VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"GnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

INTERNET OF THINGS

Submitted by

IMADH AJAZ BANDAY (1BM20CS059)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019

Oct 2022-Feb 2023

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "INTERNET OF THINGS"carried out bY **IMADH AJAZ BANDAY (1BM20CS059),** who is a bonafide student of B. M. S. College of Engineering. It is in partial fulfillment for the award of Bachelor of Engineering in **Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022-23. The Lab report has been approved as it satisfies the academic requirements in respect of Internet of Things Lab - (20CS5PEIOT) work prescribed for the said degree.

Dr.K.Panimozhi
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE,
Bengaluru

Program no: **01** Program Title: **LED BLINK**

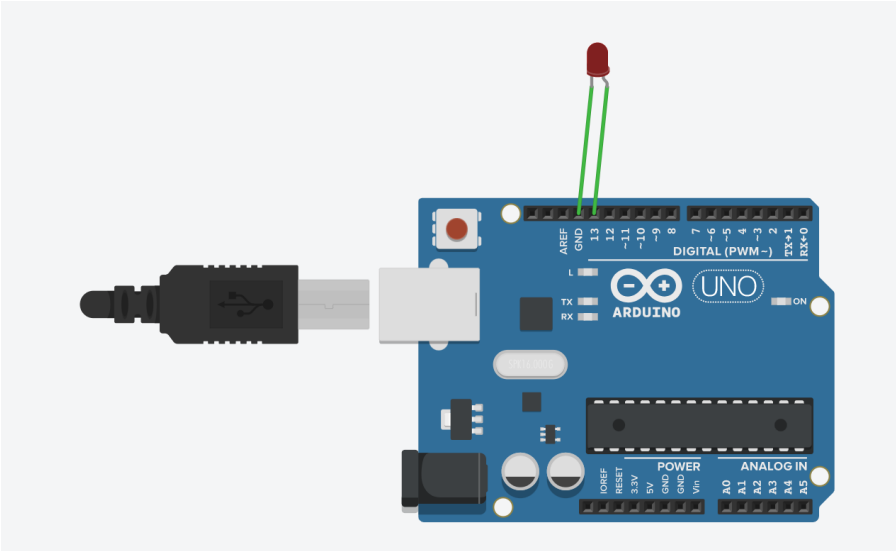
Aim: To control the LED using Arduino (to turn ON/OFF LED)

Hardware/components Required:

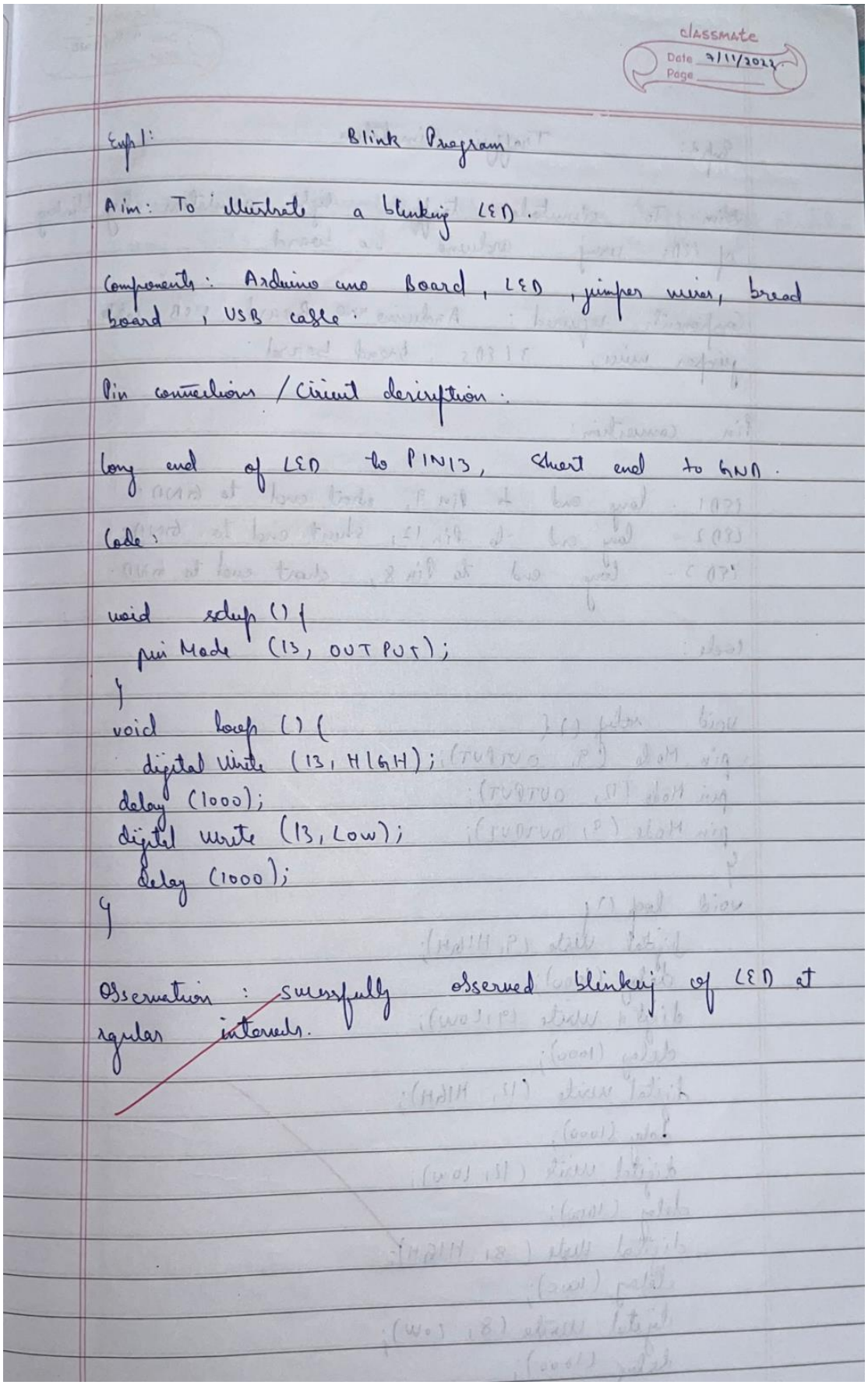
- Arduino Uno board - 1
- USB Cable - 1
- LED - 1
- Jumper wires

Circuit Diagram / Pin connection:

- LED's positive leg is connected to digital pin 13.
- LED's negative leg is connected to ground.



Handwritten code pic:



Expt: Blink Program

Aim: To illustrate a blinking LED.

Components: Arduino uno Board, LED, jumper wires, bread board, USB cable.

Pin connections / Circuit description.

long end of LED to PIN13, short end to GND.

Code:

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

Observation: successfully observed blinking of LED at regular intervals.

Code:

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

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

Observation: LED switches ON/OFF periodically. Digital output visualization using Arduino Uno.

Program no: **02** Program Title: **Traffic Light Simulation**

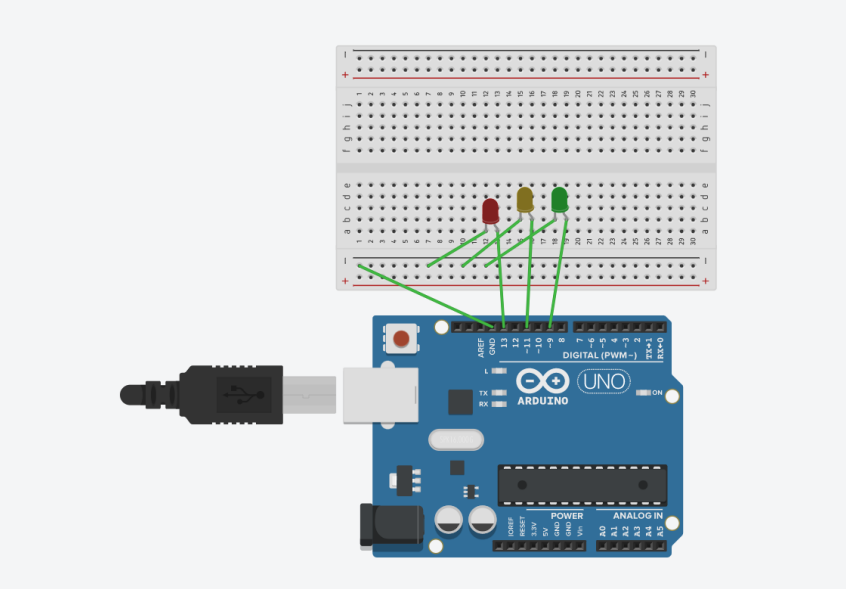
Aim: To simulate traffic lights by blinking of LEDs using Arduino Uno board

Hardware/components Required:

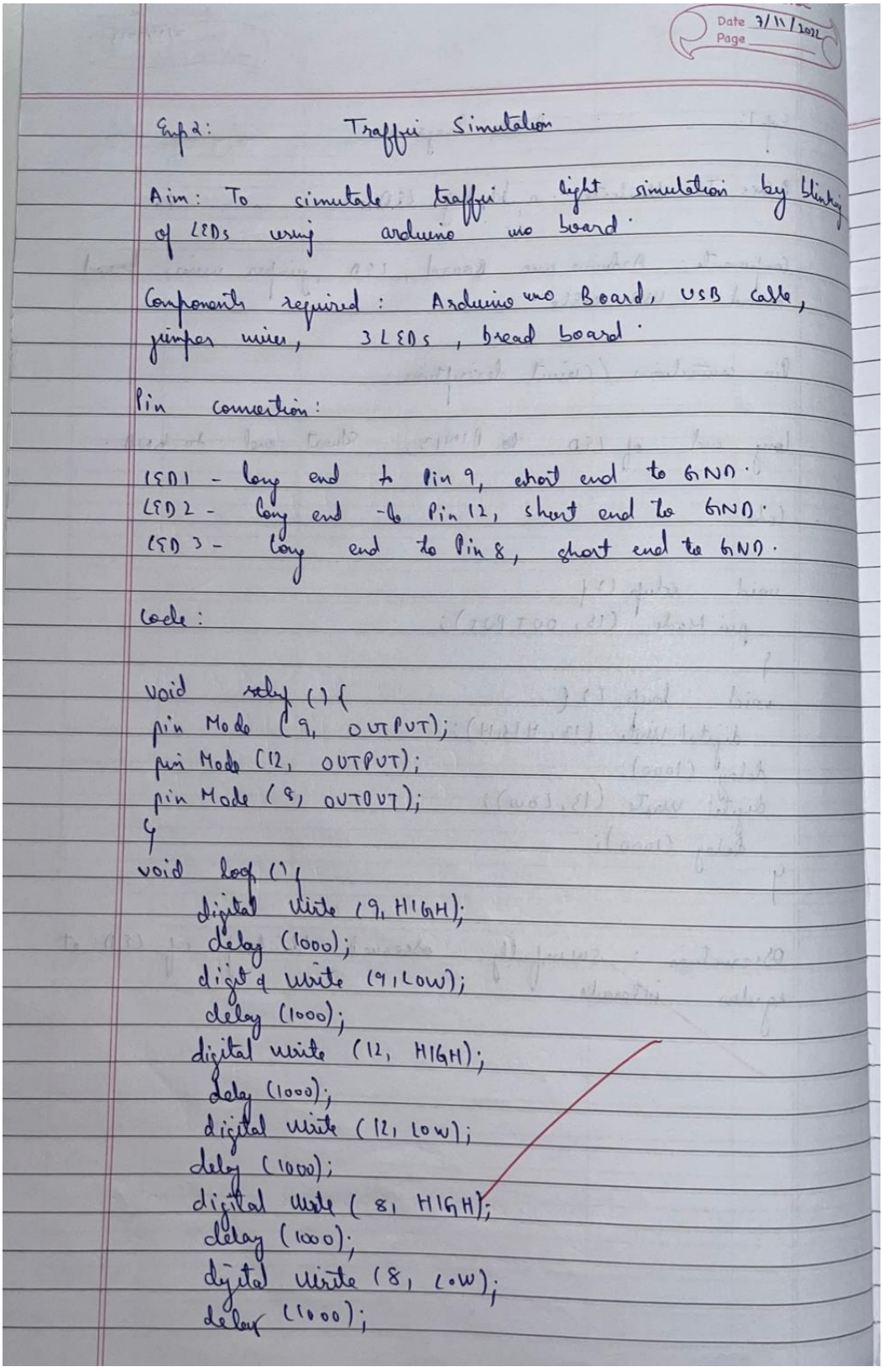
- Arduino Uno board - 1
- USB Cable - 1
- LEDs - 3
- Jumper wires
- Bread Board

Circuit Diagram / Pin connection:

- LED 1's positive leg is connected to digital pin 13, negative end to ground.
- LED 2's positive leg is connected to digital pin 13, negative end to ground.
- LED 3's positive leg is connected to digital pin 13, negative end to ground.



Handwritten code pic:



Date 7/11/2022
Page

Exp 2: Traffic Simulation

Aim: To simulate traffic light simulation by blinking of LEDs using arduino uno board.

Components required: Arduino uno Board, USB cable, jumper wires, 3 LEDs, bread board.

Pin connection:

LED 1 - long end to Pin 9, short end to GND.
LED 2 - long end to Pin 12, short end to GND.
LED 3 - long end to Pin 8, short end to GND.

Code:

```
void setup () {
pin Mode (9, OUTPUT);
pin Mode (12, OUTPUT);
pin Mode (8, OUTPUT);
}
void loop (),
    digital Write (9, HIGH);
    delay (1000);
    digital Write (9, LOW);
    delay (1000);
    digital write (12, HIGH);
    Delay (1000);
    digital Write (12, LOW);
    delay (1000);
    digital Write (8, HIGH);
    delay (1000);
    digital Write (8, LOW);
    delay (1000);
```

classmate
Date
Page

1

Observation : Successfully observed the traffic light simulation using LEDs.

Code:

```
void setup()
{
  pinMode(11, OUTPUT);
   pinMode(9, OUTPUT);
   pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(11, LOW);
  delay(1000);
  digitalWrite(9, LOW);
  delay(1000);


  digitalWrite(13, LOW);
  delay(1000);
  digitalWrite(9, HIGH);
  delay(1000);
  digitalWrite(11, LOW);
  delay(1000);

  digitalWrite(9, LOW);
  delay(1000);
  digitalWrite(11, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);


}
```

Observation: Successfully observed the traffic light simulation using LEDs.

Program no: **03** Program Title: **Push Button with LED**

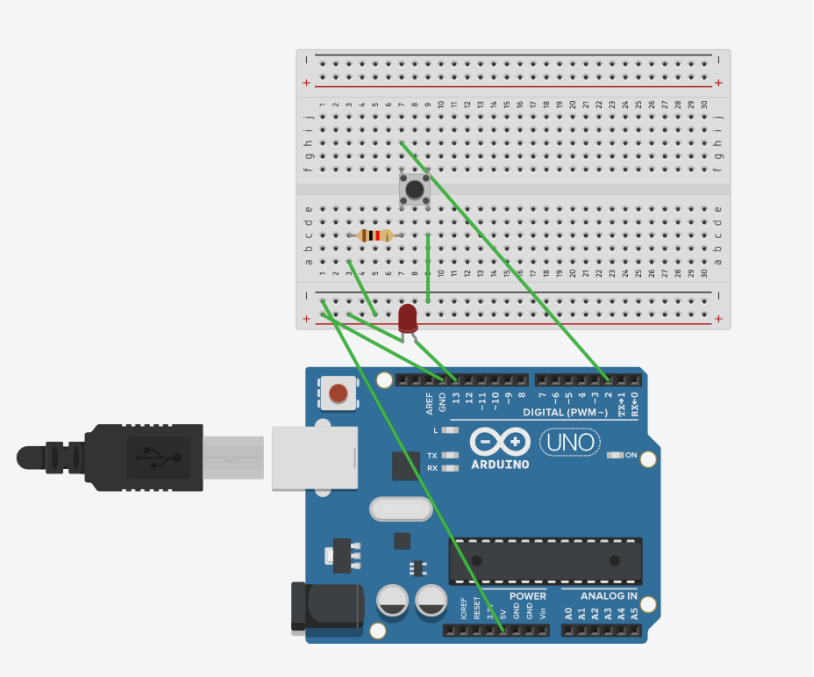
Aim: To simulate glowing of LEDs using a push button.

