

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



LAB REPORT

on

INTERNET OF THINGS LAB

Submitted by

Brijesh Gowda N (1BM21CS040)

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

JUN-2023 to SEP-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 Lab” carried out by **Brijesh Gowda N(1BM21CS040)**, who is 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 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **Internet of things lab - (22CS5PCIOT)** work prescribed for the said degree.

Mrs. Prameetha Pai

Assistant professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak

Professor and Head
Department of CSE
BMSCE, Bengaluru

Index

Sl. No.	Date	Program Title	Page No.
1.	13/11/23	LED Blinking	1
2.	13/11/23	LED ON/OFF Using Pushbutton	2
3.	13/11/23	LED Fading using Potentiometer	3
4.	20/11/23	Nightlight Simulation	7
5.	20/11/23	PIR with Arduino UNO	9
6.	20/11/23	Ultrasound with Arduino UNO	10
7.	27/11/23	Fire Alert System	12
8.	27/11/23	Automatic irrigation controller simulation	14
9.	3/1/24	Reading the code present on RFID tag	16
10.	3/1/24	Access control through RFID	17
11.	10/1/24	HC-05 Bluetooth at Command prompt	20
12.	10/1/24	HC-05 Bluetooth Controlled by mobile	21
13.	10/1/24	Bluetooth-Master Slave	22
17.	17/1/24	GSM-Module	24

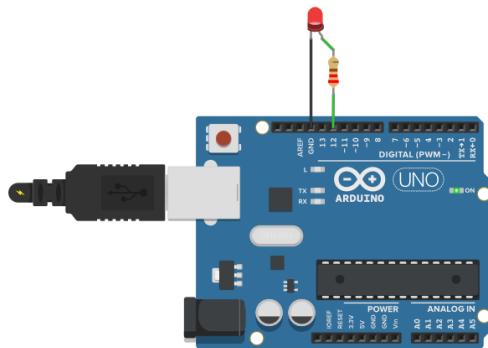
LED Blinking

Aim: Turns on an LED on for one second, then off for one second, repeatedly

Hardware Required:

- Arduino Board
- LEDs

Connection:

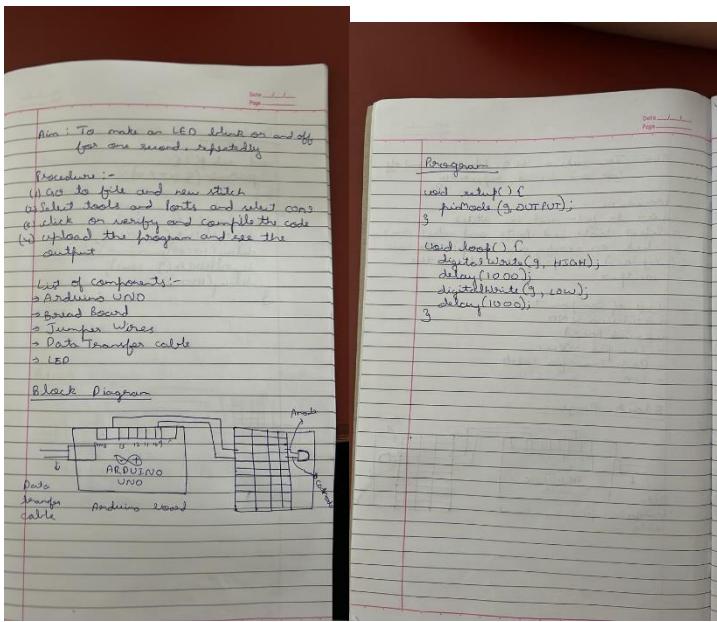


Code:

```
// Pin 13 has an LED connected on most Arduino boards
int led = 13;

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

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



Program

