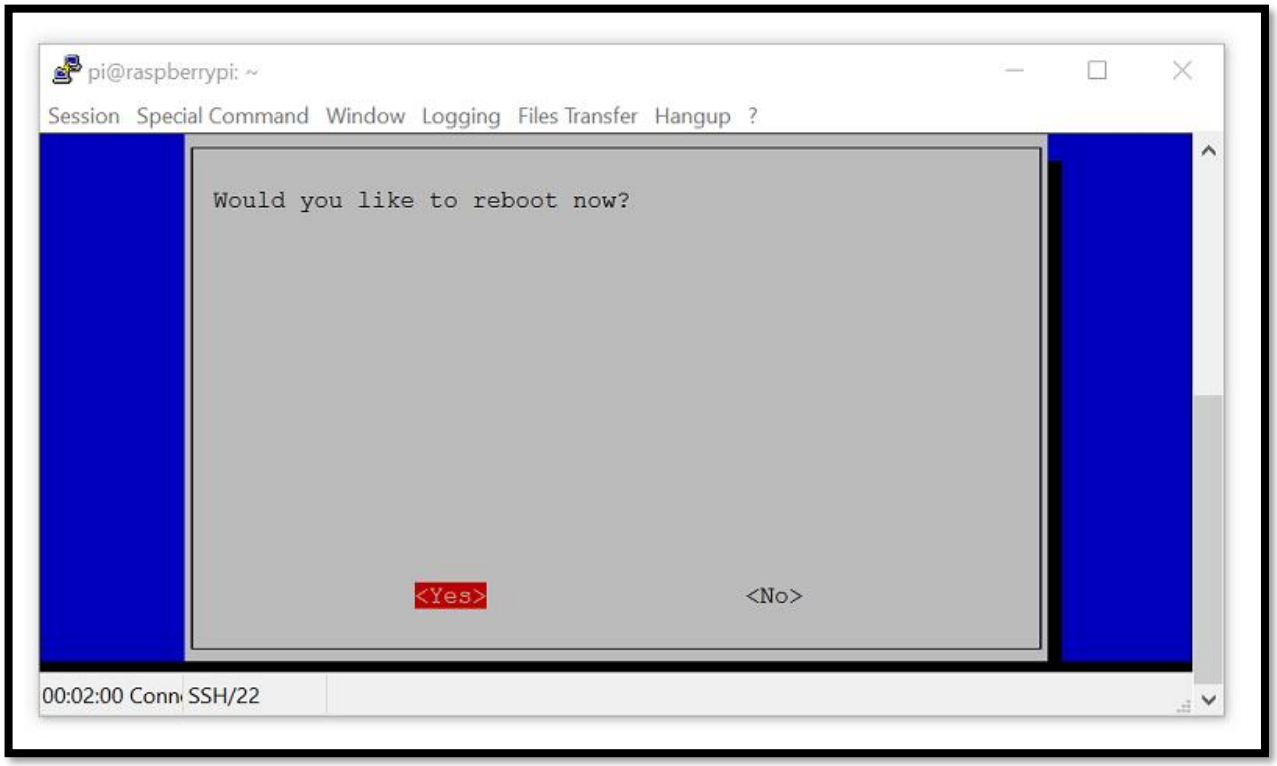
Enable Camera

pi@raspberrypi:~ /Desktop$ sudo raspi-config

Interfacing option Enable camera



Yes to enable, and then go ahead and reboot:



Step4:

Let's open the terminal next (control+alt+t) and do a $ sudo apt-get install python3-picamera. You will likely find that you already have it, but we want to be sure.

Now, in our cameraexample.py file:

Step 5:

Write the Phython program

pi@raspberrypi:~ /Desktop$ nano cameraexample.py

import picamera

import time

camera = picamera.PiCamera()

camera.capture('example.jpg')

```
camera.vflip = True

camera.capture('example2.jpg')

camera.start_recording('examplevid.h264')

time.sleep(5)

camera.stop_recording()
```

Step 6:

Command to view recorded video

$ omxplayer examplevid.h264.

Experiment – 20
Raspberry pi Remote Login

Aim : To implement the camera module in Raspberry Pi.

Components: Raspberry Pi, VGA cable.

