VISVESVARAYA TECHNOLOGICAL UNIVERSITY “JnanaSangama”, Belgaum -590014, Karnataka.

LAB REPORT
on

INTERNET OF THINGS LAB

Submitted by

RAHUL RAJ

(1BM21CS158)

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-2023 to Feb-2024

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 **RACHUL RAJ (1BM21CS158),** 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 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **Internet of things lab (21CS5PCIOT)** work prescribed for the said degree.

Dr.SEEMA PATIL

Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak

Professor and Head
Department of CSE
BMSCE, Bengaluru

TABLE OF CONTENTS

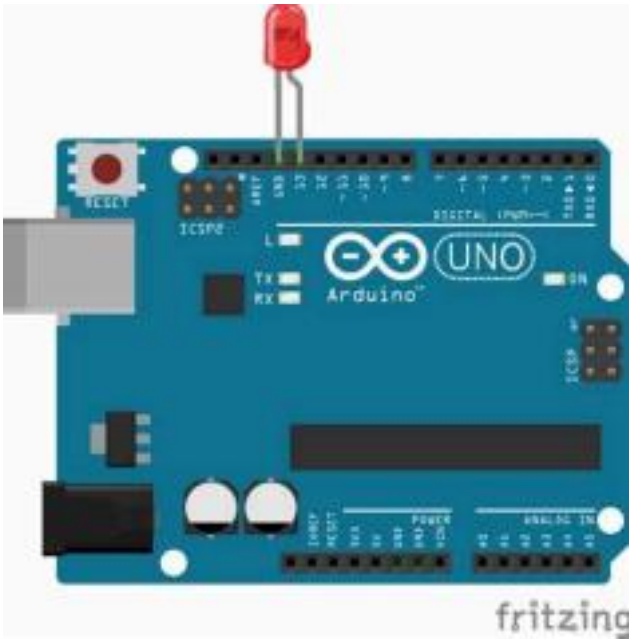
Program no: **01** Program Title: **LED BLINK** Date:23/11/2023 **Aim:** To control the LED using arduino (to turn ON/OFF LED)

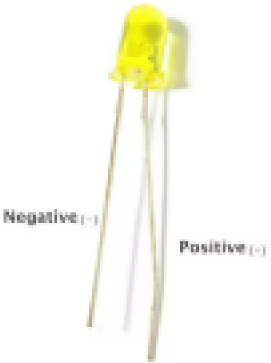
Hardware/components Required

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

Circuit Diagram / Pin connection

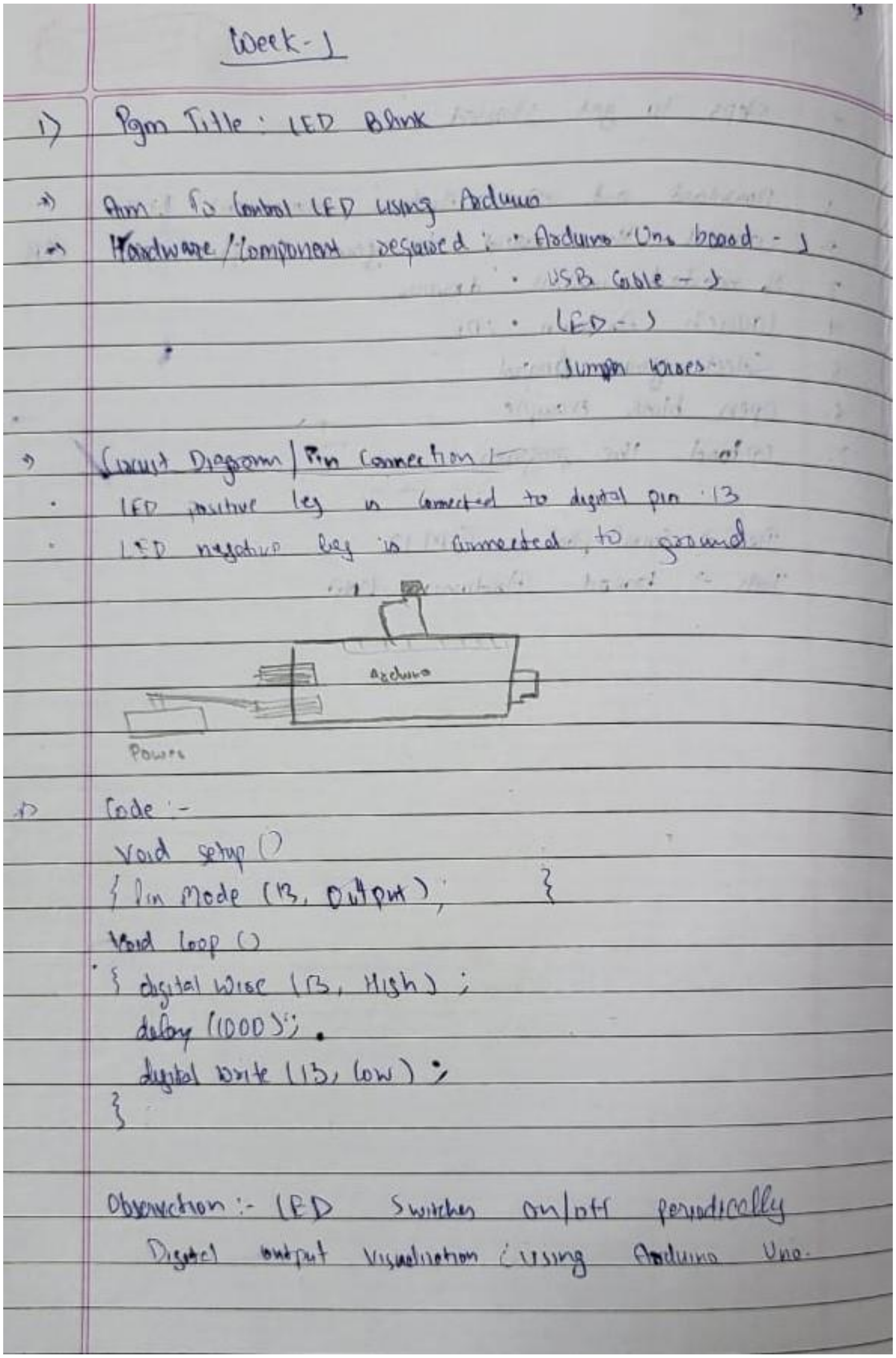
- LED's positive leg is connected to digital pin 13
- LED's negative leg is connected to ground