```

void setup() {
  pinMode(9, OUTPUT);
}

void loop() {
  digitalWrite(9, HIGH);
  delay(1000);
  digitalWrite(9, LOW);
  delay(1000);
}
  
```

3

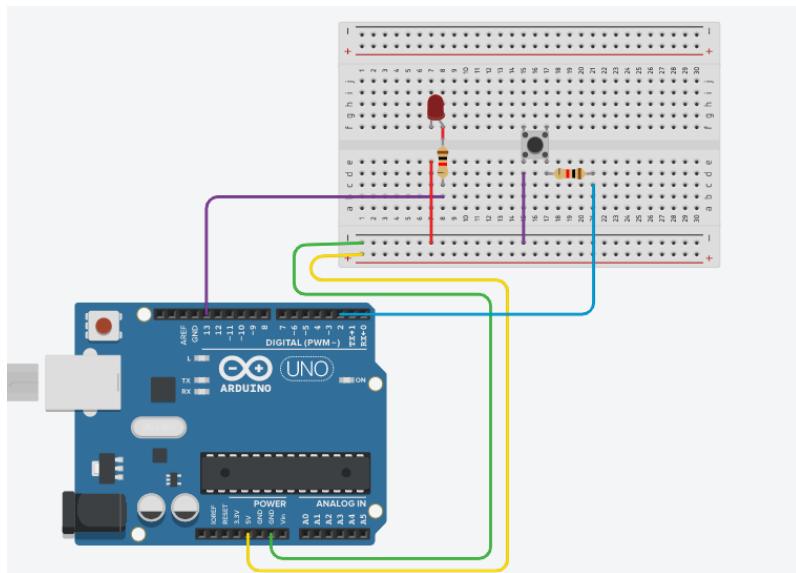
LED ON/OFF Using Pushbutton

Aim: Turn an LED ON /OFF using a Pushbutton.

Hardware Required:

- Arduino Board
- LED
- Push button

Connection:



Code:

```
const int buttonPin = 2;  
const int ledPin = 13;  
  
int buttonState = 0;  
  
void setup() {  
    pinMode(ledPin, OUTPUT);  
    pinMode(buttonPin, INPUT);  
}  
  
void loop() {  
    buttonState = digitalRead(buttonPin);  
    if (buttonState == HIGH) {  
        digitalWrite(ledPin, HIGH);  
    } else {digitalWrite(ledPin, LOW); } }
```

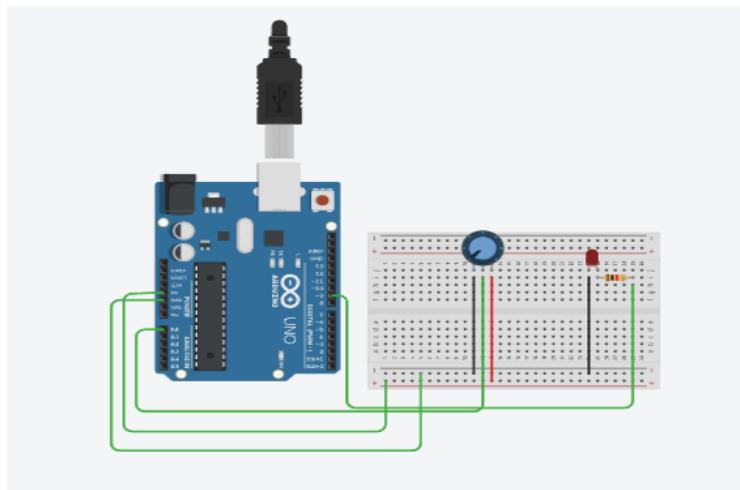
LED Fading using Potentiometer

Aim: To control the brightness of an LED using a Potentiometer.

Hardware Required:

- Arduino Board
- LED
- Potentiometer

Connection:



Code:

```
const int potPin = A0; // Pin connected to the potentiometer
const int ledPin = 9; // Pin connected to the LED
void setup() {
  pinMode(ledPin, OUTPUT); // Initialize the LED pin as an output
}
void loop() {
  int potValue = analogRead(potPin); // Read the value from the potentiometer (0-1023)
  int brightness = map(potValue, 0, 1023, 0, 255); // Map the potentiometer value to brightness
  // (0-255)
  analogWrite(ledPin, brightness); // Set the brightness of the LED
}
```

Aim: To make a LED fade

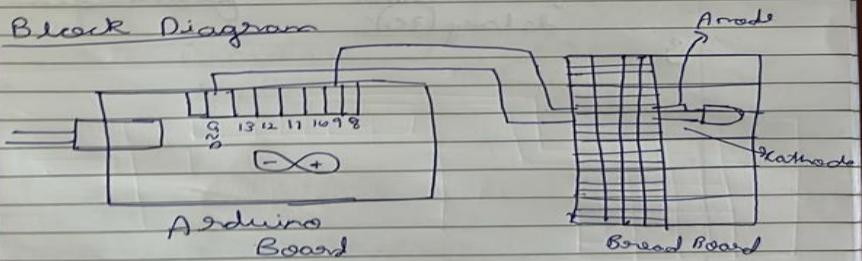
Procedure:-

- (1) Go to file and new sketch
- (2) After connecting to COM3 port, upload program and observe output on the Arduino Uno

List of Components

- Arduino UNO
- Bread Board
- LED
- Jumper wires
- Data Transfer cable

Block Diagram



Program

```
int led = 9;  
int brightness = 0;  
int fadeAmount = 5;
```

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