Hardware/components Required:

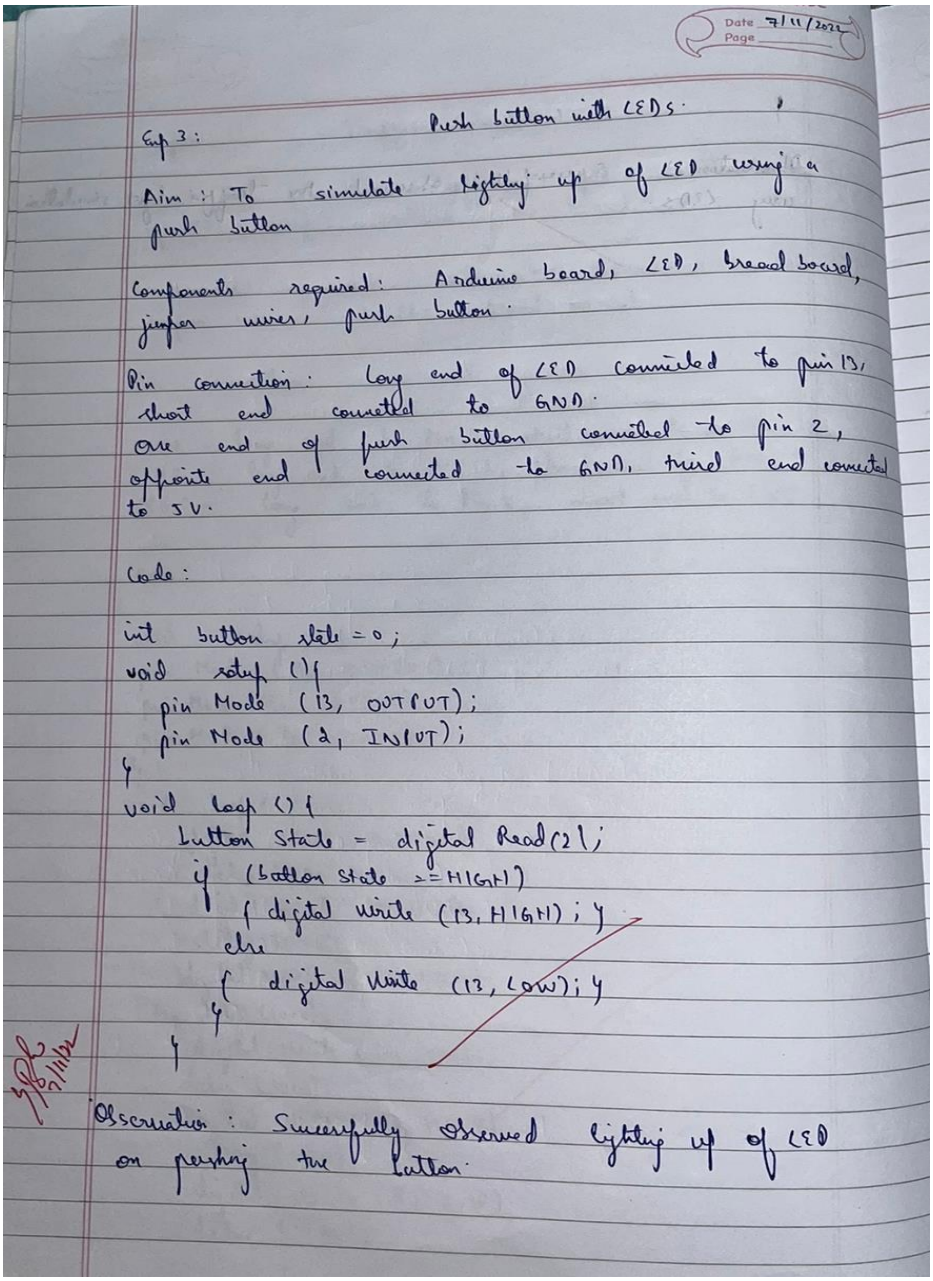
- Arduino Uno board - 1
- USB Cable - 1
- LED - 1
- Jumper wires
- Bread Board
- Push Button

Circuit Diagram / Pin connection:

- LED’s positive leg is connected to digital pin 13, negative end to ground.
- One end of push button connect to pin 2, opposite end connected to ground.
- Third end of push button connected to 5v supply.



Handwritten code pic:



Date 7/11/2022
Page

Exp 3: Push button with LEDs.

Aim: To simulate lighting up of LED using a push button

Components required: Arduino board, LED, bread board, jumper wires, push button.

Pin connection: long end of LED connected to pin 13, short end connected to GND.
One end of push button connected to pin 2, opposite end connected to GND, third end connected to 5V.

Code:

```
int button state = 0;
void setup (){
  pin Mode (13, OUTPUT);
  pin Mode (2, INPUT);
}
void loop (){
   button State = digital Read(2);
    if (button State == HIGH)
      { digital Write (13, HIGH); }
     else
      { digital Write (13, LOW); }
    }
```

Observation: Successfully observed lighting up of LED on pushing the button.

Code:

```
int buttonState= 0;
const int buttonPin =2;
const int ledPin =13;

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

void loop() {
  buttonState =digitalRead(buttonPin);
  if(buttonState==HIGH){
    digitalWrite(ledPin,HIGH);
  }
  else{
    digitalWrite(ledPin, LOW);
  }
}
```

Observation: Successfully observed the glowing of LED on pushing the button.

Program no: **04** Program Title: **LED fading without Potentiometer**

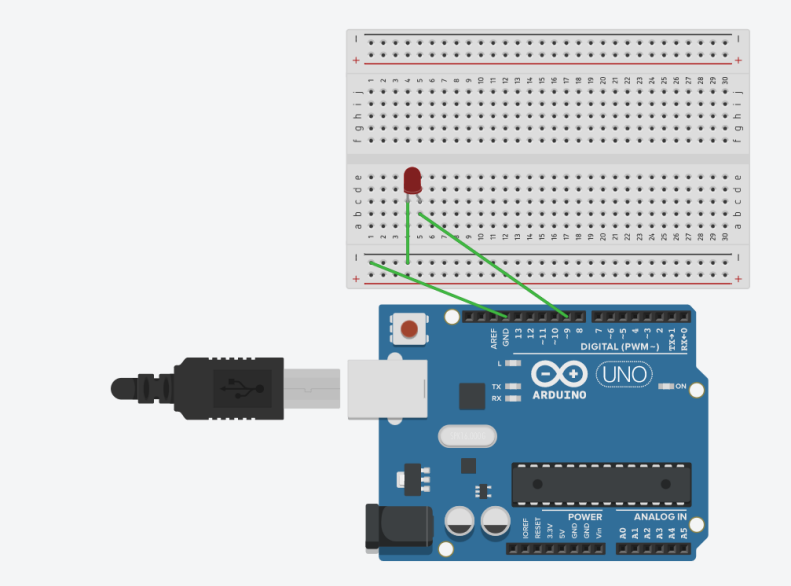
Aim: To simulate fading of LEDs without the use of potentiometer.

Hardware/components Required:

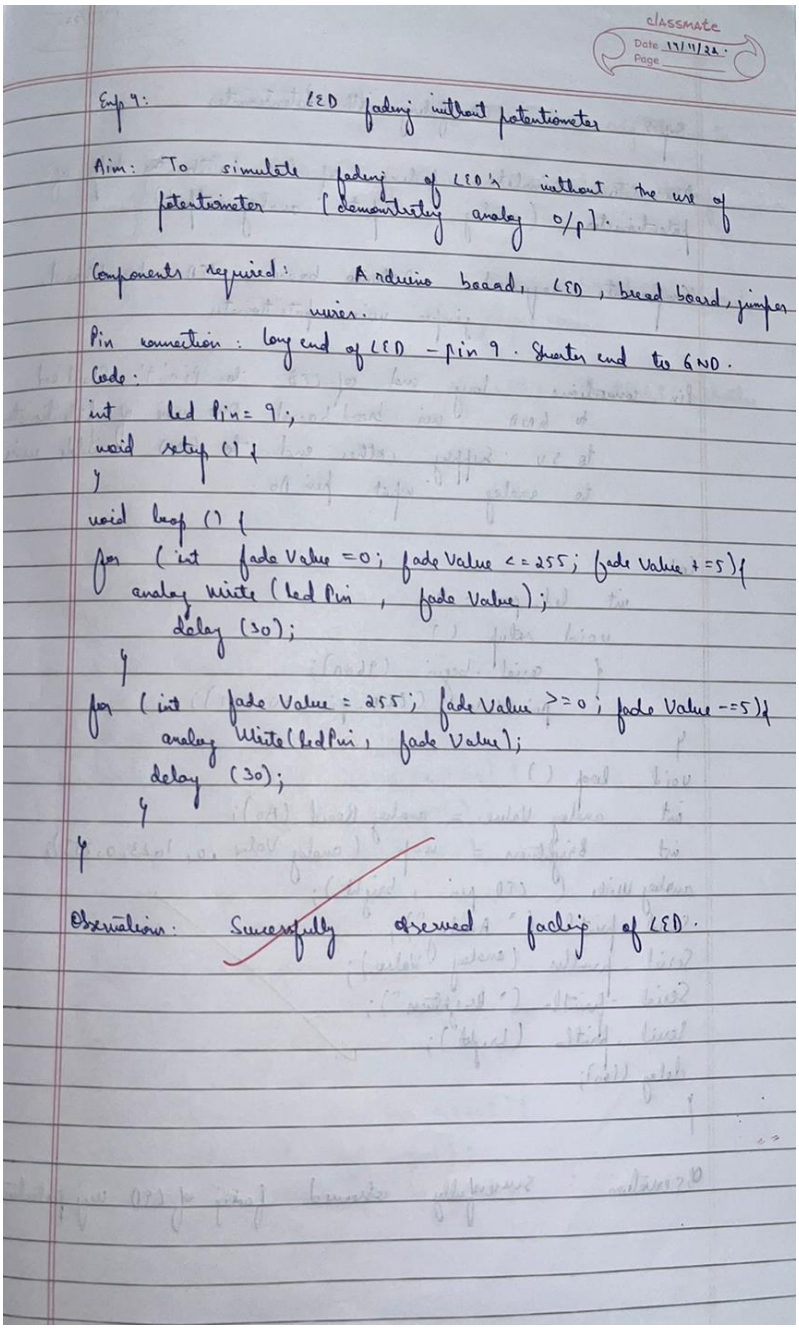
- Arduino Uno board - 1
- USB Cable - 1
- LED - 1
- Jumper wires
- Bread Board

Circuit Diagram / Pin connection:

- LED's positive leg is connected to digital pin 9, negative end to ground.



Handwritten code pic:



Exp 7: LED fading without potentiometer

Aim: To simulate fading of LED's without the use of potentiometer [demonstrating analog o/p].

Components required: Arduino board, LED, bread board, jumper wires.

Pin connection: long end of LED - pin 9. Shorter end to GND.

Code:

```
int led Pin = 9;
void setup () {
}
void loop () {
for (int fade Value = 0; fade Value <= 255; fade Value += 5){
    analog Write (led Pin , fade Value);
      delay (30);
  }
for (int fade Value = 255; fade Value >= 0; fade Value -= 5){
    analog Write(led Pin, fade Value);
    delay (30);
  }
}
```

Observation: Successfully observed fading of LED.

Code:

```
int ledPin = 9;

void setup() {
}

void loop() {

  for (int fadeValue = 0; fadeValue <= 255; fadeValue += 5) {
        analogWrite(ledPin, fadeValue);
      delay(30);
  }

  for (int fadeValue = 255; fadeValue >= 0; fadeValue -= 5) {
    analogWrite(ledPin, fadeValue);
      delay(30);
  }
}
```

Observation: Successfully observed the fading of LED without the use of potentiometer.

Program no: **05** Program Title: **LED fading with Potentiometer**

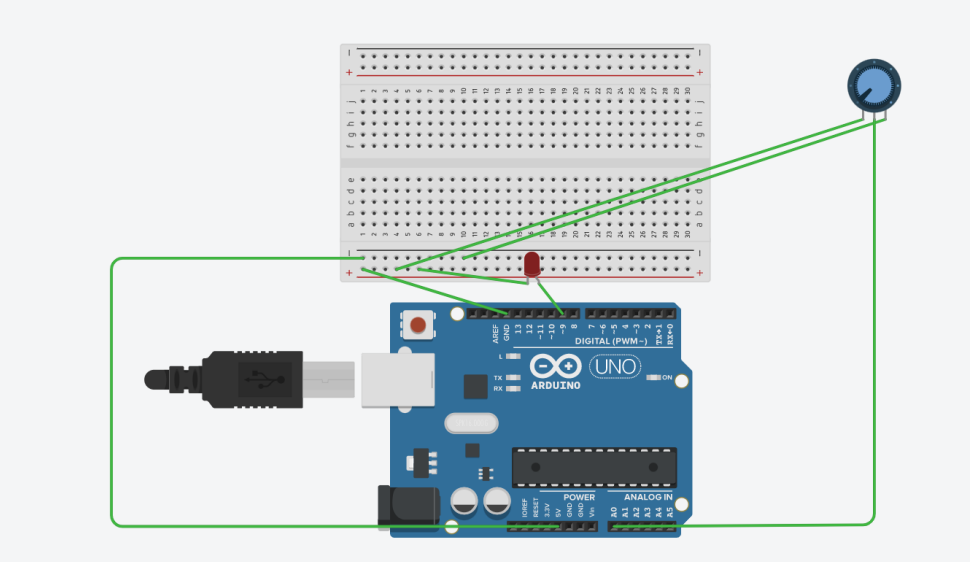
Aim: To simulate fading of LEDs without the use of potentiometer.

Hardware/components Required:

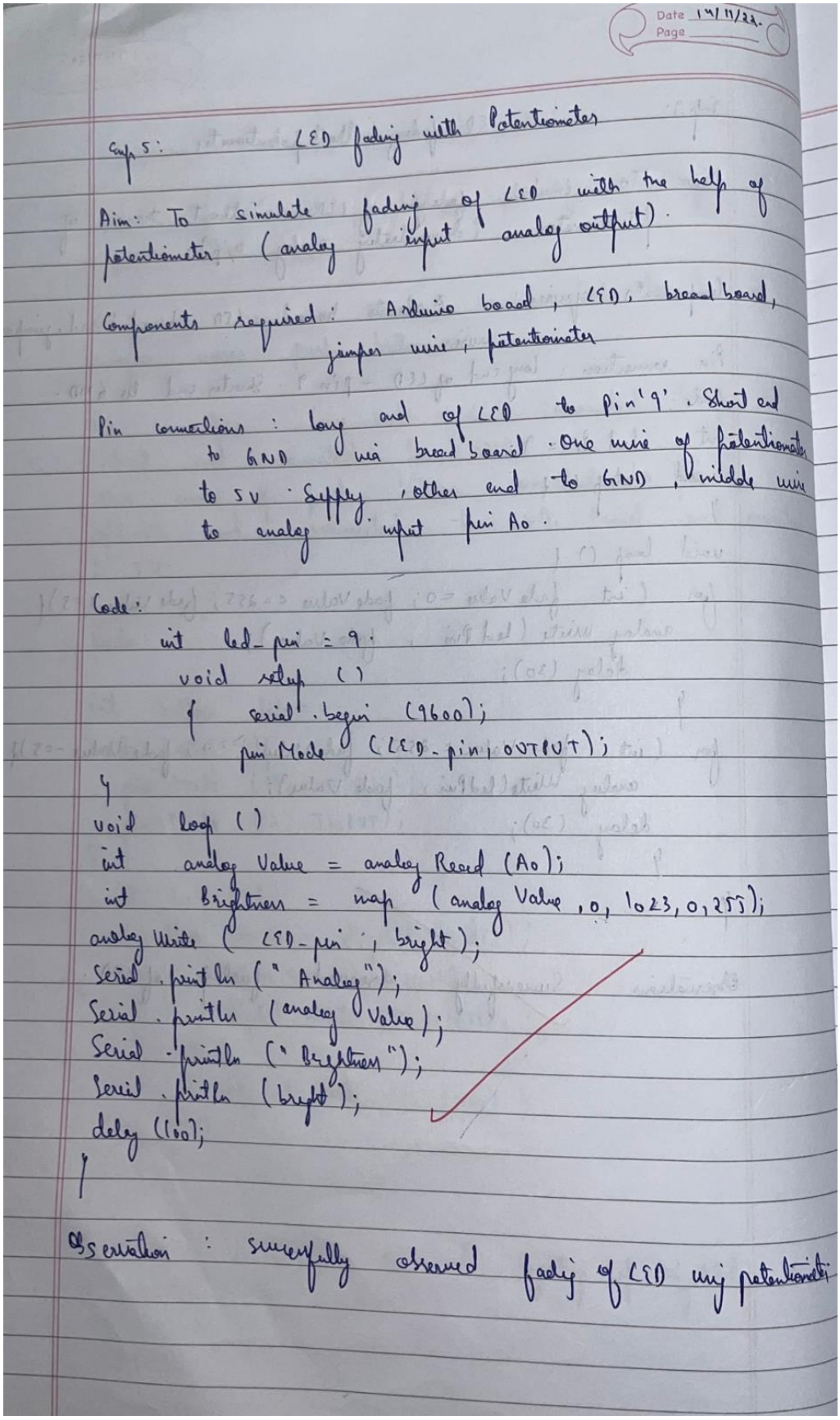
- Arduino Uno board - 1
- USB Cable - 1
- LED - 1
- Jumper wires
- Bread Board
- Potentiometer

Circuit Diagram / Pin connection:

- LED’s positive leg is connected to digital pin 9, negative end to ground.
- One end of potentiometer to 5v supply other end to ground.
- Middle wire of potentiometer to analog input pin A0.