Implementation:

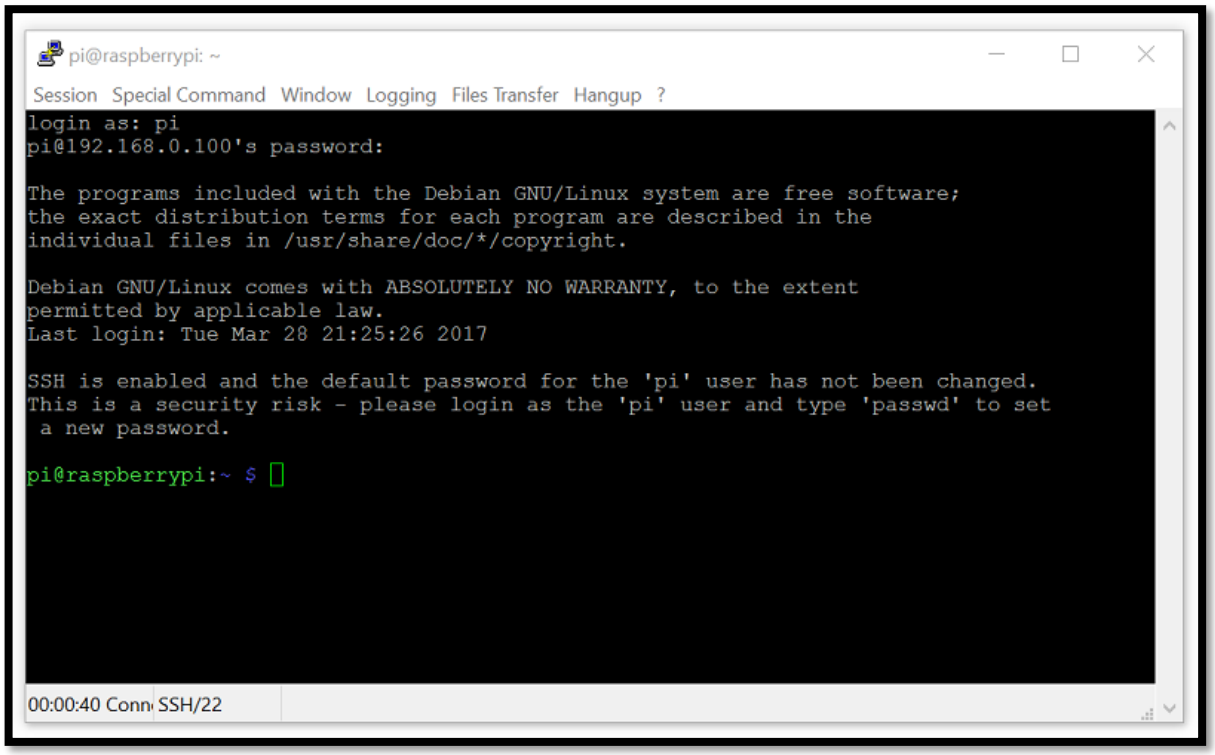
Introduction :

In this Experiment, we're going to cover how we can remotely access our Raspberry Pi, both with SSH and with a remote desktop client. We want to eventually be able to remotely access our Raspberry Pi because much of the "value" of the Raspberry Pi is its size, and that it can be put in a variety of places that we might not want to have a keyboard, mouse, and monitor attached to it at all times and we probably don't want to have to carry over all of this stuff when we do want to access it.

First, let's connect via shell (SSH). Open the terminal on the Raspberry Pi (control+alt+t), and type ifconfig. If you're connected via wifi, then go under the wlan section, and look for your inet address. This will be your local ip, something like 192.168.XX.XXX. We can use this to connect via SSH (user: pi, pass: raspberry), BUT we first have to enable the SSH server. To do this, type sudo raspi-config. Here, we can do quite a few things, but let's head into option #5 interfacing options, next choose the 2nd option forSSH and enable the server. Once this is done, you can shell into the Raspberry Pi.

On Windows, you will need to use an SSH client. But the recommended is PuTTY . Once downloaded, you can open PuTTy, fill in "host name" field with your Pi's local ip, hit enter, and then you will be asked for a username and password.

When successful, you should have something like:



This is identical to the terminal you accessed earlier from the Pi's desktop.