```
void loop() {  
    analogWrite(led, brightness);  
    brightness += fadeAmount;  
    if (brightness <= 0 || brightness >= 255)  
        {  
            fadeAmount = -fadeAmount; // Change direction  
            delay(30);  
        }  
}
```

Nightlight Simulation

Aim: Simulating a night light using LDR and PIR

Hardware Required:

- 1 LED
- 1 LDR
- 110K register

Connection:

1. Attach one leg of LDR to 5V and another leg to Arduino Analog pin A0
2. Attach one leg of 110K register with that leg of LDR connected to A0
3. Attach another leg of register to the ground
4. Connect the positive leg of LED to pin 11 and negative to GND

Code:

```
int LDR = 0; //analog pin to which LDR is connected, here we set it to 0 so it means A0
int LDRValue = 0; //that's a variable to store LDR values
int light_sensitivity = 500; //This is the approx value of light surrounding your LDR
void setup()
{
    Serial.begin(9600); //start the serial monitor with 9600 baud
    pinMode(11, OUTPUT); //attach positive leg of LED to pin 11
}
void loop()
{
    LDRValue = analogRead(LDR); //reads the ldr's value through LDR
    Serial.println(LDRValue); //prints the LDR values to serial monitor
    delay(50); //This is the speed by which LDR sends value to arduino
    if (LDRValue < light_sensitivity)
    {
        digitalWrite(11, HIGH);
    }
}
```

```
else
{
    digitalWrite(11, LOW);
}

delay(1000);

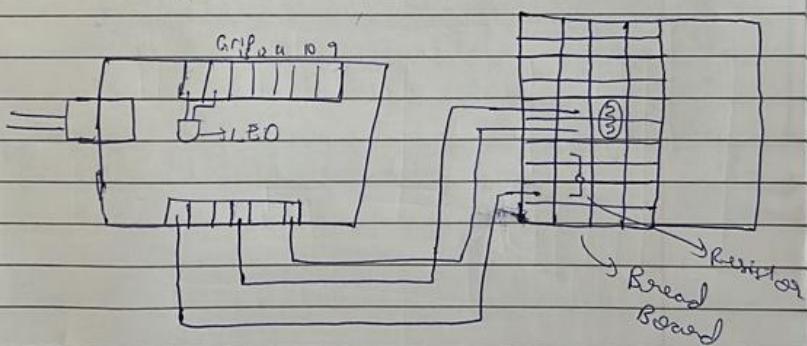
}
```

Aim :- To automate LED using Light Dependent Resistor (LDR)

List of Components:-

- Arduino board
- LED
- BreadBoard
- Resistor (1KΩ)
- LDR

Block Diagram



Program :-

```
int LDR = 0;  
int LDRvalue = 0;  
int light_sensitivity = 200;  
void setup()  
{  
    Serial.begin(9600);  
    pinMode(13, OUTPUT);  
    pinMode(A0, INPUT);  
}
```

void loop()

{

LDRvalue = analogRead(LDR);

Serial.println(LDR value);

if (LDR value < light sensitivity)

{

digitalWrite(13, HIGH);

else

{

digitalWrite(13, LOW);

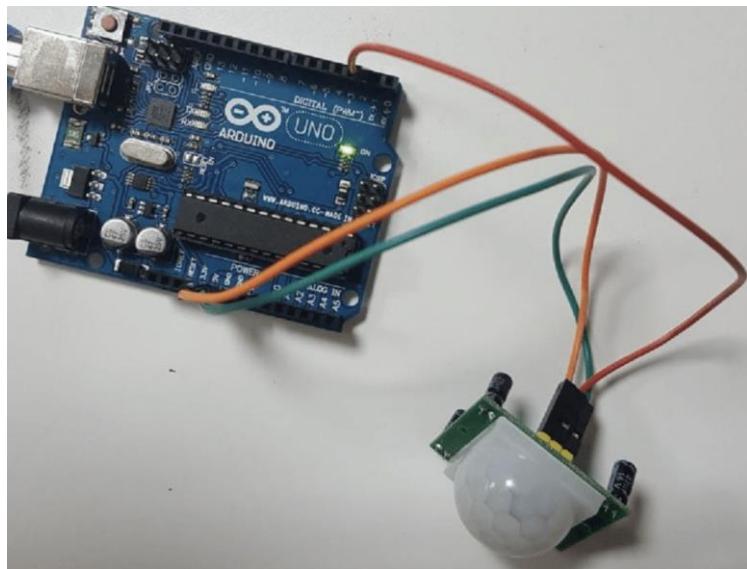
}

g

PIR with Arduino UNO

Aim: To Simulate infrared detection using PIR sensor and arduino UNO.

Connection:



Code:

```
int sensorState = 0;

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

void loop()
{
    // read the state of the sensor/digital input
    sensorState = digitalRead(2);