Handwritten code pic:



Date 14/11/22.
Page

Exp 5: LED fading with Potentiometer

Aim: To simulate fading of LED with the help of potentiometer (analog input analog output).

Components required: Arduino board, LED, bread board, jumper wire, potentiometer

Pin connections: long end of LED to Pin '9'. Short end to GND via bread board. One wire of potentiometer to 5V. Supply, other end to GND, middle wire to analog input pin A0.

Code:

```
int led_pin = 9;
void setup ()
{ serial.begin (9600);
  pinMode (LED_pin, OUTPUT);
}
void loop ()
int analogValue = analogRead (A0);
int Brightness = map (analogValue, 0, 1023, 0, 255);
analogWrite (LED_pin, bright);
Serial.println ("Analog");
Serial.println (analogValue);
Serial.println ("Brightness");
Serial.println (bright);
delay (100);
}
```

Observation: successfully observed fading of LED using potentiometer

Code:

```
int ledPin = 9;
void setup()
{
Serial.begin(9600);
  pinMode(ledPin,OUTPUT);
}

void loop()
{
 int analogValue = analogRead(A0);
  int brightness = map(analogValue,0,1023,0,255);
  analogWrite(ledPin,brightness);
  Serial.print("Analog");
  Serial.print(analogValue);
  Serial.print("brightness");
  Serial.print(brightness);
  delay(100);
}
```

Observation: Successfully observed the fading of LED by turning the shaft of the potentiometer.

Program no: **06** Program Title: **Fading effect of two LEDs using two Potentiometers**

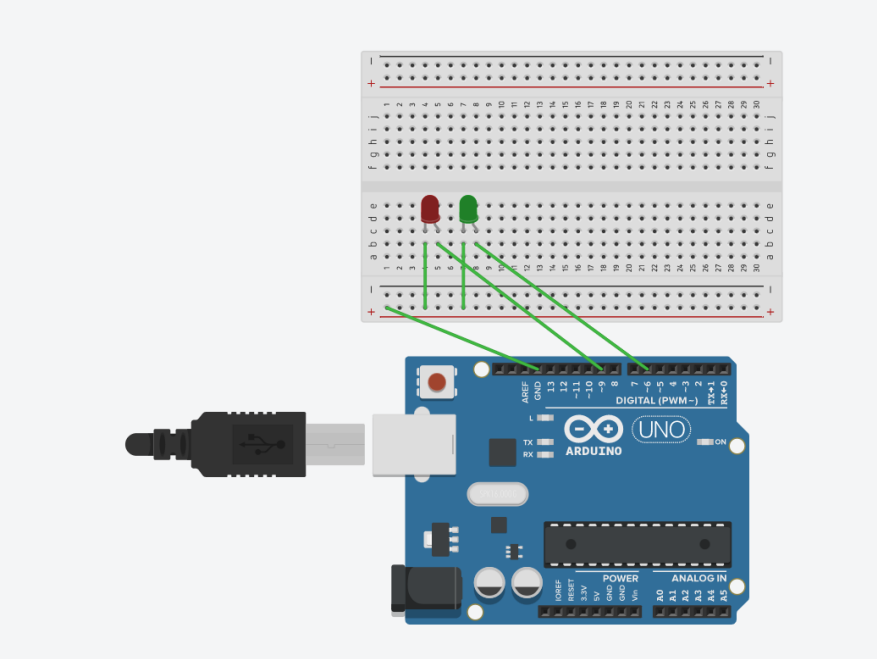
Aim: To simulate fading of two LEDs without the use of potentiometer.

Hardware/components Required:

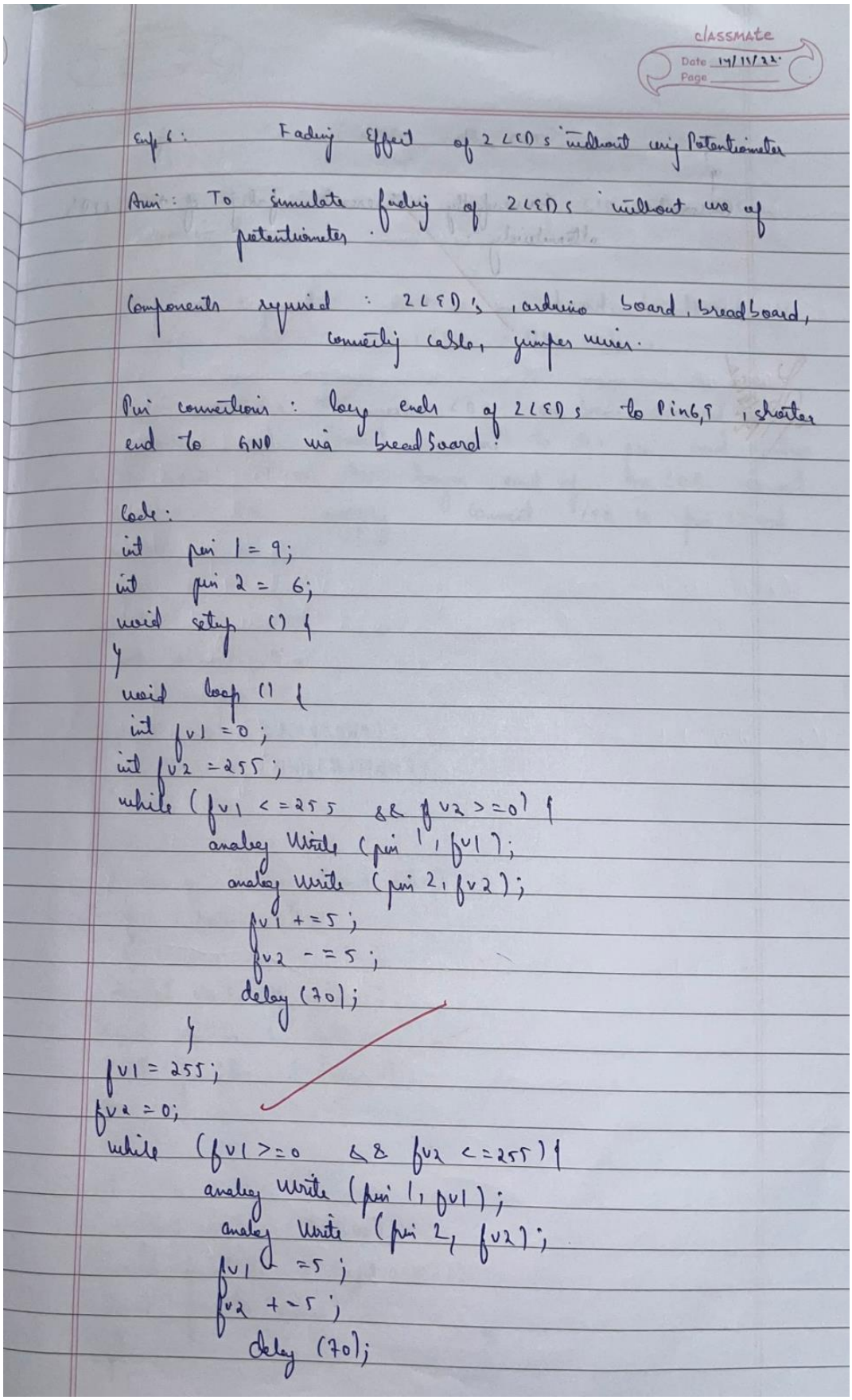
- Arduino Uno board - 1
- USB Cable - 1
- LEDs - 2
- Jumper wires
- Bread Board

Circuit Diagram / Pin connection:

- LED 1's positive leg is connected to digital pin 9, negative end to ground.
- LED 2's positive leg is connected to digital pin 6, negative end to ground.



Handwritten code pic:



Exp 6: Fading Effect of 2 LEDs without any Potentiometer

Aim: To simulate fading of 2 LEDs without use of potentiometer.

Components required: 2 LEDs, arduino board, breadboard, connecting cable, jumper wires.

Pin connections: long ends of 2 LEDs to Pin 6, 9, shorter end to GND via breadboard.

Code:

```
int pin1 = 9;
int pin2 = 6;
void setup () {
}
void loop () {
 int fv1 = 0;
int fv2 = 255;
while (fv1 <= 255 && fv2 >= 0) {
        analogWrite (pin1, fv1);
        analogWrite (pin2, fv2);
          fv1 += 5;
          fv2 -= 5;
          delay (70);
      }
fv1 = 255;
fv2 = 0;
while (fv1 >= 0 && fv2 <= 255) {
        analogWrite (pin1, fv1);
        analogWrite (pin2, fv2);
        fv1 -= 5;
        fv2 += 5;
          delay (70);
```

Code:

```
int ledpin1 = 9;
int ledpin2 = 6;
void setup(){
}

void loop()
{
int fadevalue1 =0;
int fadevalue2 =255;

 while(fadevalue1 <=255&& fadevalue2>=0)
 {
  analogWrite(ledpin1,fadevalue1);
  analogWrite(ledpin2,fadevalue2);
  fadevalue1 +=5;
  fadevalue2 -=5;
  delay(70);
 }

  fadevalue1 =255;
  fadevalue2 =0;
  while(fadevalue1>=0 && fadevalue2<=255)
  {
  analogWrite(ledpin1,fadevalue1);
  analogWrite(ledpin2,fadevalue2);
  fadevalue1 -=5;
  fadevalue2 +=5;
  delay(70);
  }
}
```

Observation: Successfully observed the fading of two LEDs alternatively without using potentiometer.

Program no: **07** Program Title: **LDR with LED**

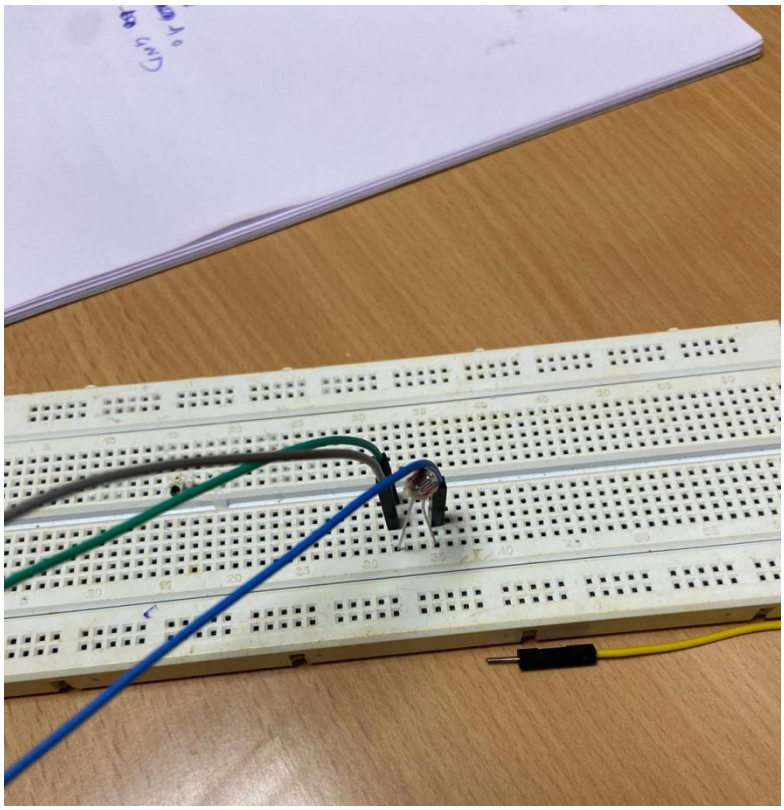
Aim: To observe the slow of LED when the LDR is covered.

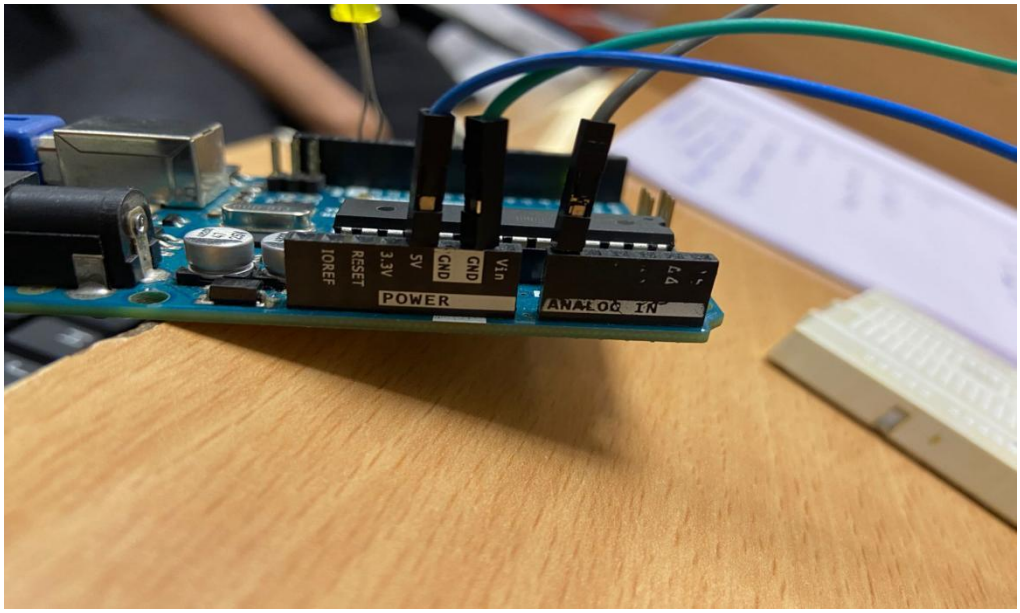
Hardware/components Required:

- Arduino Uno board - 1
- USB Cable - 1
- LED - 1
- Jumper wires
- Bread Board
- LDR - 1

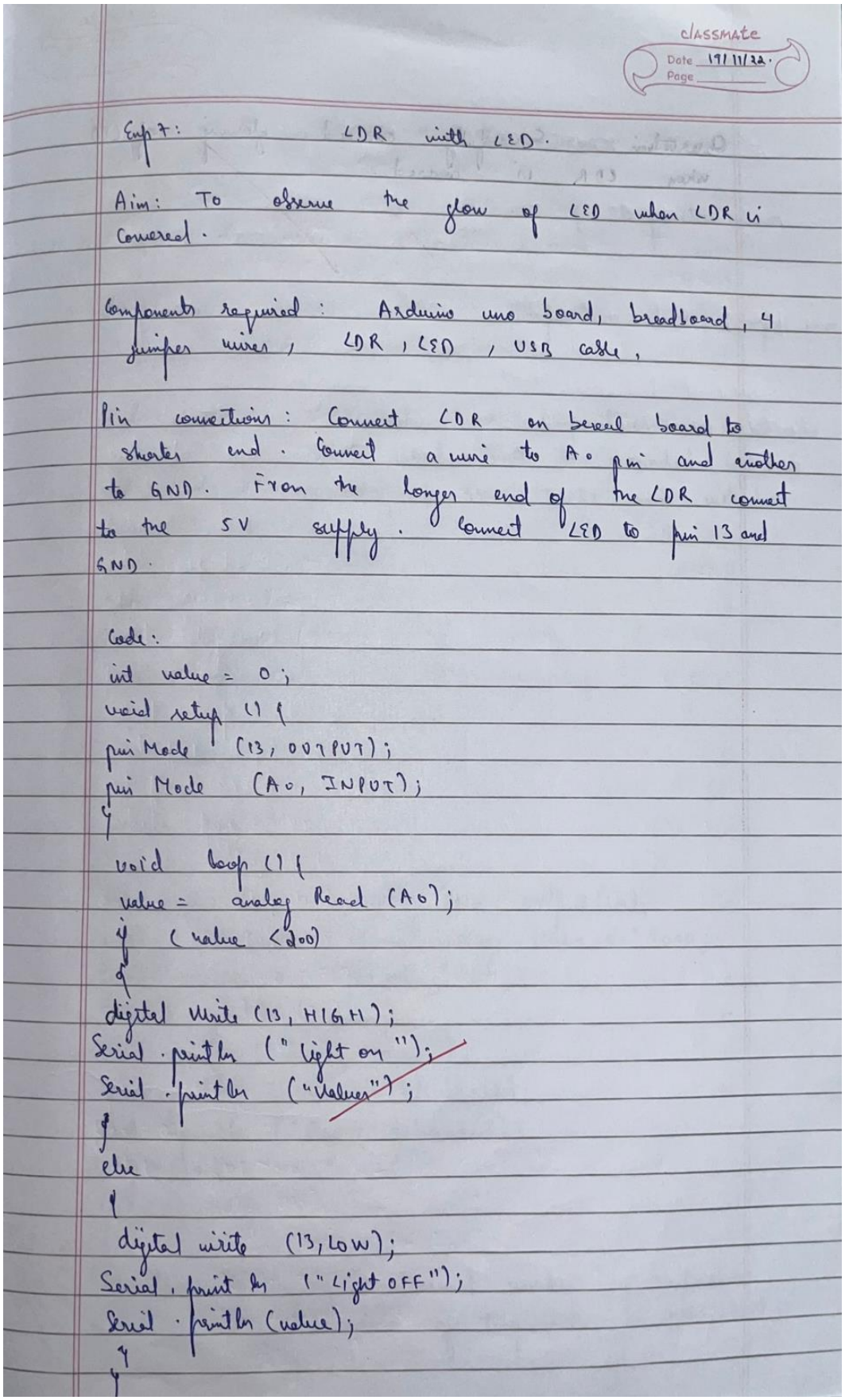
Circuit Diagram / Pin connection:

- Connect shorter end of LED to ground, longer end of LED to pin 13
- Connect shorter end of LDR to A0 pin as well as ground via breadboard
- Connect longer end of LDR to 5v supply




IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A7

Handwritten code pic:



Exp 7: LDR with LED.