Now, sometimes, you might really want to get access to the desktop instead of just the terminal. To do this, you need the "host" to have remote desktop capabilities, as well as whatever remote PC you attempt to use to access it. First, let's deal with the Raspberry Pi. We're going to install xrdp.

sudo apt-get remove xrdp vnc4server tightvncserver

seudo apt-get update

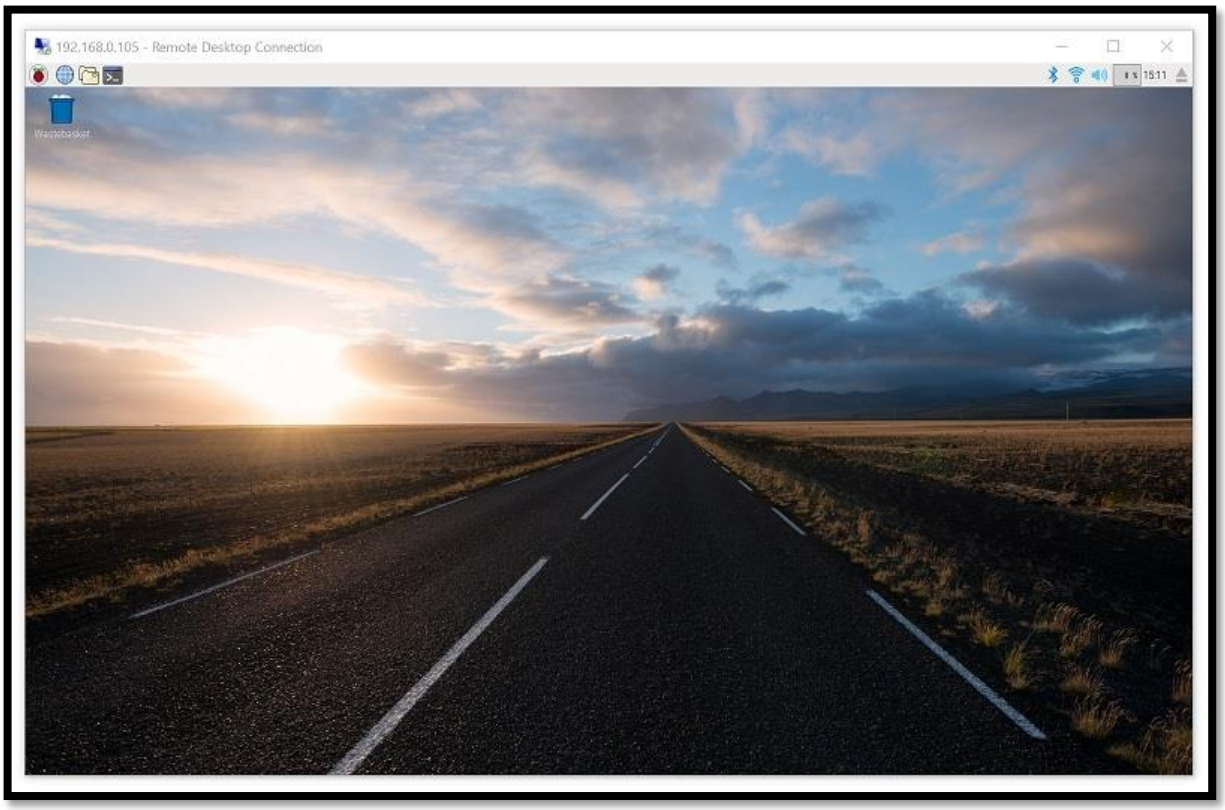
sudo apt-get install xrdp

sudo apt-get install tightvncserver

Now let's deal with the computer from which you plan to connect your Raspberry Pi.

For Mac and Windows, you can use the Microsoft application called Remote Desktop. On linux, you can use grdesktop (sudo apt-get install grdesktop). All three of these examples are going to act the same way. Run them, fill in the IP address of the Raspberry pi, the username, the password, and connect!

When done, you should have something like:



With the SSH server turned on, you will likely be seeing warnings every time you log in that your password is still the default password. The SSH server is turned off by default because, as the Pi has gained popularity, people who might not realize the security risk are using it in places like their homes and businesses. All it takes is someone to connect to your wifi, scan for other local ips, and try them all with default usernames and passwords for various devices, like the Raspberry Pi, and boom they're in. If you want to change your password, you can do sudo raspi-config, and it's the first option.