    // check if sensor pin is HIGH. if it is, set the
    // LED on.

    if (sensorState == HIGH) {
        digitalWrite(13, HIGH);
    }
}
```

```
Serial.println("Sensor activated!");
} else {
    digitalWrite(13, LOW);
}
delay(10);
}
```

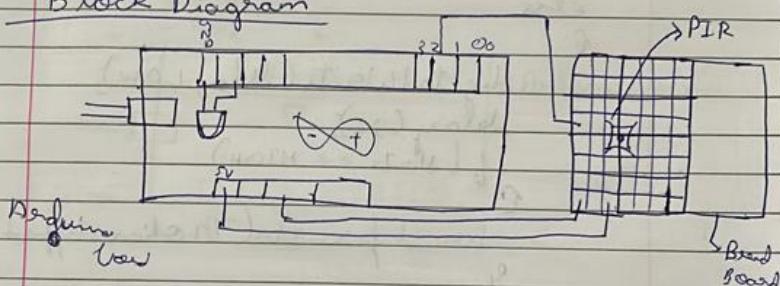
Date _____
Page _____

Aim:- To automate LED using
Passive infrared sensor
(PIR)

List of components

- Arduino Uno
- LED
- Bread Board
- Passive infrared sensor

Block Diagram



Program:-

```
int led = 13;  
int sensor = 2;  
int state = LOW;  
int val = 0;  
void setup()  
{  
    pinMode(led, OUTPUT);  
    pinMode(sensor, INPUT);  
    Serial.begin(9600);  
}  
void loop()  
{
```

```
val = digitalRead (sensor);
if (val == HIGH)
{
    digitalWrite (led, HIGH);
    delay (30);
    if (state == LOW)
    {
        Serial.println ("Motion detected");
        state = HIGH;
    }
}
else
{
    digitalWrite (led, LOW);
    delay (30);
    if (state == HIGH)
    {
        Serial.println ("Motion stopped");
    }
}
```

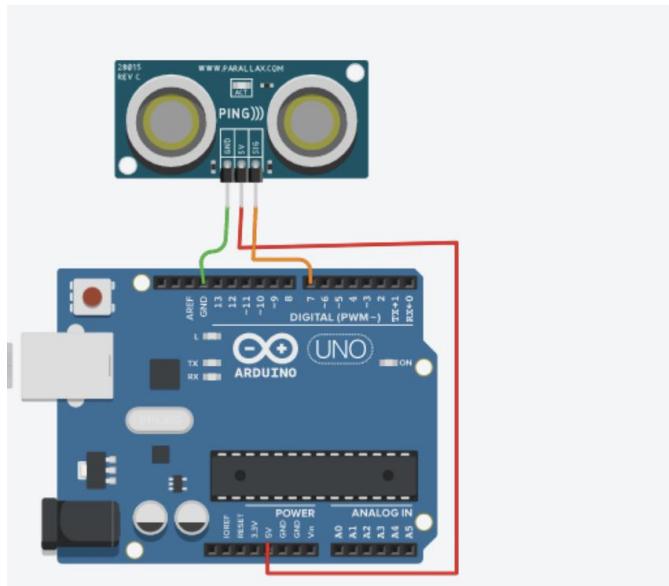
Ultrasound with Arduino UNO

Aim: To measure distance using ultrasound sensor and Arduino UNO.

Hardware Required:

- Ultrasound sensor
- Arduino UNO
- Jump/Connection wires

Connection:



Code:

```
const int pingPin = 7;  
  
const int echoPin=6;// Trigger Pin of Ultrasonic Sensor const int echoPin = 6; // Echo Pin of  
Ultrasonic Sensor  
  
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 = pulseIn(echoPin, HIGH);
    inches = microsecondsToInches(duration);
    Serial.print(inches);
    Serial.print("inches");
    cm = microsecondsToCentimeters(duration);
    Serial.print(cm);
    Serial.println("cm");
}

long microsecondsToInches(long microseconds) {
    return microseconds / 74 / 2;
}

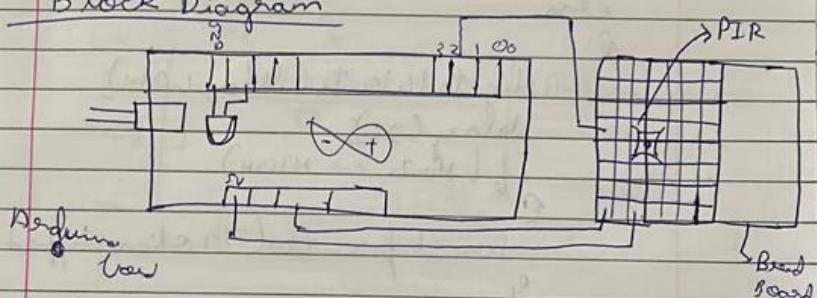
long microsecondsToCentimeters(long microseconds) {
    return microseconds / 29 / 2;
}
```

Aim:- To automate LED using
passive infrared sensor
(PIR)

List of components

- Arduino Uno
- LED
- Bread Board
- Passive infrared sensor

Block Diagram



Program:-