Code:

```
void setup()
 {
  // initialize digital pin 9 as an output.
  pinMode(13, OUTPUT);
 }

// the loop function runs over and over again forever
void loop()
 {
  digitalWrite(13, HIGH);
     delay(1000);
   digitalWrite(13, LOW); }
```



Week-1

1) Pgm Title : LED Blink

Aim : To control LED using Arduino

Hardware / Component required : · Arduino Uno board - 1
· USB Cable - 1
· LED - 1
· Jumper wires

Circuit Diagram / Pin Connection :-
· LED positive leg is connected to digital pin 13
· LED negative leg is connected to ground

Arduino
Power

Code :-

```
void setup ()
{ Pin Mode (13, Output);        }
Void loop ()
{ digital Wrie (13, High) ;
  delay (1000);
  digital write (13, low) ;
}
```

Observation :- LED Switches on/off periodically
Digital output visualization using Arduino Uno.

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

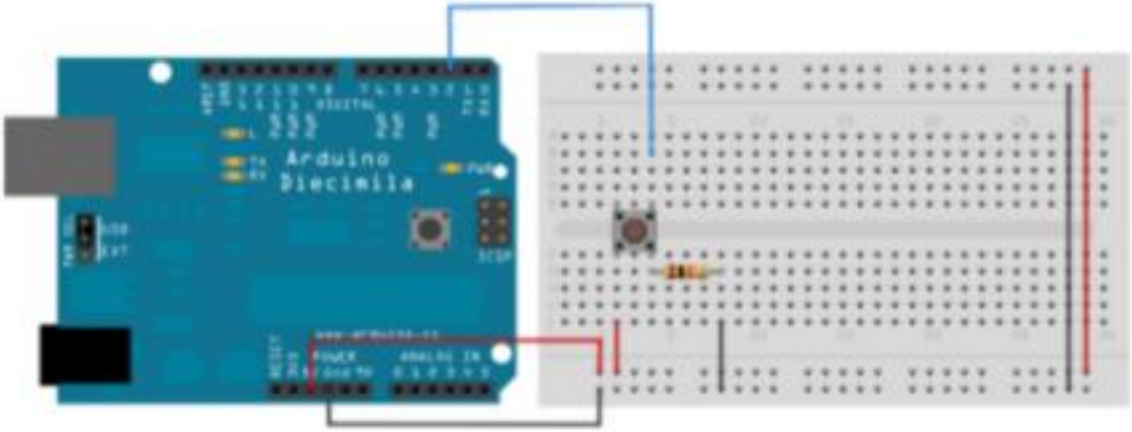
Program no: **02** Program Title: **LED ON/OFF** Date:23/11/2023

Aim: To turn an LED ON /OFF using a Pushbutton.