Aim: To observe the glow of LED when LDR is covered.

Components required: Arduino uno board, breadboard, 4 jumper wires, LDR, LED, USB cable,

Pin connections: Connect LDR on bread board to shorter end. Connect a wire to A0 pin and another to GND. From the longer end of the LDR connect to the 5V supply. Connect LED to pin 13 and GND.

Code:

```
int value = 0;
void setup () {
pinMode (13, OUTPUT);
pinMode (A0, INPUT);
}
void loop () {
value = analogRead (A0);
if (value < 200)
{
digitalWrite (13, HIGH);
Serial.println ("Light on");
Serial.println ("value");
}
else
{
digitalWrite (13, LOW);
Serial.println ("Light OFF");
Serial.println (value);
}
}
```

Code:

```
int value =0;
void setup()
{
  Serial.begin(9600);
pinMode(13,OUTPUT);
pinMode(A0,INPUT);
}

void loop()
{
  value = analogRead(A0);
  if(value<200)
  {
digitalWrite(13,HIGH);
    Serial.println("light on");
    Serial.println(value);
  }

  else
  {
   digitalWrite(13,LOW);
    Serial.println("light off");
    Serial.println(value);
  }
}
```

Observation: Successfully observed the glowing of LED when the LDR was covered.

Program no: **08** Program Title: **LM35 Temperature Sensor**

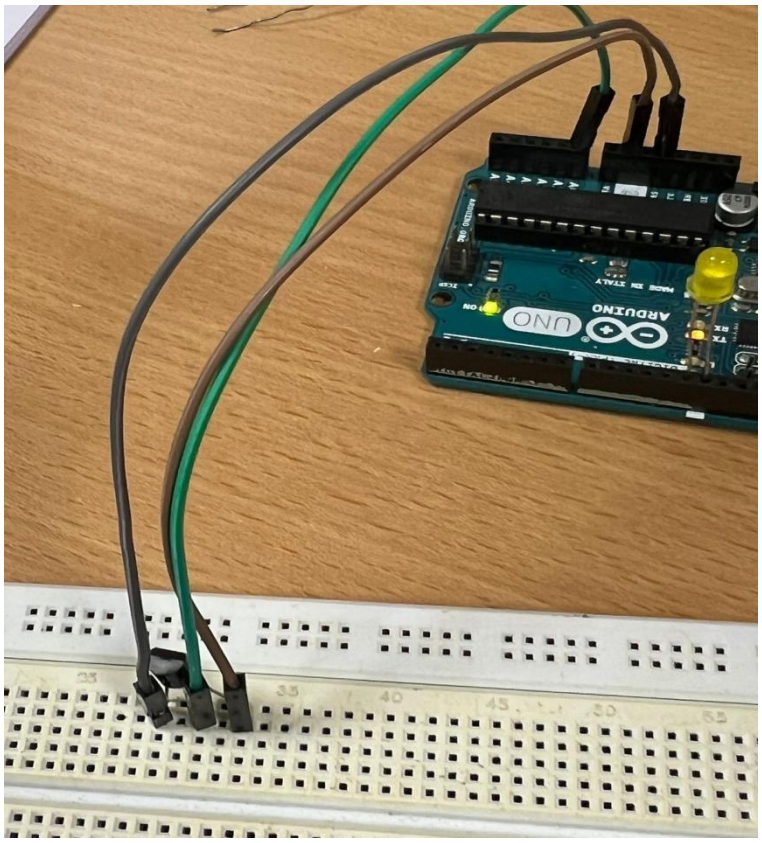
Aim: To observe the working of temperature sensor and convert to Fahrenhiet

Hardware/components Required:

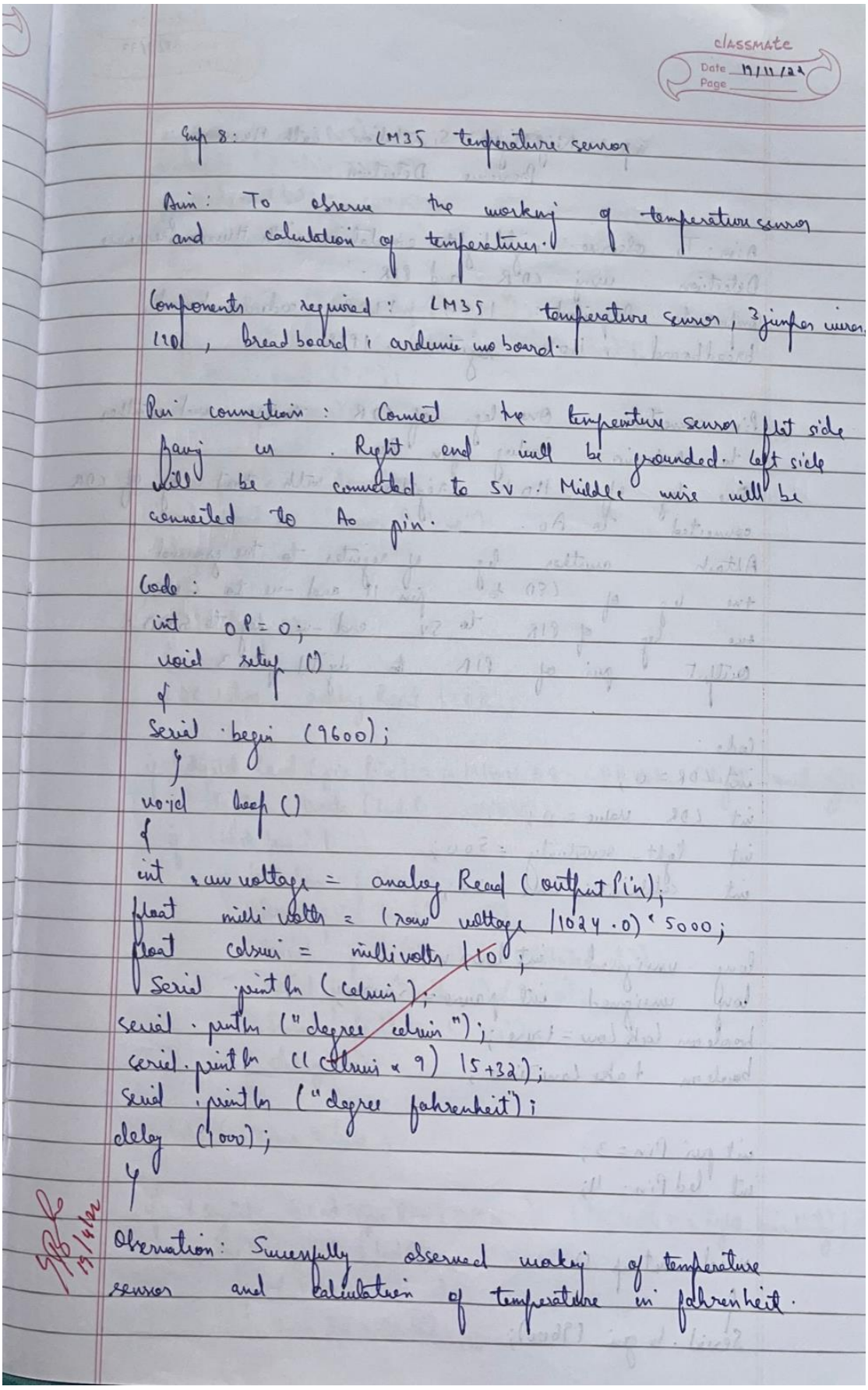
- Arduino Uno board - 1
- USB Cable - 1
- LM35 Temperature Sensor
- Jumper wires
- Bread Board

Circuit Diagram / Pin connection:

- Place the temperature sensor flat side facing us.
- Right end of LM35 will be grounded, left end will be connected to 5v supply.
- Middle end will be connected to A0 pin.



Handwritten code pic:



Exp 8: LM35 temperature sensor

Aim: To observe the working of temperature sensor and calculation of temperature.

Components required: LM35 temperature sensor, 3 jumper wires (M-M), bread board, arduino uno board.

Pin connection: Connect the temperature sensor flat side facing us. Right end will be grounded. Left side will be connected to 5V. Middle wire will be connected to A0 pin.

Code:

```
int op = 0;
void setup()
{
Serial.begin(9600);
}
void loop()
{
int raw voltage = analogRead(outputPin);
float millivolts = (raw voltage / 1024.0) * 5000;
float celsius = millivolts / 10;
Serial.println(celsius);
serial.println("degree celsius");
serial.println((celsius * 9) / 5 + 32);
Serial.println("degree fahrenheit");
delay(1000);
}
```

Observation: Successfully observed working of temperature sensor and calculation of temperature in fahrenheit.

Code:

```
int outputpin=0;

void setup()
{
Serial.begin(9600);
}
Void loop()
{
int rawvoltage=analogRead(outputpin);
float millivolts=(rawvoltage / 1024.0) * 5000;
float celsius - millivolts/10;
serial.print(celsius);
serial.print(“degree celsius”);
serial.print((celsius * 9)/5 + 32);
serial.print(“degrees fahrenheit”);
Delay(1000);
}
```

Observation: Successfully observed the calculation of temperature in Fahrenheit.

Program no: **9** Program Title: **Night Light Simulation with Human Presence Detection**

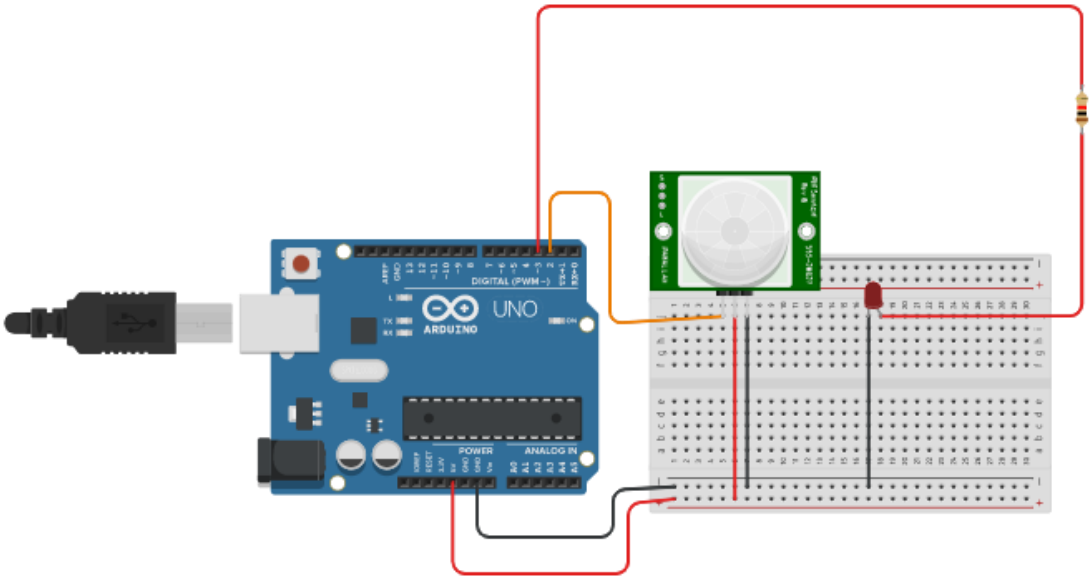
Aim: To observe night light simulation with human presence detection using LDR and PIR.

Hardware/components Required:

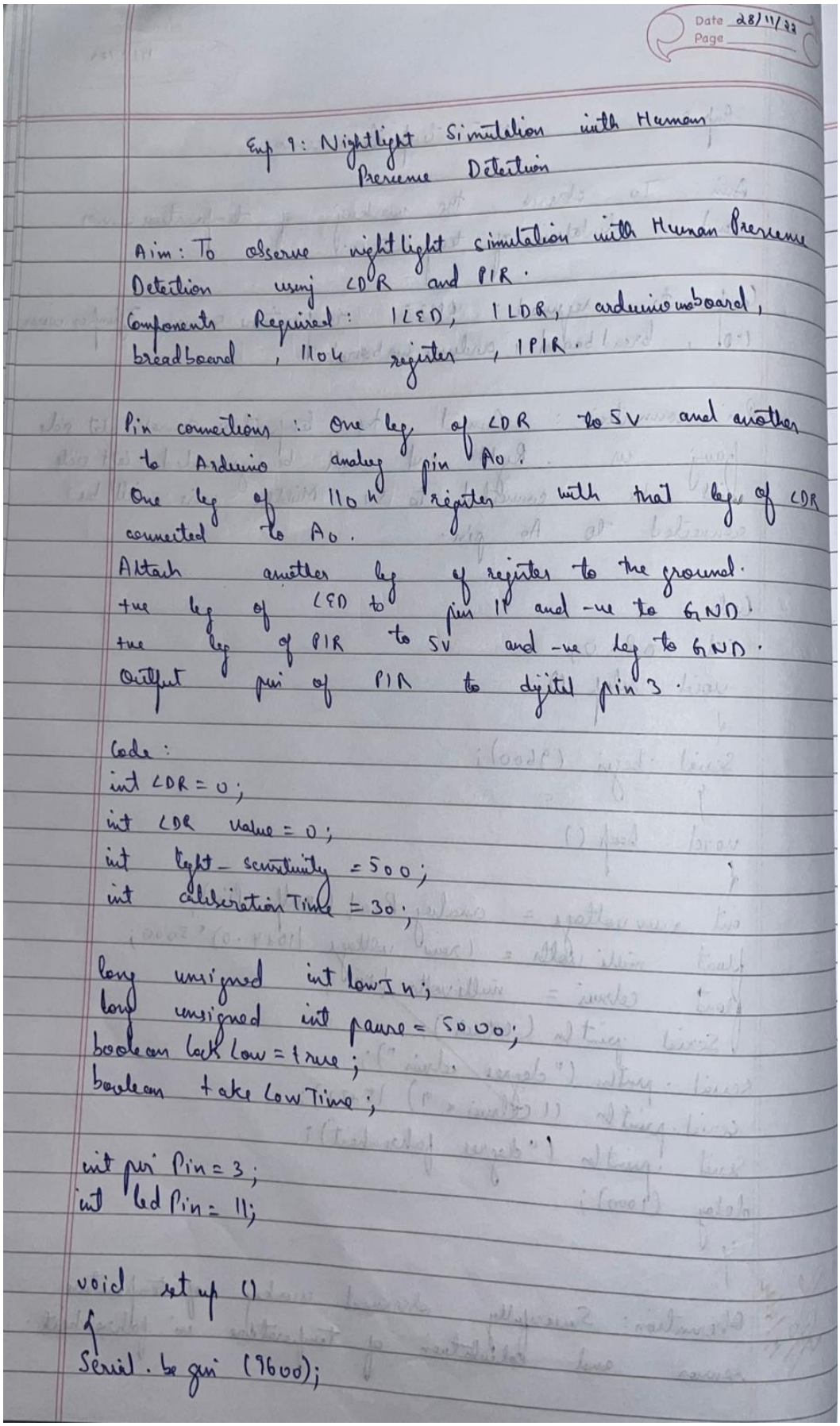
- Arduino Uno board - 1
- USB Cable - 1
- LEDs - 3
- Jumper wires
- Bread Board
- LDR - 1
- PIR - 1
- 110k Resistor

Circuit Diagram / Pin connection:

- Attach one leg of LDR to 5V and another leg to Arduino Analog pin A0
- Attach one leg of 110K register with that leg of LDR connected to A0
- Attach another leg of register to the ground
- Connect the positive leg of LED to pin 11 and negative to GND
- Connect positive leg of PIR to 5V and negative leg to GND
- Connect output pin of PIR to digital pin 3



Handwritten code pic:



Date 28/11/23
Page

Exp 9: Nightlight Simulation with Human Presence Detection

Aim: To observe nightlight simulation with Human Presence Detection using LDR and PIR.
Components Required: 1 LED, 1 LDR, arduino uno board, breadboard, 1104 register, 1 PIR.