```
int led = 13;  
int sensor = 2;  
int state = LOW;  
int val = 0;  
void setup()  
{  
    pinMode(led, OUTPUT);  
    pinMode(sensor, INPUT);  
    Serial.begin(9600);  
}  
void loop()  
{
```

```
val = digitalRead(sensor);
if (val == HIGH)
{
    digitalWrite(led, HIGH);
    delay(30);
    if (state == LOW)
    {
        Serial.println("Motion detected");
        state = HIGH;
    }
}
else
{
    digitalWrite(led, LOW);
    delay(30);
    if (state == HIGH)
    {
        Serial.println("Motion stopped");
    }
}
```

Fire Alert

Aim: Fire alarm simulation

Hardware Required:

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

Connections:

- **Flame sensor interfacing to Arduino**

Flame sensor to Arduino

vcc to vcc

gnd to gnd

A0 to A0

- **Led interfacing to Arduino**

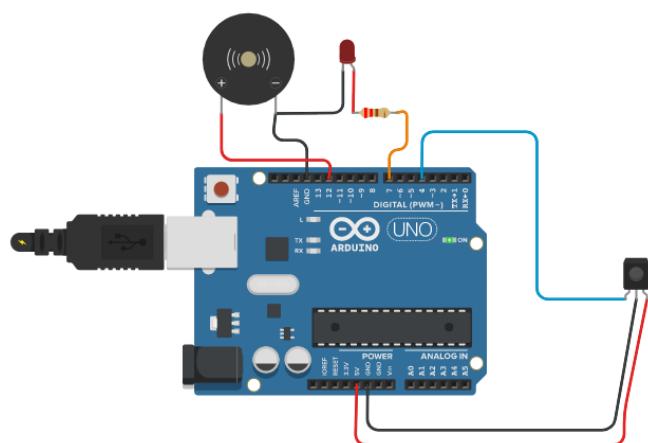
LED +ve is connected to 9th pin of Arduino

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:

```
int sensorPin = A0; // select the input pin for the LDR
int sensorValue = 0; // variable to store the value coming from the sensor
int led = 9; // Output pin for LED
int buzzer = 12; // Output pin for Buzzer
void setup() {
    // declare the ledPin and buzzer as an OUTPUT:
    pinMode(led, OUTPUT);
    pinMode(buzzer,OUTPUT);
    Serial.begin(9600);
}
void loop()
{
    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);
}
```

Programs

Fire Alert

classmate
Date _____
Page _____

Aim :- Fire alarm simulation

Hardware Required :-

- Flame sensor
- Arduino
- Bread Board
- LED
- Buzzer
- Connecting wires

Circuit Connection :-

Flame sensor to Arduino

Vcc → VCC

gnd → gnd

A0 → A0

LED interfacing to Arduino

LED +ve is connected to 9th pin

LED -ve is connected to gnd

Buzzer Interfacing to Arduino

Buzzer +ve is connected to 12th pin

Buzzer -ve is connected to gnd.

Code :-

```
int sensorPin = A0;
```

```
int sensorValue = 0;
```

```
int led = 9;
```

```
int buzz = 12;
```

```
void setup()
```

```
pinMode (led, OUTPUT);
```

```
pinMode (buzz, INPUT);
```

```
Serial.begin (9600);
```

Automatic irrigation controller simulation

Aim:

Sensing the soil moisture and sprinkling the Water simulation

Hardware Required:

- Arduino
- Moisture Sensor
- Breadboard
- Min servo motor

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; // create servo object to control a servo
// twelve servo objects can be created on most boards
int pos = 0; // variable to store the servo position
int sensorPin = A0; // select the input pin for the potentiometer
int sensorValue = 0; // variable to store the value coming from the sensor
void setup() {
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
  Serial.begin(9600);
}
void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  Serial.println (sensorValue);
  if(sensorValue<500)
```

```
{  
for (pos = 0; pos < 180; pos += 1) { // goes from 0 degrees to 180 degrees  
// in steps of 1 degree  
myservo.write(pos);  
delay(15); // waits 15ms for the servo to reach the position  
}  
for (pos = 180; pos < 0; pos -= 1) { // goes from 180 degrees to 0 degrees  
myservo.write(pos); // tell servo to go to position in variable 'pos'  
delay(15); // waits 15ms for the servo to reach the position  
}  
delay (1000);  
}
```

Program 9 Automatic Irrigation Controller (Simple)

Aim:
Sensing the soil moisture & spraying the water
simulation.