Hardware/components Required

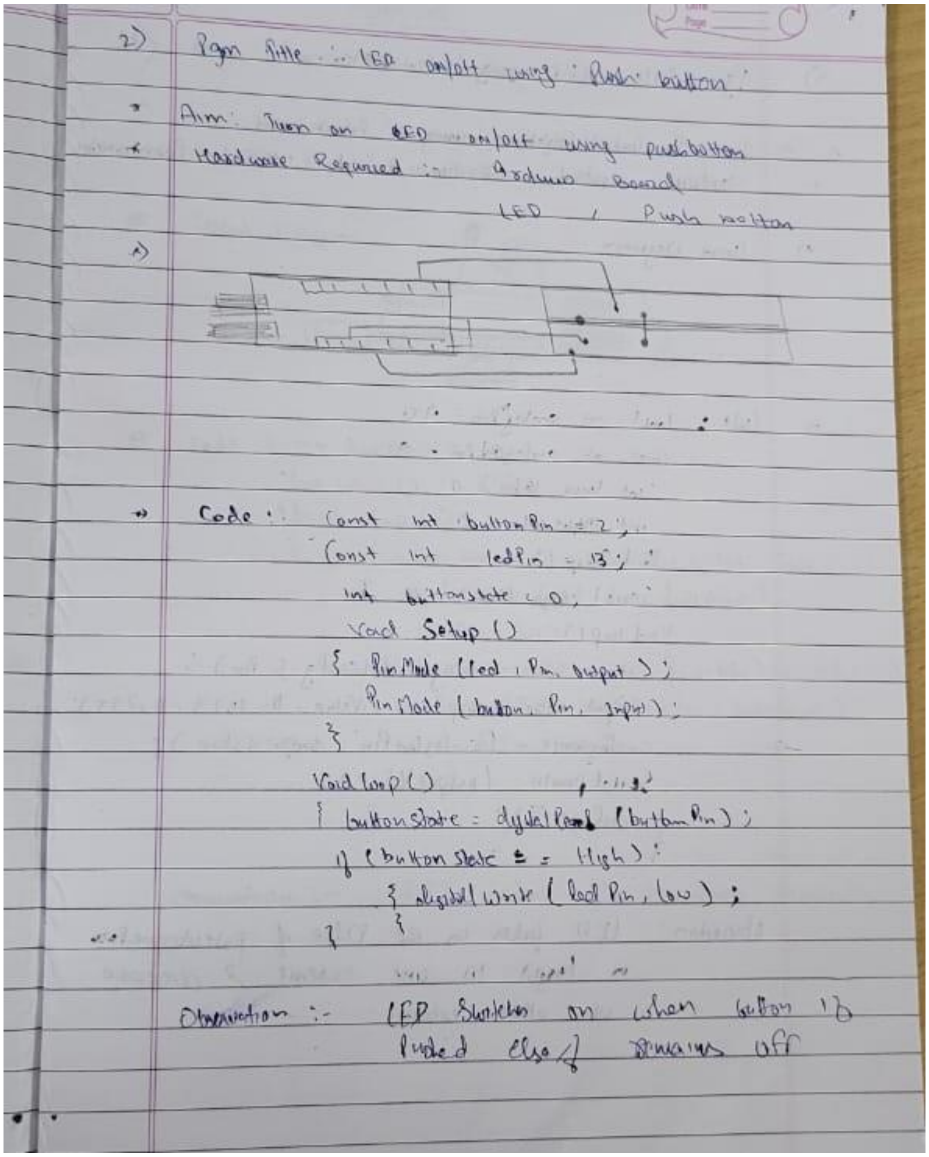
Arduino Uno board - 1
USB Cable - 1
LED - 1
Pushbutton
Jumper wires

Circuit Diagram / Pin connection



Code:

```
const int buttonPin=2;
const int ledPin=13;
int buttonState=0;
void setup()
 {.
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin,OUTPUT);
 }


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




  }

 }
```



2) Pgm title :- LED on/off using Push button

* Aim: Turn on LED on/off using pushbutton

* Hardware Required :- Arduino Board
LED , Push button

A)

Code :

```
Const int buttonPin = 2;
Const int ledPin = 13;
 int buttonstate = 0;
 Void Setup ()
{ PinMode (led, Pin, output);
  PinMode (button, Pin, Input);
}
Void loop ()
{ buttonState = digitalRead (buttonPin);
 if (button state = = High)
   { digitalWrite (led Pin, low);
   }
}
```

Observation :- LED Switches on when button is pushed else it remains off

Handwritten code pic:

Observation: LED turns ON when push button is pressed and turns OFF when it is released. Digital output visualization using Arduino Uno.