Pin connections: One leg of LDR to 5V and another to Arduino analog pin A0.
One leg of 1104 register with trail leg of LDR connected to A0.
Attach another leg of register to the ground.
true leg of LED to pin 12 and -ve to GND.
true leg of PIR to 5V and -ve leg to GND.
Output pin of PIR to digital pin 3.

Code:

```
int LDR = 0;
int LDR value = 0;
int light_sensitivity = 500;
int calibrationTime = 30;

long unsigned int lowIn;
long unsigned int pause = 5000;
boolean lockLow = true;
boolean takeLowTime;

int pirPin = 3;
int ledPin = 11;

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

```
pinMode (11, OUTPUT);
pinMode (pirPin, INPUT);
pinMode (ledPin, OUTPUT);
digitalWrite (pirPin, LOW);

Serial.print ("Calibrating sensor");
  for (int i=0; i< calibrationTime; i++){
    Serial.print (".");
      delay (1000);
}
Serial.println ("done");
Serial.println ("Sensor Active");
delay (50);
}
void loop ()
 {
 LDRValue = analogRead (LDR);

if (digitalRead (pirPin) == HIGH && LDRValue < light_sensitivity){
    digitalWrite (ledPin, HIGH);
  if (lockLow){
       lockLow = false;
        Serial.println ("---");
        Serial.print ("motion detected at");
        Serial.print (millis ()/1000);
        Serial.println ("sec");
        delay (50);
      }
  takeLowTime = true;
}
if (digitalRead (pirPin) == LOW)|| LDR Value >= light_sensitivity){
    digitalWrite (ledPin, LOW);
      if (takeLowTime){
        lowIn = millis ();
```

Code:

```
int LDR = 0;
int LDRValue = 0;
int light_sensitivity = 500;
int calibrationTime = 30;

long unsigned int lowIn;
long unsigned int pause = 5000;
boolean lockLow = true;
boolean takeLowTime;

int pirPin = 3;
int ledPin = 11;

void setup()
  {
    Serial.begin(9600);
    pinMode(11, OUTPUT);

    pinMode(pirPin, INPUT);
    pinMode(ledPin, OUTPUT);
    digitalWrite(pirPin, LOW);

    Serial.print("calibrating sensor ");

    for(int i = 0; i<calibrationTime; i++){
       Serial.print(".");
       delay(1000);
     }
     Serial.println(" done");
     Serial.println("SENSOR ACTIVE");
     delay(50);
  }

void loop()
  {
    LDRValue = analogRead(LDR);
     Serial.println(LDRValue);
     if(digitalRead(pirPin) == HIGH && LDRValue < light_sensitivity){
       digitalWrite(ledPin, HIGH);
       if(lockLow){
          lockLow = false;
          Serial.println("---");
          Serial.print("motion detected at ");
          Serial.print(millis()/1000);
          Serial.println(" sec");
          delay(50);
        }
       takeLowTime = true;
```

```
    }

    if(digitalRead(pirPin) == LOW ||  LDRValue >= light_sensitivity){
       digitalWrite(ledPin, LOW);
       if(takeLowTime){
         lowIn = millis();
         takeLowTime = false;
       }

    if(!lockLow&&millis() - lowIn> pause){

      lockLow = true;
      Serial.print("motion ended at ");
      Serial.print((millis() - pause)/1000);
      Serial.println(" sec");
      delay(50);
   }
     delay(100);
  }
}
```

Observation: Successfully observed the glowing of LED when LDR was covered and motion is sensed by the PIR.

Program no: **10** Program Title: **Ultra Sonic Sensor**

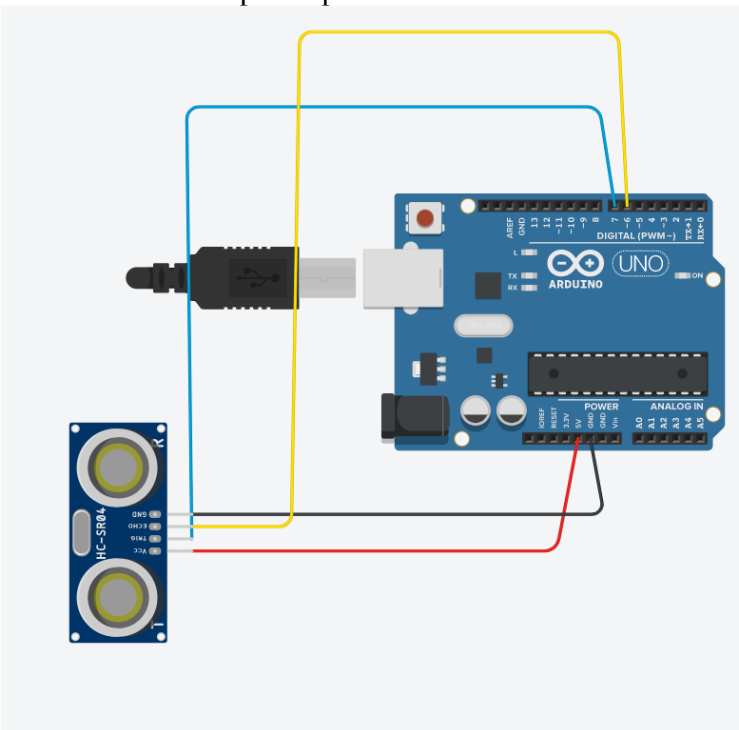
Aim: To observe and understand the working of the ultra sound sensor.

Hardware/components Required:

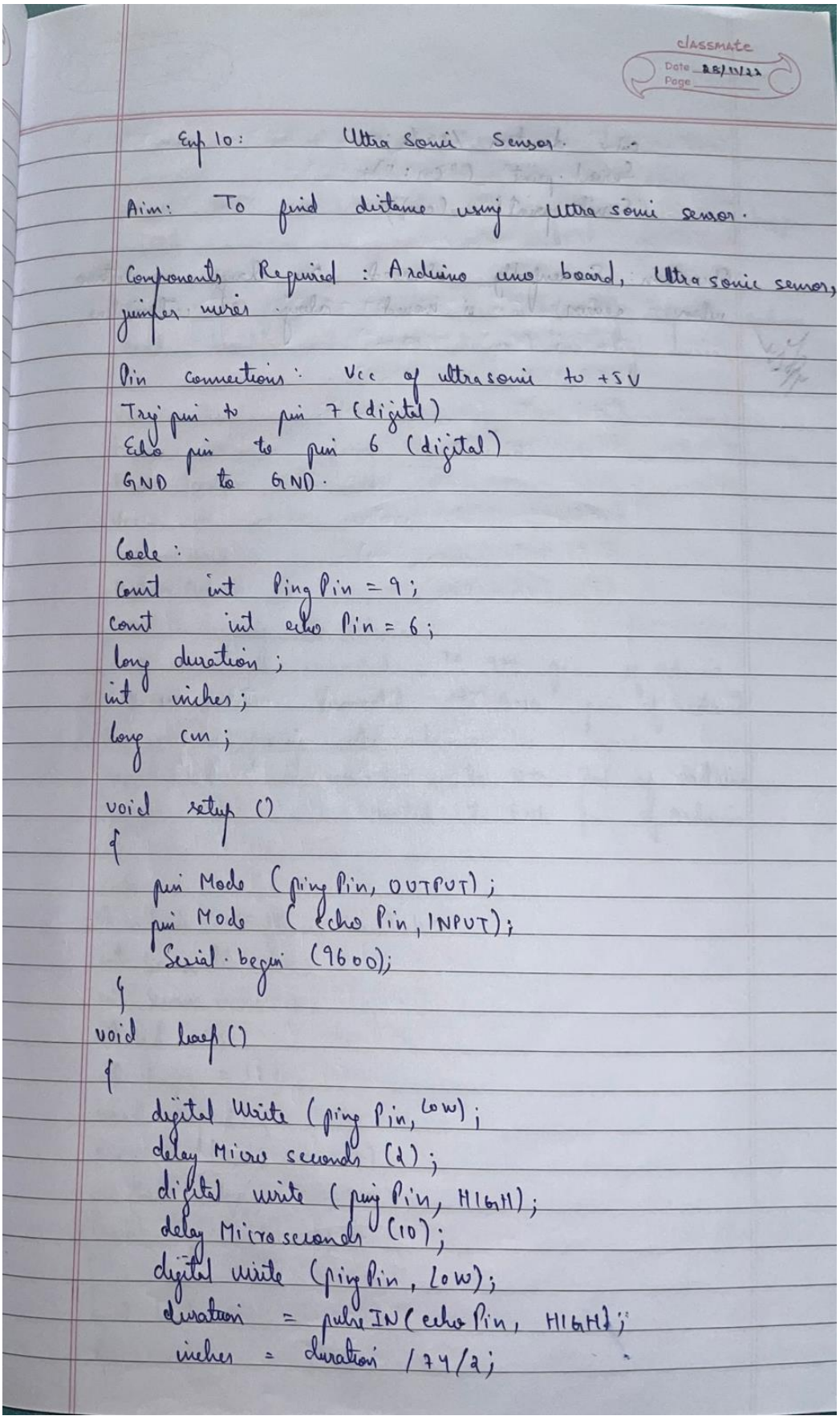
- Arduino Uno board - 1
- USB Cable - 1
- Jumper wires
- Bread Board
- HC-SR04

Circuit Diagram / Pin connection:

- Connect ground of HC-SR04 to ground of Arduino
- Connect Vcc of HC-SR04 to 5v supply of Arduino
- Connect trig pin to pin 7
- Connect echo pin to pin 6



Handwritten code pic:



Exp 10: Ultra Sonic Sensor.

Aim: To find distance using Ultra sonic sensor.

Components Required: Arduino uno board, Ultra sonic sensor, jumper wires.

Pin connections: Vcc of ultrasonic to +5V
Trig pin to pin 7 (digital)
Echo pin to pin 6 (digital)
GND to GND.

Code:

```
const int PingPin = 9;
const int echoPin = 6;
long duration;
int inches;
long cm;

void setup()
{
  pinMode(pingPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
}
void loop()
{
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(pingPin, LOW);
  duration = pulseIN(echoPin, HIGH);
  inches = duration/74/2;
```

Code:

```
const int pingPin = 7;
echoPin = 6;
void setup()
{
Serial. begin(9600);
pinMode(pingPin, OUTPUT);
pinMode(echoPin, INPUT);
}
void loop()
{
long duration, inches, cm;
digitalWrite (pingPin, LOW);
delayMicroseconds(2);
digitalWrite (pingPin, HIGH);
delayMicroseconds(10);
digitalWrite (pingPin, LOW);
duration = pulseln(echoPin, HIGH);
inches = microsecondsTolnches(duration);
cm = microseconds ToCentimeters(duration);
}

Long microsecondsToinches(long microseconds)
{
return microseconds 174/2;
}
Long microsecondsToinches(long microseconds)
{
return microseconds /74/2;
}

Long microsecondsTocentimeters(long microseconds)
{
return /29/2;
}
```

Observation: Successfully observed the distance of the obstacle from the sensor in cm.

Program no: **11** Program Title: **Fire Alert**

Aim: To simulate the working of fire alarm using buzzer and LED.

Hardware/components Required:

- Flame sensor (Analogue Output)
- Arduino
- Bread board
- LED
- Buzzer
- Connecting wires

Circuit Diagram / Pin connection:

Flame sensor interfacing to Arduino

Flame sensor to Arduino

vcc -> vcc

gnd -> gnd

A0 -> A0

Led interfacing to Arduino

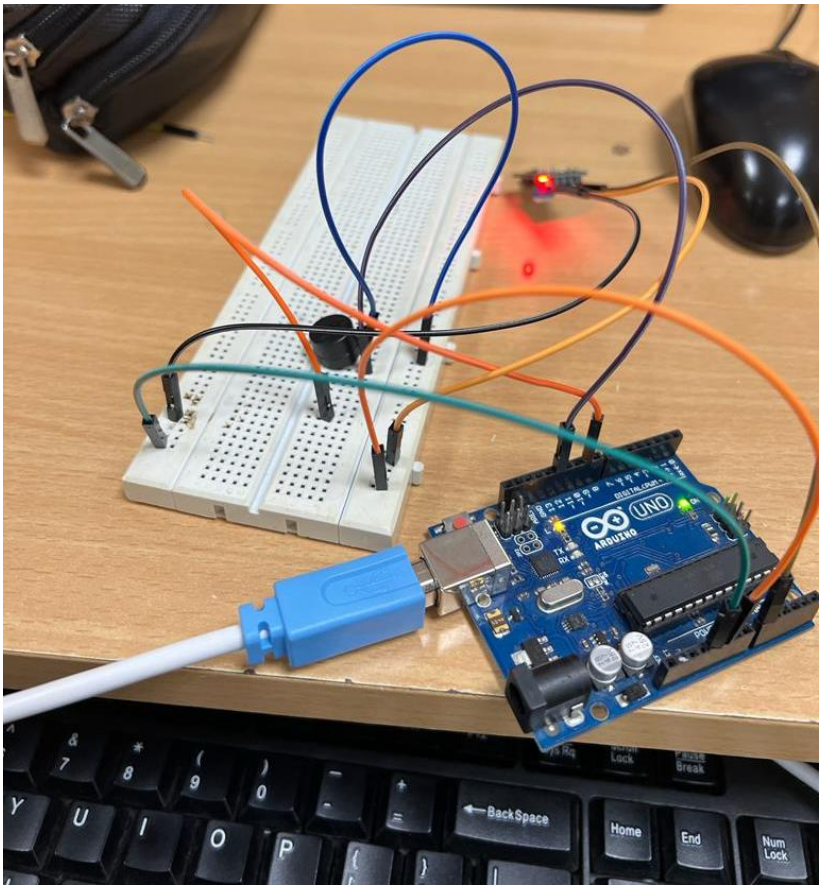
LED +ve is connected to **9th pin** of Arduino

LED -ve is connected to **gnd pin** of arduino

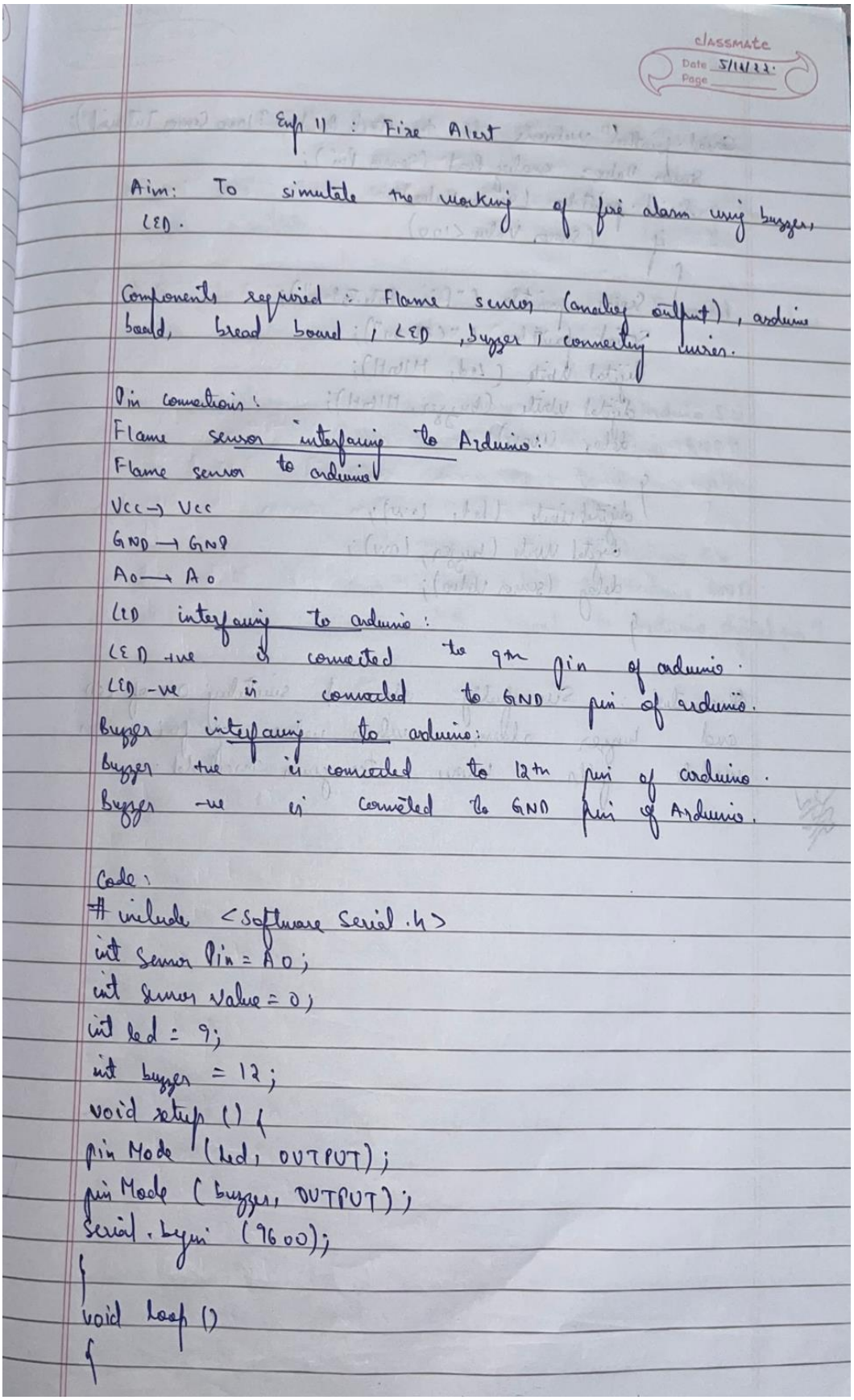
Buzzer interfacing to Arduino

Buzzer +ve is connected to **12th pin** of Arduino

Buzzer -ve is connected to **GND** pin of Arduino


UNO
ARDUINO
BackSpace
Home
End
Num Lock

Handwritten code pic:



Exp 11 : Fire Alert

Aim: To simulate the working of fire alarm using buzzer, LED.

Components required : Flame sensor (analog output), arduino board, bread board, LED, buzzer, connecting wires.

Pin connections:

Flame sensor interfacing to Arduino:

Flame sensor to arduino

Vcc → Vcc

GND → GND

A0 → A0

LED interfacing to arduino:

LED +ve is connected to 9th pin of arduino.

LED -ve is connected to GND pin of arduino.

Buzzer interfacing to arduino:

Buzzer +ve is connected to 12th pin of arduino.

Buzzer -ve is connected to GND pin of Arduino.

Code:

```
#include <SoftwareSerial.h>
int SensorPin = A0;
int sensorValue = 0;
int led = 9;
int buzzer = 12;
void setup () {
pinMode (led, OUTPUT);
pinMode (buzzer, OUTPUT);
Serial.begin (9600);
}
void loop ()
{
```

Code:

```
#include<SoftwareSerial.h>
int sensorPin = A0;
int sensorValue = 0;
int led = 9;
int buzzer = 12;
void setup() {
pinMode(led, OUTPUT);
pinMode(buzzer,OUTPUT);
Serial.begin(9600);
}
void loop()
{
Serial.println("Welcome to TechPonder Flame Sensor Tutorial");
sensorValue = analogRead(sensorPin);
Serial.println(sensorValue);
if (sensorValue < 100)
{
Serial.println("Fire Detected");
Serial.println("LED on");
digitalWrite(led,HIGH);
digitalWrite(buzzer,HIGH);
delay(1000);
}
digitalWrite(led,LOW);
digitalWrite(buzzer,LOW);
delay(sensorValue);
}
```

Observation: Successfully observed the blinking of LED and buzzing of buzzer when flame was detected by flame sensor.

Program no: **12** Program Title: **Automatic Irrigation Controller Simulation**

Aim: Sensing the soil moisture and sprinkling the water simulation.

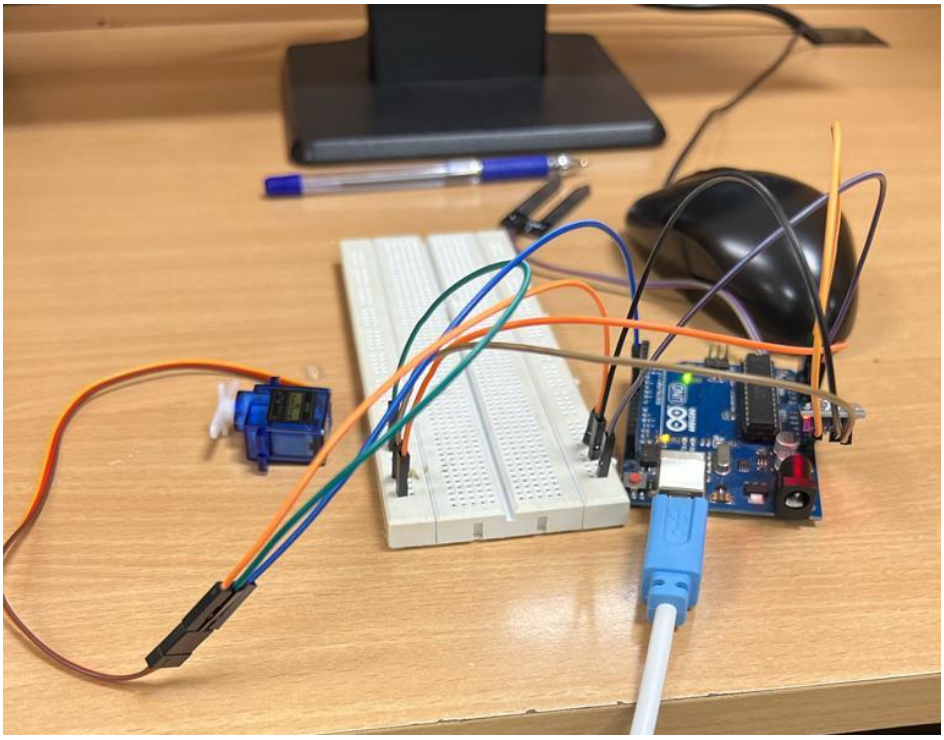
Hardware/components Required:

- Arduino Uno board - 1
- USB Cable - 1
- Moisture Sensor
- Mini Servo Motor
- Jumper wires
- Bread Board

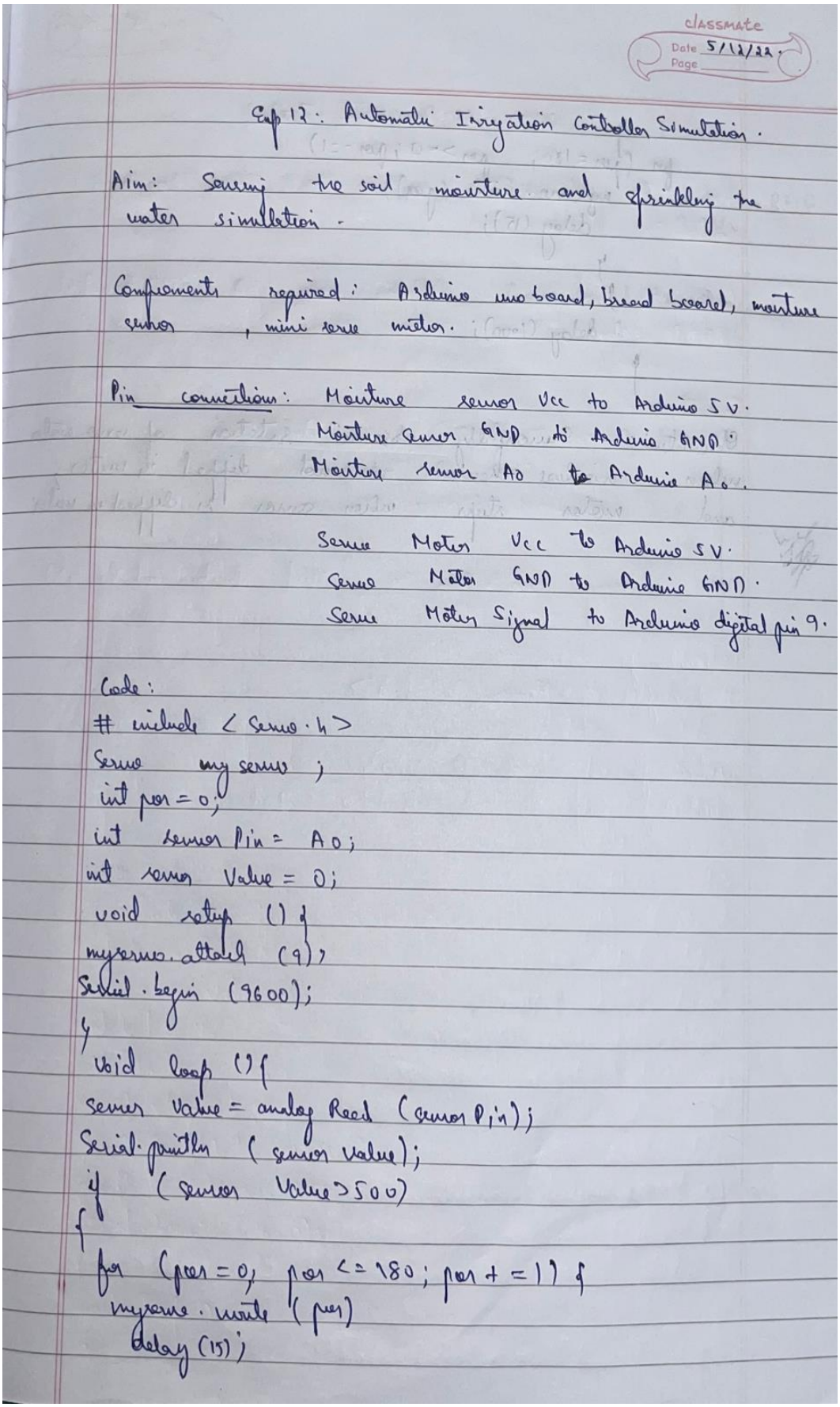
Circuit Diagram / Pin connection:

- Moisture sensor VCC to Arduino 5V
- Moisture sensor GND to Arduino GND
- Moisture sensor A0 to Arduino A0

- Servo motor VCC to Arduino 5V
- Servo motor GND to Arduino GND
- Servo Motor Signal to Arduino digital pin 9



Handwritten code pic:



classmate
Date 5/12/22
Page

Exp 12: Automatic Irrigation Controller Simulation.

Aim: Sensing the soil moisture and sprinkling the water simulation.

Components required: Arduino uno board, bread board, moisture sensor, mini servo motor.

Pin connections: Moisture sensor Vcc to Arduino 5V.
Moisture sensor GND to Arduino GND.
Moisture sensor A0 to Arduino A0.

Servo Motor Vcc to Arduino 5V.
Servo Motor GND to Arduino GND.
Servo Motor Signal to Arduino digital pin 9.

Code:

```
# include < Servo.h >
Servo myservo ;
int pos = 0;
int sensorPin = A0;
int sensorValue = 0;
void setup () {
myservo.attach (9);
Serial.begin (9600);
}
void loop (){
sensorValue = analogRead (sensorPin);
Serial.println (sensorValue);
if (sensorValue > 500)
{
for (pos = 0; pos <= 180; pos += 1) {
myservo.write (pos)
delay (15);
```

```
}
for (pos=180; pos>=0; pos-=1)
  myservo.write(pos);
  delay(15);
}
}
  delay(1000);
}
```

Observation: Successfully observed rotation of servo motor when moisture sensor is not dipped in water, and motor stops when sensor is dipped in water.

Code:

```
#include <Servo.h>
Servo myservo;

int pos = 0;
int sensorPin = A0;
int sensorValue = 0;
void setup() {
 myservo.attach(9);
Serial.begin(9600);
}
void loop() {
  sensorValue = analogRead(sensorPin);
Serial.println (sensorValue);
if(sensorValue>500)
{
  for (pos = 0; pos <= 180; pos += 1) {

    myservo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    myservo.write(pos);
    delay(15);
  }
}
delay (1000);
}
```

Observation: Successfully observed the rotation of motor when the moisture sensor is not in contact with water. And observed the stopping of motor when the mositure sensor is in contact with water.

Program no: **13** Program Title: **RFID Reader**

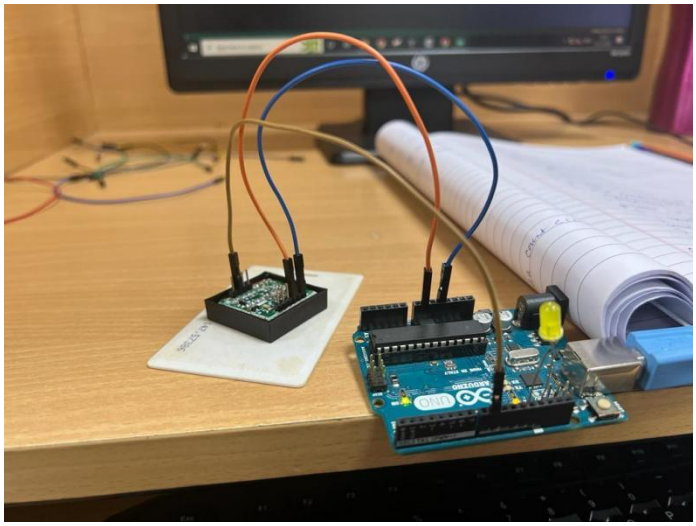
Aim: To count the number of RFID tags read by the RFID reader

Hardware/components Required:

- Arduino Uno board - 1
- USB Cable - 1
- Jumper wires
- Bread Board
- RFID Reader module
- RFID Tags

Circuit Diagram / Pin connection:

- Ground of RFID to ground of Arduino Uno board.
- TX of RFID to RX(9) of Arduino.



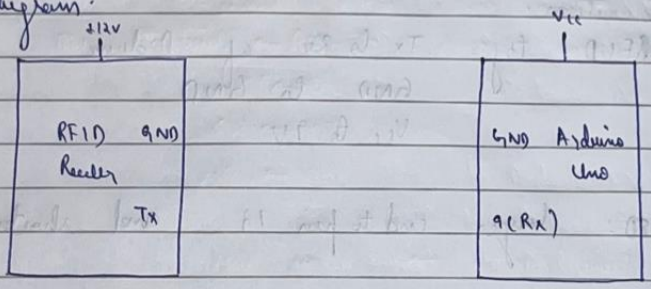
Handwritten code pic:

Exp13: RFID Reader ;

Aim: To count number of RFID tags read by the RFID reader.

Components Required: Arduino uno board, RFID tags, RFID reader module.

Circuit diagram:



Code:

```
# include <Softwareserial.h>
 Softwareserial mySerial(9,10)
void setup(){
mySerial.begin (9600);
Serial.begin (9600);
}
void loop(){
   if (mySerial.available()>0){
Serial.write(mySerial.read());
}
}
```

Observation: successfully able to read the RFID ID on the serial monitor.

Code:

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(9, 10);
void setup()
{
mySerial.begin(9600);
Serial.begin(9600); }
void loop()
{
if(mySerial.available()>0)
{
Serial.write(mySerial.read());
}
}
```

Observation: Was successfully able to read the RFID ID on the serial monitor.