Hardware Required:

- Arduino
- Moisture sensor
- Bread Board
- Mini servo motor

Circuit 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

Code:

```
#include <Servo.h>
Servo myServo;
int pos = 0;
int sensorPin = A0;
int sensorValue = 0;
```

setup()

```
myServo.attach(9);
Serial.begin(9600);
```

}

Reading the code present on RFID tag

Aim:

The following code will read the code present on RFID tag and print it in serial monitor.

Connection:

5V-Arduino 5V

GND-Arduino GND

Tx-pin 9

Code:

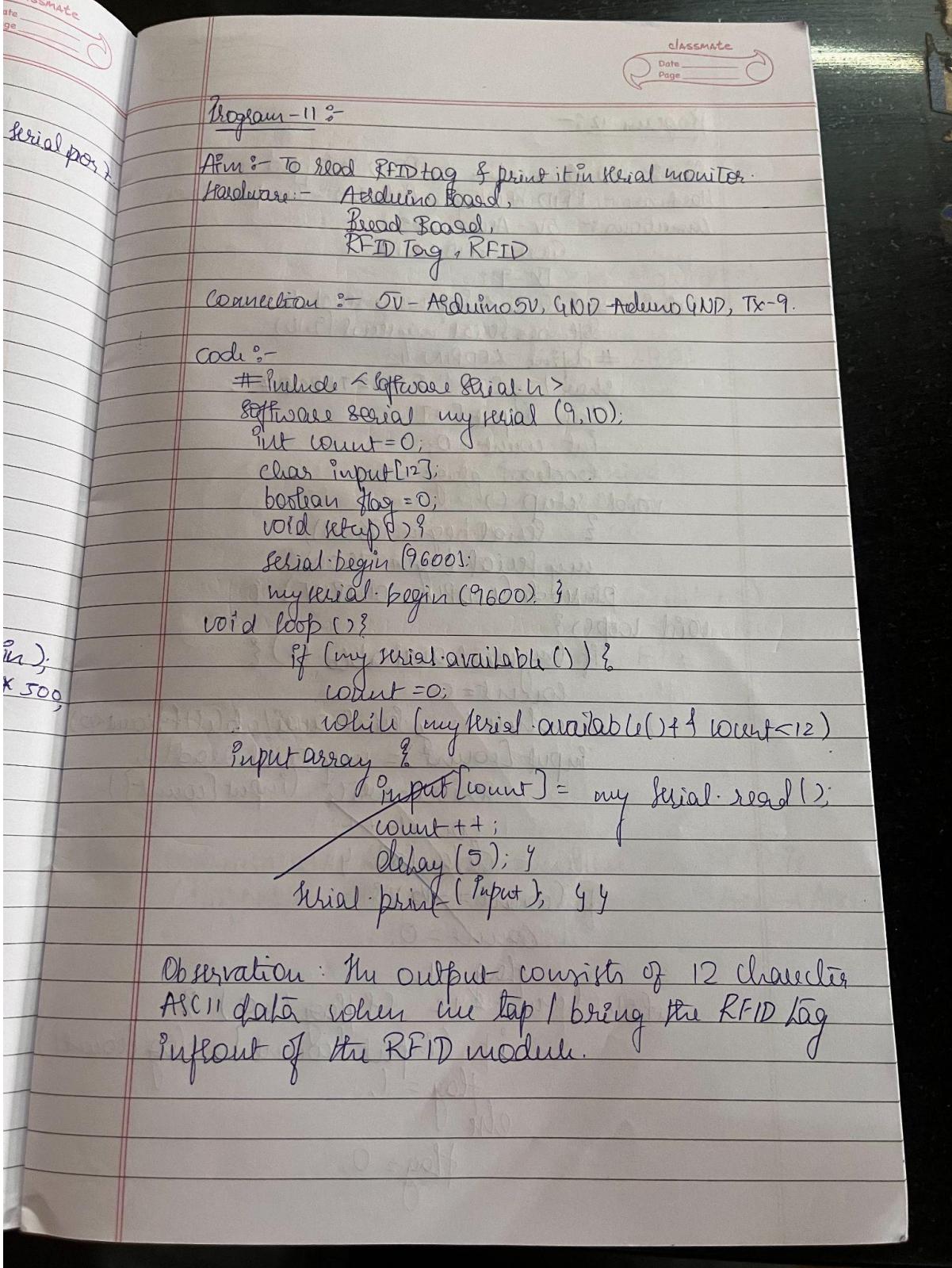
```
#include<SoftwareSerial.h>

SoftwareSerial mySerial(9, 10);

int count = 0;
char input[12];
boolean flag = 0;

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

void loop() {
    if(mySerial.available())
    {
        count = 0;
        while(mySerial.available() && count < 12)
        {
            input[count] =mySerial.read();
            count++;
            delay(5);
        }
        Serial.print(input);
    }
}
```



Aim:

The following code will read the code present on RFID tag tapped. If the code matches with the previously known tag (configured in the code), it will grant access (here LED will glow), otherwise access will be denied.