Program no: **03** Program Title: **LED FADING** Date:23/11/2023

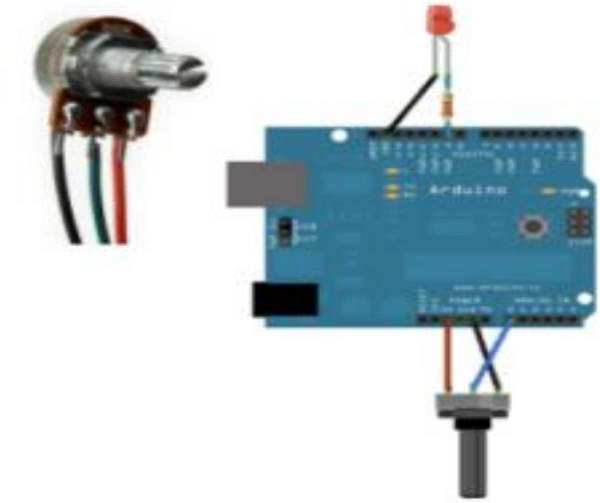
Aim: To control the brightness of an LED using aPotentiometer.

Hardware/components Required

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

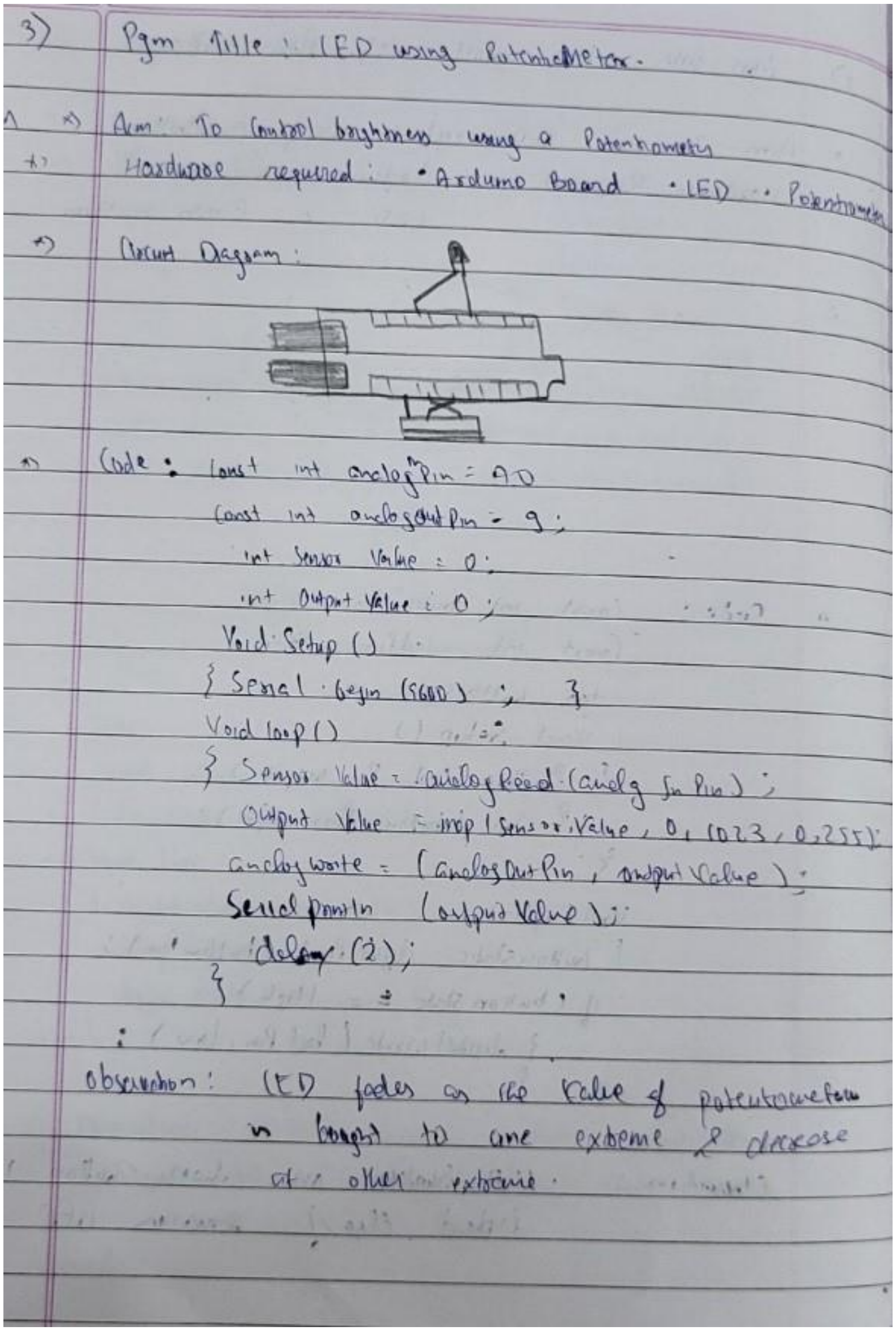
Circuit Diagram / Pin connection

LED positive to pin 9,LED negative to ground
Potentiometer: VCC - 5V , A0 -A0 , GND-GND



Code:

```
const int analogPin=A0;
const int analogOutPin=9;
int sensorValue=0;
int outputValue=0;
void setup(){
 Serial.begin(9600);
 }
void loop(){
 sensorValue=analogRead(analogPin);
outputValue=map(sensorValue,0,1023,0,255);



analogWrite(analogOutPin,outputValue);
Serial.print(sensorValue);
Serial.print(outputValue);
delay(2);
}
```



3) Pgm Title : LED using PotentioMeter.

*) Aim : To control brightness using a Potentiometer

*) Hardware required : • Arduino Board • LED • Potentiometer