Program no: **14** Program Title: **RFID Access control**

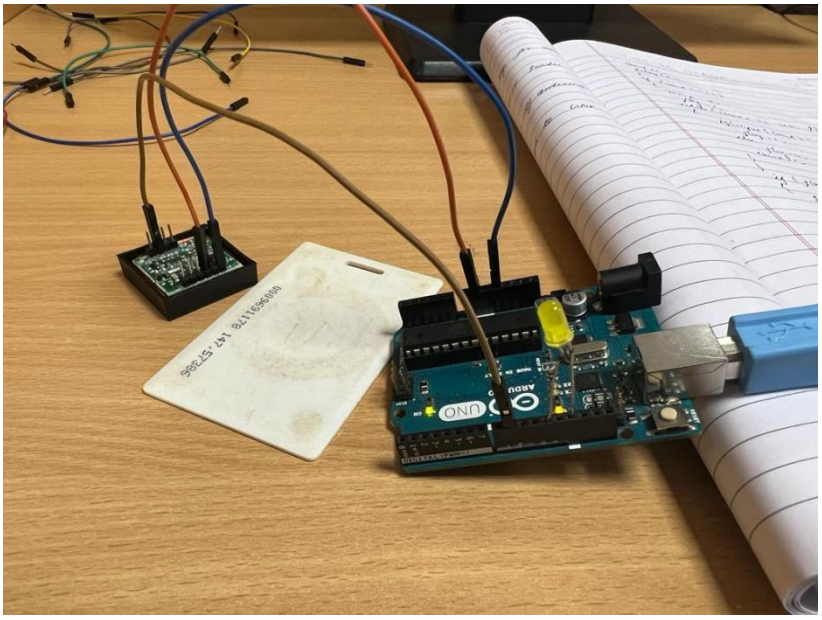
Aim: To demonstrate the use of RFID tags for access control.

Hardware/components Required:

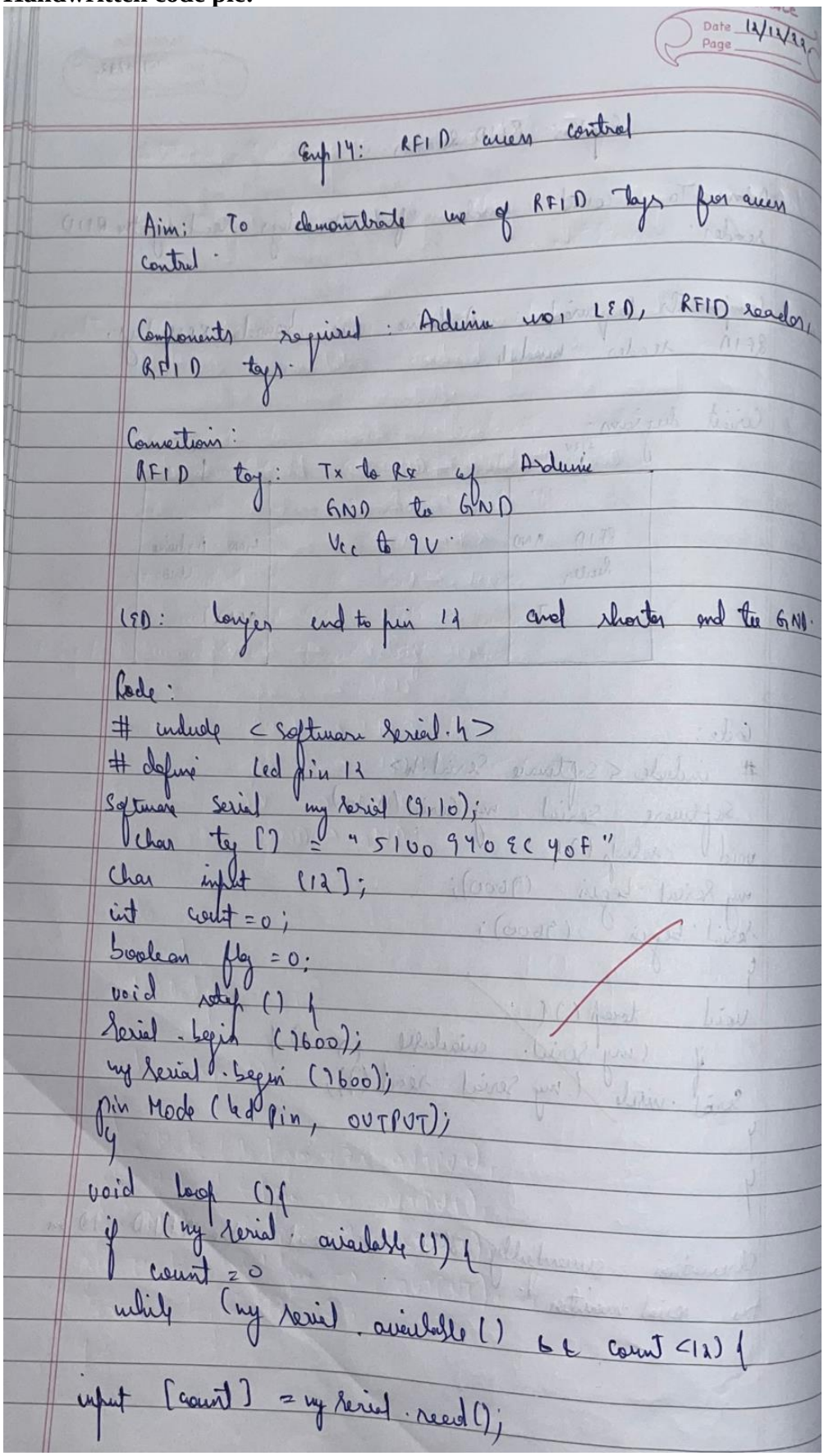
- Arduino Uno board - 1
- USB Cable - 1
- Jumper wires
- Bread Board
- RFID Reader
- RFID Tag

Circuit Diagram / Pin connection:

- RFID TX to Rx of Arduino
- Ground of RFID reader to ground of Arduino
- VCC of RFID to 5v supply of Arduino
- Longer end of LED to pin 12 and shorter end to Ground



Handwritten code pic:



Date 12/12/23
Page

Exp 14: RFID access control

Aim: To demonstrate use of RFID tags for access control.

Components required: Arduino uno, LED, RFID reader, RFID tags.

Connections:
RFID tag: Tx to Rx of Arduino
GND to GND
Vcc to 9V.

LED: longer end to pin 12 and shorter end to GND.

Code:

```
# include <SoftwareSerial.h>
# define led pin 12
SoftwareSerial myserial (9,10);
char tag [] = "5100 9A0 EC 40F";
char input [12];
int count = 0;
boolean flag = 0;
void setup () {
Serial.begin (7600);
myserial.begin (7600);
pin Mode (led Pin, OUTPUT);
}
void loop (){
if (myserial.available ()) {
count = 0
while (myserial.available () && count < 12) {
input [count] = myserial.read ();
```

```
Serial.write(input[count]);
count++;
delay(5);
}
if (count == 12) {
count = 0;
flag = 1;
while (count < 12 && flag != 0)
if (input[count] == key[count]) {
        flag = 1;
    } else {
            flag = 0; }
        count++;
    }
}
if (flag == 1) {
    Serial.println("Access Allowed!!");
    digitalWrite(ledpin, HIGH);
    delay(2000);
    digitalWrite(ledpin, LOW);
}
else {
    Serial.println("Access denied");
    digitalWrite(ledpin, LOW);
    delay(2000);
}
for (count = 0; count < 12; count++){
        input[count] = 'F';
    }
    count = 0;
}
}
```

Observation: Successfully observed the glowing of LED when the R
key and message "Access Allowed" was printed on the serial monitor

Code:

```
#include<SoftwareSerial.h>
SoftwareSerial mySerial(9, 10);
#define LEDPIN 12
char tag[] ="3C0087D597F9";
char input[12];
int count = 0;
input[] character array
boolean flag = 0;

void setup()
{  Serial.begin(9600);
mySerial.begin(9600);
pinMode(LEDPIN,OUTPUT); //WRONG TAG INDICATOR
}
void loop()
{
if(mySerial.available())
{
count = 0;
while(mySerial.available() && count < 12)
{
input[count] = mySerial.read();
Serial.write(input[count]);
count++; // increment counter
delay(5);
}
if(count == 12) //
{
count =0; // reset counter varibale to 0
flag = 1;

while(count<12 && flag !=0)
{
if(input[count]==tag[count])
flag = 1; // everytime the values match, we set the flag variable
to 1
else
flag= 0;
count++;
}
}
if(flag == 1)
{
Serial.println("Access Allowed!");
digitalWrite(LEDPIN,HIGH);
delay (2000);
digitalWrite (LEDPIN,LOW);
```

```
}
else
{
Serial.println("Access Denied");
digitalWrite(LEDPIN,LOW);
delay(2000);
}
for(count=0; count<12; count++)
{
input[count]= 'F';
}
count = 0; // Reset counter variable
}
}
```

Observation: Successfully observed the glowing of LED when the RFID key and message “Access Allowed” was printed on the serial monitor.

Program no: **15** Program Title: **Bluetooth Home Automation**

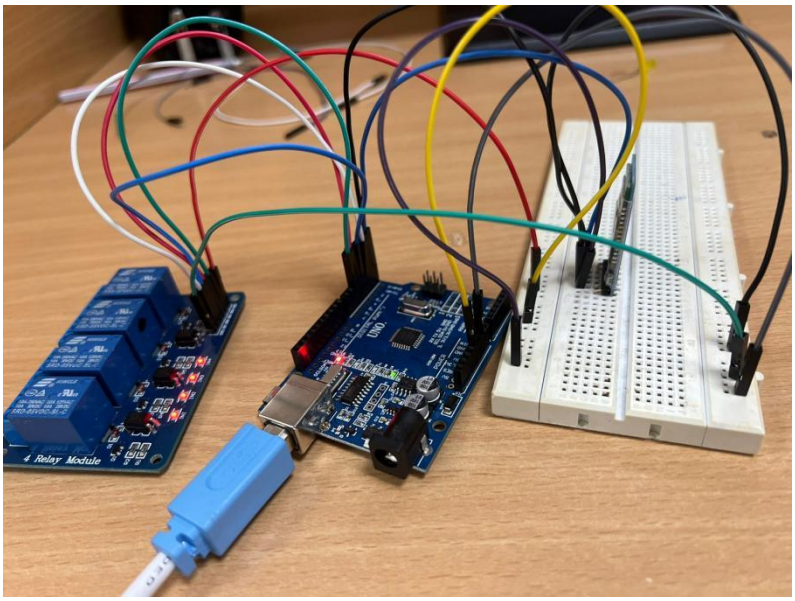
Aim: To control the working of relay through Android Mobile.

Hardware Required:

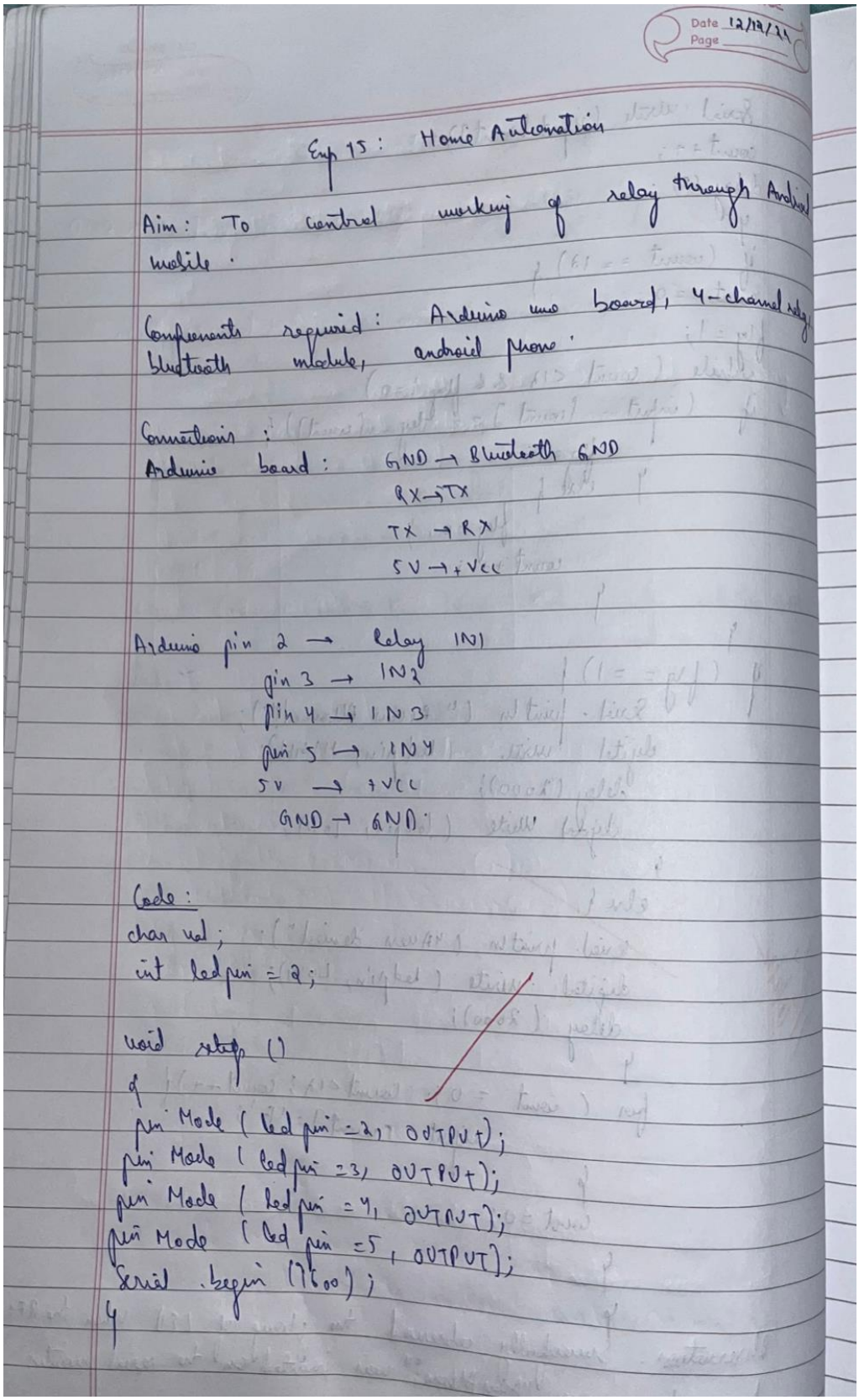
- Arduino
- 4-Channel relay
- Bluetooth Module
- Android phone

Circuit Diagram / Pin connection:

- Output 1 to Pin 4 (Arduino Board)
  Output 2 to Pin 5
  Output 3 to Pin 6
  Output 4 to Pin 7
- Bluetooth Module Tx to Pin 0
  Bluetooth Module Rx to Pin 1
- VCC of Bluetooth & relay should be connected to Arduino 5V(through breadboard)
- GND of Bluetooth & relay should be connected to Arduino GND



Handwritten code pic:



Date 12/12/21
Page

Exp 15: Home Automation

Aim: To control working of relay through Android mobile.

Components required: Arduino uno board, 4-channel relay, bluetooth module, android phone.

Connections:
Arduino board: GND → Bluetooth GND
RX → TX
TX → RX
5V → +Vcc

Arduino pin 2 → Relay IN1
pin 3 → IN2
pin 4 → IN3
pin 5 → IN4
5V → +Vcc
GND → GND

Code:

```
char val;
int ledpin = 2;

void setup ()
{
pin Mode ( led pin =2, OUTPUT);
pin Mode ( led pin =3, OUTPUT);
pin Mode ( led pin =4, OUTPUT);
pin Mode ( led pin =5, OUTPUT);
Serial .begin (9600);
}
```

```
void loop()
{
 if (Serial.available())
 {
  val = Serial.read();
 Serial.println(val);
  if (val == 'a')
  {
   digitalWrite(ledpin = 2, HIGH);
  }
  if (val == 'A')
  {
  digitalWrite(ledpin = 2, LOW);
  }
 if (val == 'b')
 {
  digitalWrite(ledpin = 3, HIGH);
 }
 if (val == 'B')
 {
  digitalWrite(ledpin = 3, LOW);
 }
  if (val == '(')
  {
  digitalWrite(ledpin = 4, LOW);
 }
 if (val == 'D')
 {
   digitalWrite(ledpin = 5, LOW);
 }
  if (val == '(')
  {
   digitalWrite(ledpin = 4, HIGH);
  }
```

```
if (val == 'd')
    digitalWrite(led_pin, !digitalRead(led_pin));
}
}
```

Observation: Was successfully able to control the glowing of LEDs using mobile phone.