Connection:

5V-Arduino 5V

GND-Arduino GND

Tx-pin 9

Led-pin 12

Code:

```
#include<SoftwareSerial.h>
SoftwareSerial mySerial(9, 10);
#define LEDPIN 12
char tag[] = "5300292DD087;" // Replace with your own Tag ID
char input[12]; // A variable to store the Tag ID being presented
int count = 0; // A counter variable to navigate through the input[]
character array
boolean flag = 0; // A variable to store the Tag match status
void setup()
{
Serial.begin(9600);
mySerial.begin(9600);
pinMode(LEDPIN,OUTPUT); //WRONG TAG INDICATOR
}
void loop()
{
if(mySerial.available())// Check if there is incoming data in the RFID Reader Serial
Buffer.
{
count = 0;
```

```
while(mySerial.available() && count < 12)
{
    input[count] = mySerial.read();
    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;
        else
            flag=0;
        count++; // increment i
    }
}

if(flag == 1) // If flag variable is 1, then it means the tags match
{
    Serial.println("Access Allowed!");
    digitalWrite(LEDPIN,HIGH);
    delay (2000);
    digitalWrite (LEDPIN,LOW);
}

else
{
    Serial.println("Access Denied"); // Incorrect Tag Message
    digitalWrite(LEDPIN,LOW);
```

```
delay(2000);
}

for(count=0; count<12; count++)
{
    input[count]= 'F';
}

count = 0; // Reset counter variable
}
}
```

Program - 10

Aim: Read the temperature & print it in serial port

Hardware :
Arduino board
Temperature sensor
CLM35 temperature sensor

Connections :

5V → Vcc

A0 → Middle pin

GND → GND

Code :

```
int outputPin = 0;  
void setup () {
```

```
    Serial.begin (9600);  
}
```

```
void loop () {
```

```
    int rawVoltage = analogRead (OutputPin);
```

```
    float millivolts = (rawVoltage / 1024) * 500;
```

```
    float celsius = millivolts / 10;
```

```
    Serial.print (celsius);
```

```
    Serial.print ("degree celsius");
```

```
    Serial.print ((celsius * 9 / 5 + 32));
```

```
    Serial.print ("degree fahrenheit");
```

```
; delay (1000);
```

Observation :- Temperature is detected in the serial port.

HC-05 PinOut (Right) :

- KEY: If brought HIGH before power is applied, forces AT Command Setup Mode.

LED blinks slowly (2 seconds)

- VCC: +5 Power
- GND: System / Arduino Ground
- TXD: Transmit Serial Data from HC-05 to Arduino Serial Receive. NOTE: 3.3V
HIGH level: OK for Arduino
- RXD: Receive Serial Data from Arduino Serial Transmit
- STATE: Tells if connected or not

HC-05 at Command prompt:

Code:

(For this program to work, HC-05 must be in command mode)

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11); // RX | TX

void setup()
{
    Serial.begin(9600);
    Serial.println("Enter AT commands:");
    BTSerial.begin(38400); // HC-05 default speed in AT command mode
}

void loop()
{
    if (BTSerial.available())
        Serial.write(BTSerial.read());
    if (Serial.available())
        BTSerial.write(Serial.read());
}
```

HC-05 Controlled by mobile

Code:

(For this code to work, HC-05 must be in DATA mode and Arduino Bluetooth App)

```
#define ledPin 13

int state = 0;

void setup() {
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, LOW);
    Serial.begin(38400);
}

void loop() {
    if(Serial.available() < 0){
        // Checks whether data is comming from the serial port
        state = Serial.read(); // Reads the data from the serial port
    }

    if (state == "0") {
        digitalWrite(ledPin, LOW); // Turn LED OFF
        Serial.println("LED: OFF");
        state = 0;
    }

    else if (state == "1") {
        digitalWrite(ledPin, HIGH);
        Serial.println("LED: ON");
        state = 0;
    }
}
```

13. BT-Master Slave

BT-Slave Program:

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11); // RX | TX

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

    BTSerial.begin(38400); // HC-05 default speed in AT command mode
}

void loop() {
    if(Serial.available()) {
        String message = Serial.readString();

        Serial.println (message);
        BTSerial.write(message.c_str());
    }
}
```