*) Circuit Diagram :

*) Code :

```
const int analogInPin = A0
const int analogOutPin = 9;
  int Sensor Value = 0;
  int Output Value = 0;
Void Setup ()
{ Serial . begin (9600) ;   }
Void loop ()
{ Sensor Value = analogRead (analog In Pin) ;
  Output Value = map ( Sensor Value , 0, 1023 , 0, 255);
  analog write = ( analogOutPin , output Value );
  Serial println  (output Value );
  delay (2);
}
```

observation : LED fades as the value of potentiometer is brought to one extreme & decrease at other extreme.

Handwritten code pic:

Observation: Based on the potentiometer shaft rotation output varies.LED glows if we rotate towards right and fades if we rotate towards left..

Program no: **04** Program Title: **LED FADING** Date:23/11/2023

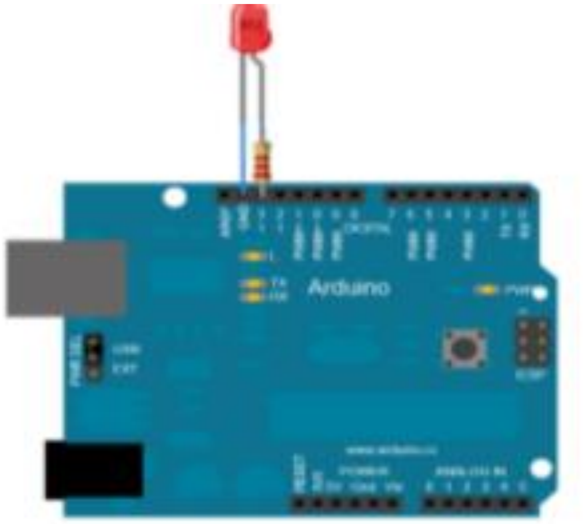
Aim: To control the brightness of an LED without using a Potentiometer.