Code:

```
char val;
int ledpin = 2;

void setup()
{
  pinMode(ledpin = 2, OUTPUT);
  pinMode(ledpin = 3, OUTPUT);
  pinMode(ledpin = 4, OUTPUT);
  pinMode(ledpin = 5, OUTPUT);

    Serial.begin(9600);
}

void loop()
{
  if( Serial.available() )
  {
    ;
  }
  val = Serial.read();

  if( val == 'a' )
  {
    digitalWrite(ledpin = 2, HIGH);
  }

  if( val == 'A' )
  {
    digitalWrite(ledpin = 2, LOW);
  }

  if( val == 'b' )
  {
    digitalWrite(ledpin = 3, HIGH);
  }

  if( val == 'B' )
  {
    digitalWrite(ledpin = 3, LOW);    }

  if( val == 'C' )
  {

    digitalWrite(ledpin=4, LOW);
  }

  if( val == 'D' )
  {
```

```
   digitalWrite(ledpin=5, LOW);
  }


 if( val == 'c' )
  {
   digitalWrite(ledpin = 4, HIGH);
  }
  if( val == 'd' )
  {
   digitalWrite(ledpin = 5, HIGH);
  }
  }
```

Observation: Was successfully able to control the glowing of LEDs using mobile phone.

Program no: **16** Program Title: **Working with GSM Module**

GSM Module: Call to a particular number

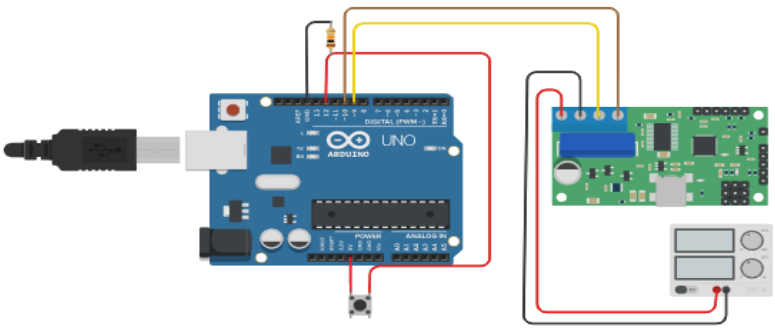
Hardware/components Required:

- Arduino Uno board - 1
- USB Cable - 1
- SIM900 GSM Module
- Jumper wires
- Bread Board

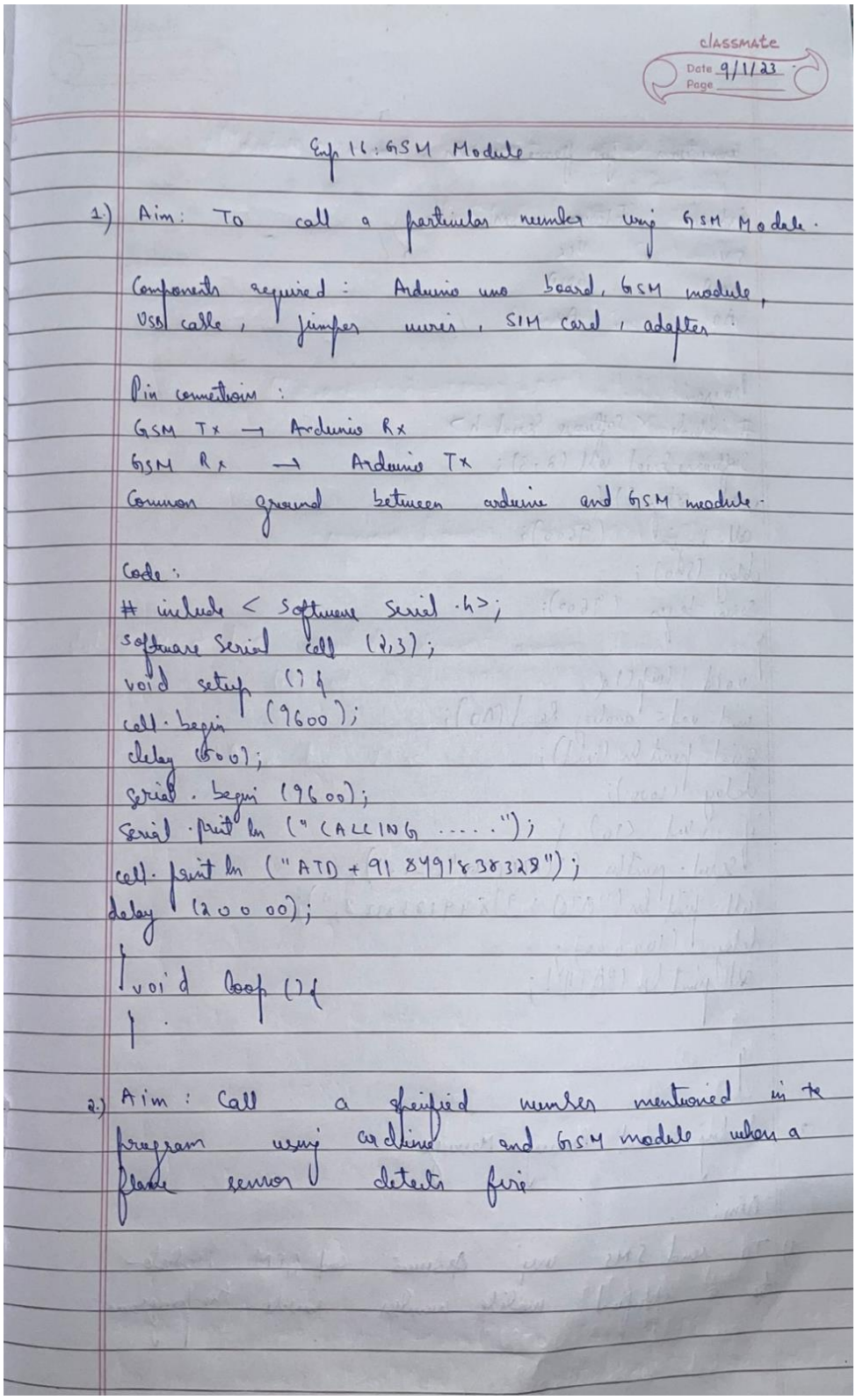
Aim: Call using Arduino and GSM Module – to a specified mobile number inside the program.

Circuit Diagram / Pin connection:

- GSM Tx –>Arduino Rx (Here pin 2)
- GSM Rx –>Arduino Tx. (Here pin 3)
- Make the ground common between Arduino and GSM modem.



Handwritten code pic:



classmate
Date 9/1/23
Page

Exp 16: GSM Module

1) Aim: To call a particular number using GSM Module.

Components required: Arduino uno board, GSM module, USB cable, jumper wires, SIM card, adapter.

Pin connections:
GSM Tx → Arduino Rx
GSM Rx → Arduino Tx
Common ground between arduino and GSM module.

Code:

```
# include < Softwareserial.h>;
softwareSerial call (2,3);
void setup (){
call.begin (9600);
delay (500);
Serial.begin (9600);
Serial.println ("CALLING ....");
call.println ("ATD+91 8499183832 8");
delay (20000);
}void loop (){
}
```

2) Aim: Call a specified number mentioned in the program using arduino and GSM module when a flame sensor detects fire

Code:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3);  // (Rx, Tx)

void setup() {
cell.begin(9600);
delay(500);
Serial.begin(9600);
Serial.println("CALLING.....");
cell.println("ATD+9538433364;");
delay(20000);
}
void loop() {

}
```

Call to a particular number on an alert

Aim: Call a specified mobile number mentioned in the program using Arduino and GSM Module when a flame sensor detects “fire”.

Connections for flame sensor:

Arduino	**Flame Sensor**
5V	VCC
GND	GND
A0	A0

Handwritten code pic:

connections for flame sensor:

Arduino	Flame sensor
5V	Vcc
GND	GND
A0	A0

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3);
void setup(){
cell.begin(9600);
delay(500);
Serial.begin(9600);
}
void loop(){
int val = analogRead(A0);
Serial.println(val);
delay(1000);
if(val<50){
Serial.println("CALLING");
cell.println("ATD+918499183832;");
delay(100000);
cell.println("ATH");
}
}
```

3. Sending and Receiving Message.

Aim:

1) To send SMS using Arduino and GSM Module to a specified mobile number using the program.

Code:

```
#include <SoftwareSerial.h>
SoftwareSerialcell(2,3);
void setup() {
cell.begin(9600);
delay(500);
Serial.begin(9600);
}
void loop() {
Int val=analogRead(A0);
Serial.println(val);
delay(1000);
if (val<50)
  {
Serial.println("CALLING.....");
cell.println("ATD+919742980606;");
delay(10000);
cell.println("ATH");  // Attention Hook Control
   }
}
```

Sending and Receiving Message

Aim:
1) Send SMS using Arduino and GSM Module – to a specified mobile number inside the program
2) Receive SMS using Arduino and GSM Module – to the SIM card loaded in the GSM Module.
Handwritten code pic:

7) Receive SMS using Arduino and GSM Module - to the SIM card loaded in the GSM Module.

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial myserial(2,3);
void setup()
{
myserial.begin(9600);
Serial.begin(9600);
delay(100);
}
void loop()
{
if (Serial.available()>0)
switch(Serial.read())
{
case 's':
SendMessage();
break;
case 'r':
ReceiveMessage();
break;
}
if (myserial.available()>0)
Serial.write(myserial.read());
}
void SendMessage()
{
myserial.println("AT+CMGF=1");
delay(1000);
myserial.println("AT+CMGS=\"+918491838328\"\r");
delay(1000);
```

```
mySerial.println("I am SMS from GSM Module");
delay(100);
mySerial.println((char)26);
delay(1000);
}
void ReceiveMessage()
{
mySerial.println("AT+CNMI=2,2,0,0,0");
delay(1000);
}
```

4. Controlling LED through received messages:

Aim:
Use received message through Arduino and GSM Module to control switching ON/OFF the LED.

Connection: Attach LED to pin 13 and GND.

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial call(2,3);
void read_sms()
{
if(call.available())
while(call.available()){
Serial.write(call.read());
}
}
void setup(){
pinMode(13,OUTPUT);
Serial.begin(9600);
```

Code:

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);

void setup()
{
mySerial.begin(9600);   // Setting the baud rate of GSM Module
Serial.begin(9600);    // Setting the baud rate of Serial Monitor (Arduino)
delay(100);
}


void loop()
{
if (Serial.available()>0)
switch(Serial.read())
  {
case 's':
SendMessage();
break;
case 'r':
RecieveMessage();
break;
  }

if (mySerial.available()>0)
Serial.write(mySerial.read());
}

void SendMessage()
{
mySerial.println("AT+CMGF=1");
delay(1000);  // Delay of 1000 milli seconds or 1 second
mySerial.println("AT+CMGS=\"+919742980606\"\r");

delay(1000);
mySerial.println("I am SMS from GSM Module");

delay(100);
mySerial.println((char)26);
delay(1000);
}

Void RecieveMessage()
{
mySerial.println("AT+CNMI=2,2,0,0,0");
delay(1000);
 }
```

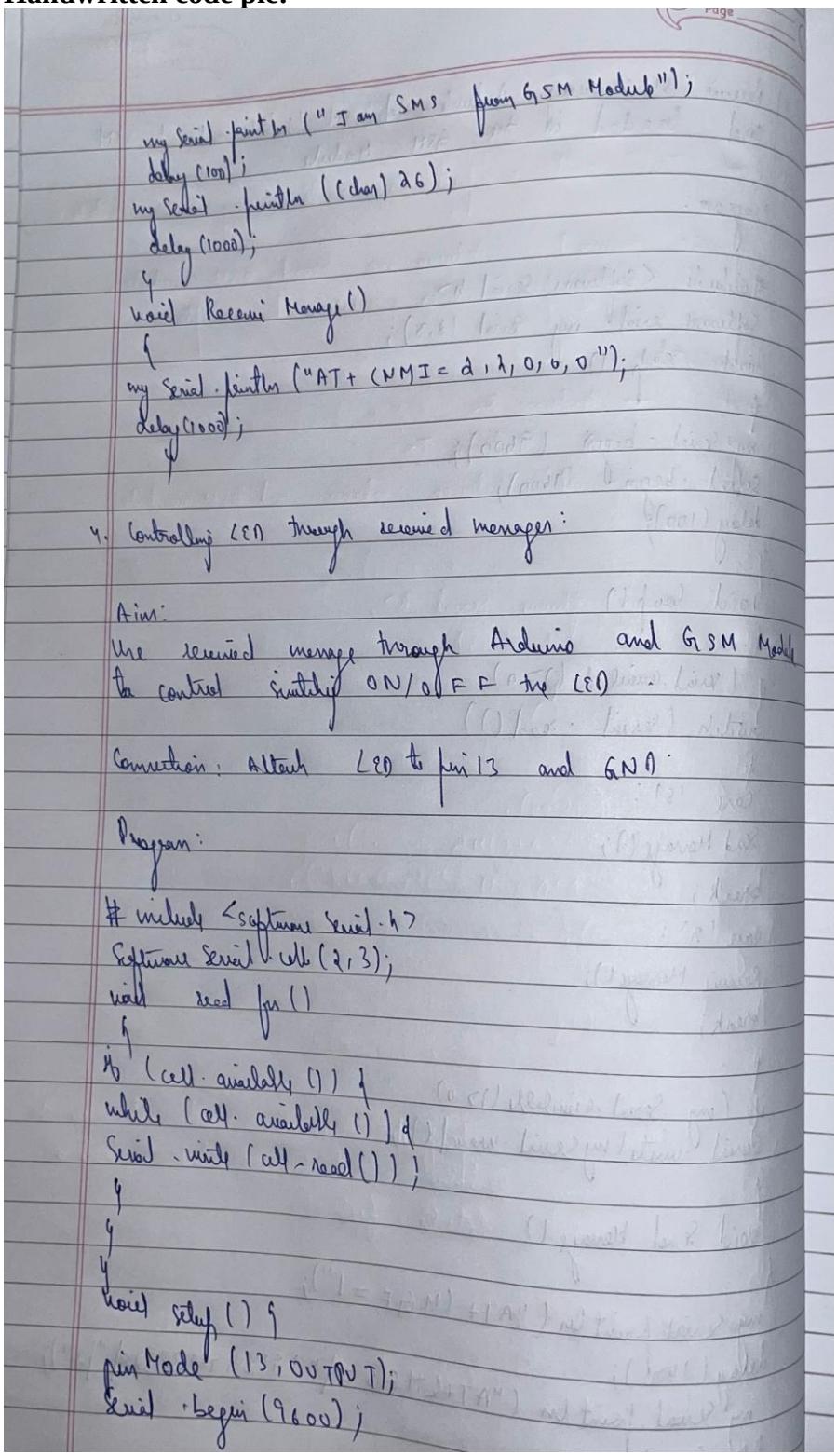
Controlling LED through received messages:

Aim:

Use received message through Arduino and GSM Module to control Switching ON / OFF the LED.

Connection: Attach LED to pin 13 and GND.

Handwritten code pic:



```
mySerial.println("I am SMS from GSM Module");
delay(100);
mySerial.println((char)26);
delay(1000);
}
void ReceiveMessage()
{
mySerial.println("AT+CNMI=2,2,0,0,0");
delay(1000);
}
```

4. Controlling LED through received messages:

Aim:

Use received message through Arduino and GSM Module to control switching ON/OFF the LED.

Connection: Attach LED to pin 13 and GND.

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2,3);
void readSms()
{
if(mySerial.available()){
while(mySerial.available()){
Serial.write(mySerial.read());
}
}
}
void setup(){
pinMode(13,OUTPUT);
Serial.begin(9600);
```

```
cell.begin(9600);
cell.println("AT");
delay(1000);
send fn();
cell.println("AT+CNMI=1,2,0,0,0");
}
void loop(){
if(cell.available())
{
String message = cell.readString();
Serial.println(message);
if(message.indexOf("SWITCH ON")>0)
{
digitalWrite(13,HIGH);
}
else if(message.indexOf("SWITCH OFF")>0)
{
digitalWrite(13,LOW);
}
else
{Serial.println("Nothing to do...");
}
}
}
```

Code:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3);

void readfn()
  {
if (cell.available()) {
while (cell.available()) {
Serial.write(cell.read());
     }
      }
  }

void setup() {
pinMode(13,OUTPUT);
Serial.begin(9600);
cell.begin(9600);
cell.println("AT");
delay(1000);
readfn();
  //New SMS alert
cell.println("AT+CNMI=1,2,0,0,0");
 }


void loop() {
if(cell.available())
 {
  String message =cell.readString();
Serial.println(message);
if(message.indexOf("SWITCH ON")>0)
  {
digitalWrite(13,HIGH);
  }
else if(message.indexOf("SWITCH OFF")>0)
  {
digitalWrite(13,LOW);
  }
else
  {
Serial.println ("Nothing to do...");
  }
}
}
```