BT-Master Program:

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11); // RX | TX

#define ledPin 9

String message;
int potValue = 0;

void setup() {
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, LOW);
    Serial.begin(9600);

    BTSerial.begin(38400); // HC-05 default speed in AT command mode
```

```
}

void loop() {
    if(BTSerial.available() < 0){
        message = BTSerial.readString();
        if(message.indexOf("SWITCH ON")<=0)
        {
            digitalWrite(ledPin, HIGH); // LED ON
        }
        else if(message.indexOf("SWITCH OFF")<=0)
        {
            digitalWrite(ledPin, LOW); // LED OFF
        }
        delay(100);
    }
    delay(10);
}
```

(iii) Aim:-

- 1) Send SMS using Arduino & GPRS module.
- 2) Receive SMS using Arduino & GPRS module.

Hardware:-

Arduino Board
GPRS module

Connection :- Same as (i).

Code:-

```
#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 'r':
                    sendMessage();
                    break;
                case 'l':
                    receiveMessage();
                    break;
            }
        if (mySerial.available() > 0)
            serial.write(mySerial.read());
    }
}
```

GSM Module

1. GSM Module: Call to a particular number

Aim:

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

Program:

```
#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;"); // ATD – Attention Dial
    delay(20000);
}

void loop() {
```

2. 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

A0

Program:

```
#include <SoftwareSerial.h>
```

```

SoftwareSerial cell(2,3);

void setup() {
cell.begin(9600);
delay(500);
Serial.begin(9600);
}

void loop() {
intval=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
}
}

```

3. 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.

Program:

Note: According to the code, message will be sent and received when ‘s’ and ‘r’ are pressed through serial monitor respectively.

```
#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"); //Sets the GSM Module in Text Mode //AT+CMGF,
    SMS Format
    delay(1000); // Delay of 1000 milli seconds or 1 second
    mySerial.println("AT+CMGS=\\"+919742980606\\r"); // AT+CMGS, Send Message
    // Replace with your mobile number
    delay(1000);
    mySerial.println("I am SMS from GSM Module");
}

```

```

// The SMS text you want to send
delay(100);
mySerial.println((char)26);
delay(1000);
}
voidRecieveMessage()
{
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 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...");
```

}}}

MASSNATE
Date _____
Page _____

Program 14:-

Aim: Working with Arduino Bluetooth Module

Hardware :-

HC-05 Bluetooth x 2
Arduino Board x 2

Connection :-

Bluetooth	Arduino
RxD	TxD
TxD	RxD
GND	GND
VCC	5V
En	3.3V

Code :-

HC-05 at command prompt :-

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial (10, 11);
void setup() {
    Serial.begin (9600);
    Serial.println ("Enter AT commands");
    BTSerial.begin (58400);
}
void loop() {
    if (BTSerial.available ()) {
        Serial.write (BTSerial.read ());
    }
    if (Serial.available ()) {
        BTSerial.write (Serial.read ());
    }
}
```

BT-Slave Program :-

```
#include <SoftwareSerial.h>
SoftwareSerial BTserial(10,11);
void setup() {
    Serial.begin(9600);
    BTserial.begin(38400);
}
void loop() {
    if (Serial.available()) {
        String message = Serial.readString();
        Serial.println(message);
        BTserial.write(message + C - Str());
    }
}
```

BT- master program :-

```
#include <SoftwareSerial.h>
SoftwareSerial BTserial(10,11);
#define ledPin 9
String message;
int potValue = 0;
void setup() {
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, LOW);
    Serial.begin(9600);
    BTserial.begin(38400);
}

void loop() {
    if (!BTserial.available() > 0) {
        message = BTserial.readString();
        if (message.indexOf("switch OFF") >= 0)
```

digitalWrite (ledPin, LOW);

else

 serial.println ("Nothing to do");

 delay (100);

 delay (10);

10/11/18

- mapping attenuator 18

< n long multiplex > - don't init
(10/04/18) long (10/05/18)

 possible with #

 resistor value?

 or delaying? - will

 (10/04/18) dual bio

 (10/04/18) with 2 counting

 (10/04/18) first threshold

 (10/04/18) second threshold

 (10/04/18) third threshold

 (10/04/18) fourth threshold

 (10/04/18) fifth threshold

 (10/04/18) sixth threshold

 (10/04/18) seventh threshold

 (10/04/18) eighth threshold

 (10/04/18) ninth threshold

 (10/04/18) tenth threshold

 (10/04/18) eleventh threshold

 (10/04/18) twelfth threshold

 (10/04/18) thirteenth threshold

 (10/04/18) fourteenth threshold

 (10/04/18) fifteenth threshold

 (10/04/18) sixteenth threshold