Hardware/components Required

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

Circuit Diagram / Pin connection

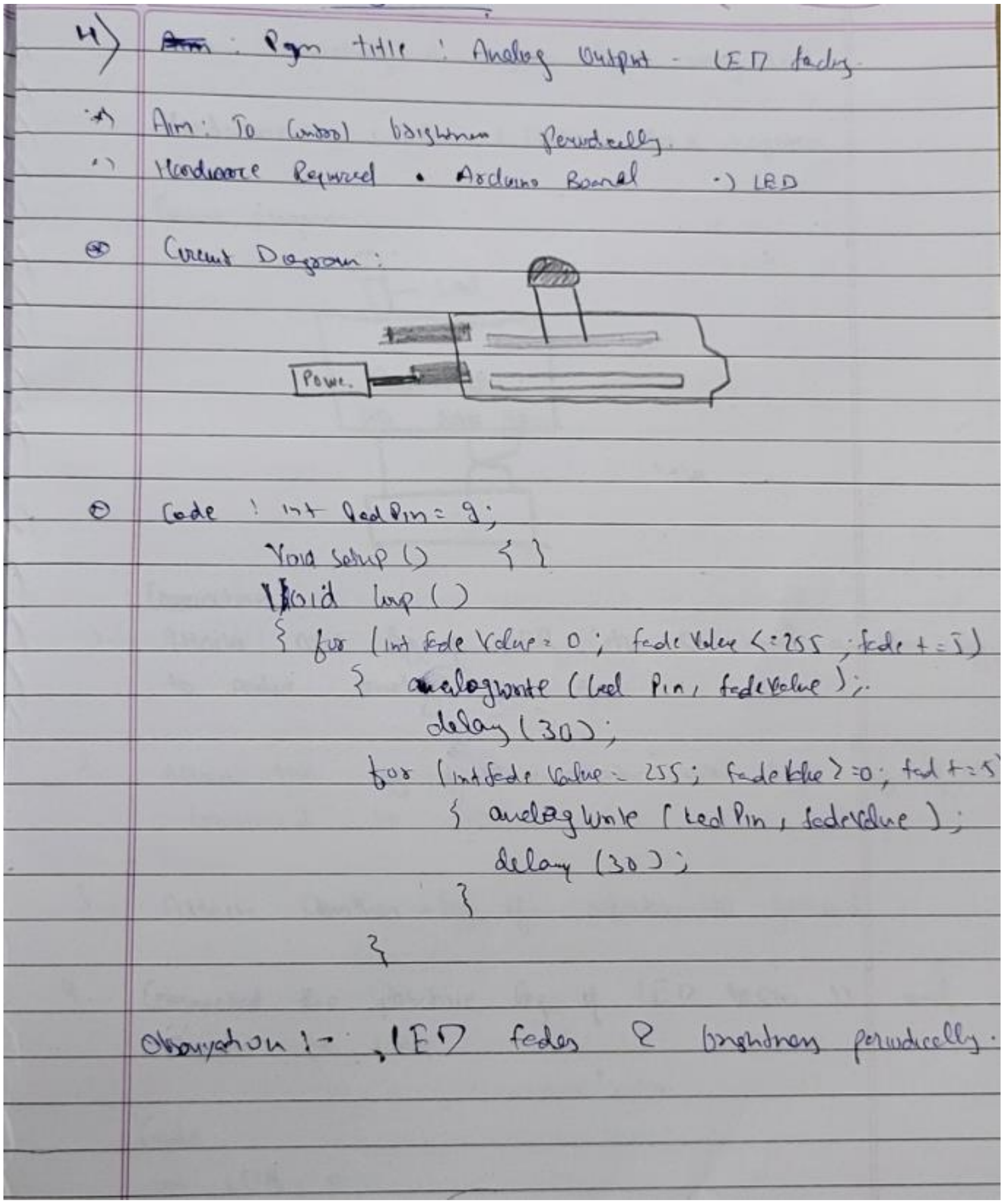
LED positive to pin 9 ,negative to ground



Code:

```
       int ledPin = 9; // LED connected to digital pin 9

       void setup() {

         }

       void loop()

        {

         // fade in from min to max in increments of 5 points:

         for (int fadeValue = 0 ; fadeValue <= 255; fadeValue += 5) {

           // sets the value (range from 0 to 255):

           analogWrite(ledPin, fadeValue);

           delay(30); // wait for 30 milliseconds to see the dimming effect

          }
// fade out from max to min in increments of 5 points:

for (int fadeValue = 255 ; fadeValue >= 0; fadeValue -= 5) {

// sets the value (range from 0 to 255):

analogWrite(ledPin, fadeValue);

delay(30); } }
```

Handwritten code pic:



4) ~~Aim~~ : Pgm title : Analog Output - LED fading.

Aim: To control brightness periodically.

Hardware Required • Arduino Board •) LED

Circuit Diagram :

Code :

```
int ledPin = 9;
        Void setup ()     { }
        Void loop ()
        { for (int fade Value = 0 ; fade Value <=255 ; fade += 5)
              {  analogwrite (led Pin, fadeValue );
                  delay (30);
              for (int fade Value = 255 ; fadeValue >=0 ; fad +=5)
                  { analogWrite ( ledPin , fadeValue );
                     delay (30);
                  }
        }
```

Observation :- , LED fades & brightness periodically.

Observationa:LED fades and glows periodically , output is visualized using arduino uno.

Program no: **05** Program Title: **Nightlight Simulation** Date:07/12/2023

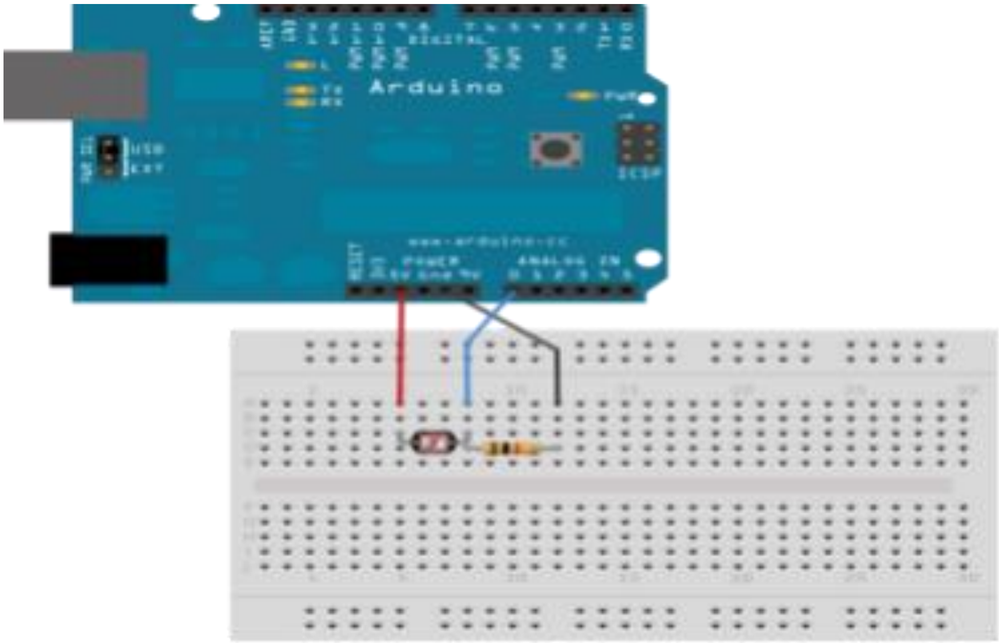
Aim: Simulating a night light using LDR

Hardware/components Required
Arduino Uno board - 1
USB Cable - 1
LED - 1
LDR-1
10K resistor-1
Jumper wires

Circuit Diagram / Pin 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

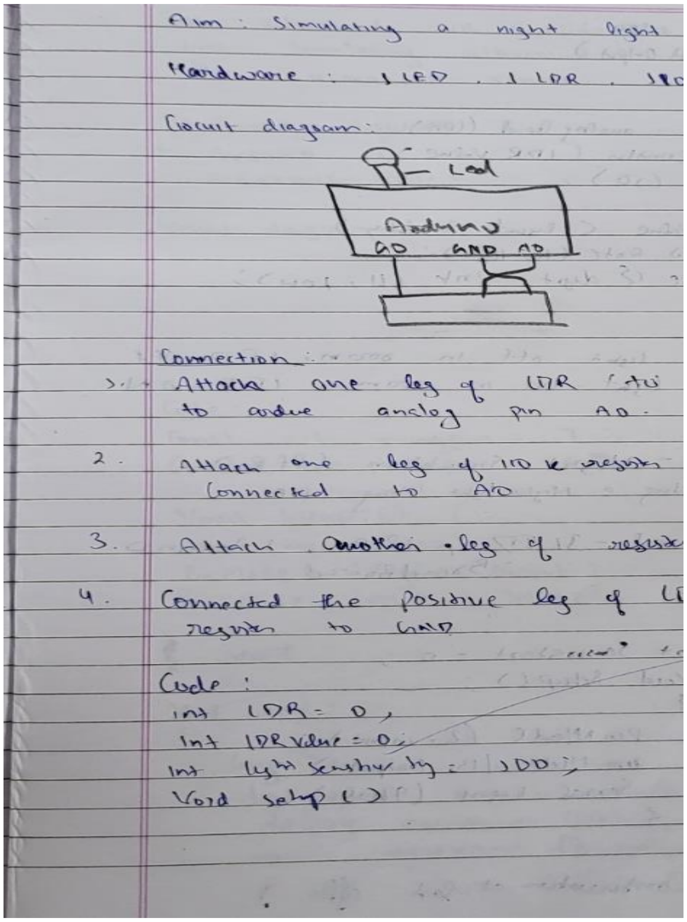


Code:

```
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 11
```

```
        void setup()
         {
         Serial.begin(9600); //start the serial monitor with 9600 buad
         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 : Simulating a night light

Hardware : 1 LED . 1 LDR . 1 10

Circuit diagram:

Led
Arduino
A0
GND
A0

Connection :

1. Attach one leg of LDR to
   to ardue analog pin A0.

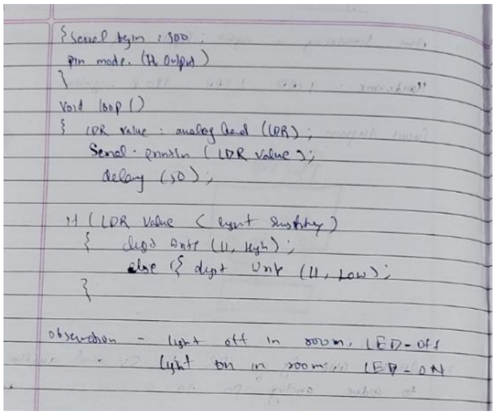
2. Attach one leg of 10 k resistor
   Connected to A0

3. Attach another leg of resistor

4. Connected the positive leg of LI
   resistor to GND

Code :

```
int LDR = 0,
Int LDR Value = 0;
Int light sensitivity = 500;
Void setup ()
```



```
{Serial begin : 9600 ;
 pin mode. (11 Output)
}
Void loop ()
{  LDR value : analog Read (LDR) ;
   Serial . println ( LDR value ) ;
    delay (50) ;

 If ( LDR Value < light Sensitivity )
   {   digitl Write (11, High) ;
        else ({ digit Write (11, Low) ;
   }
```

observation - light off in room, LED - off
 · light on in room, LED - ON

Handwritten code pic:

Observation: While lights are switched off in the room, LED should switch ON, when lights are switched on in the room, LED should switch off immediately.

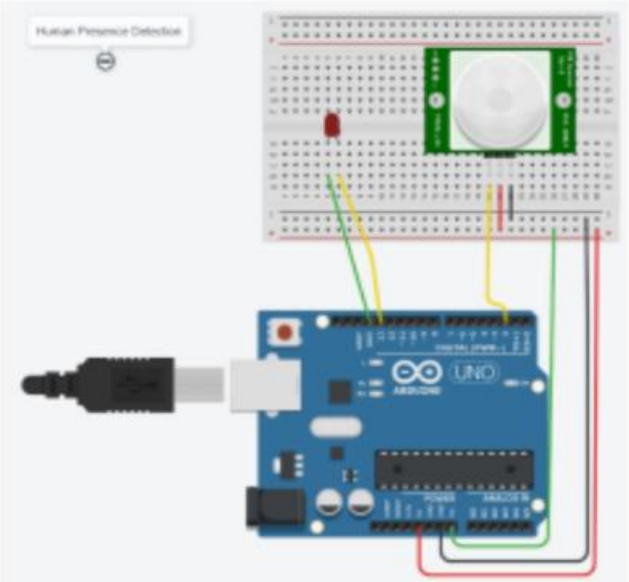
Program no: **06** Program Title: **Nightlight Simulation** Date:07/12/2023

Aim: Simulating a night light using PIR

Hardware/components Required

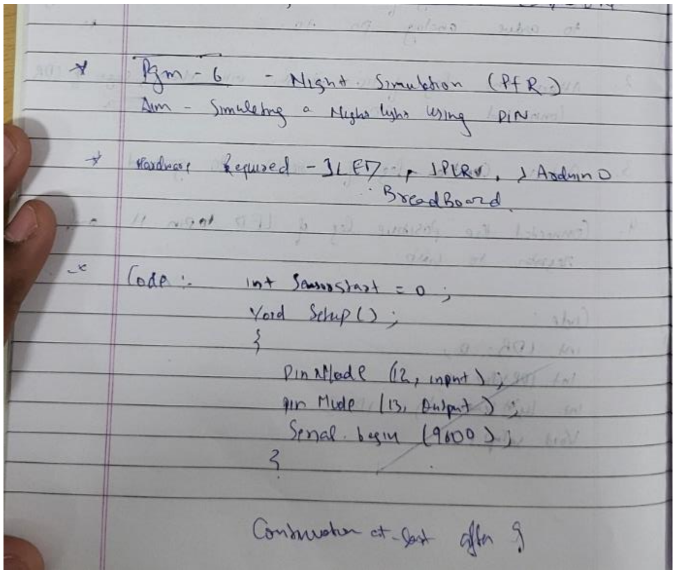
Arduino Uno board - 1
USB Cable - 1
LED - 1
PIR sensor-1
Jumper wires

Circuit Diagram / Pin 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);
}
```



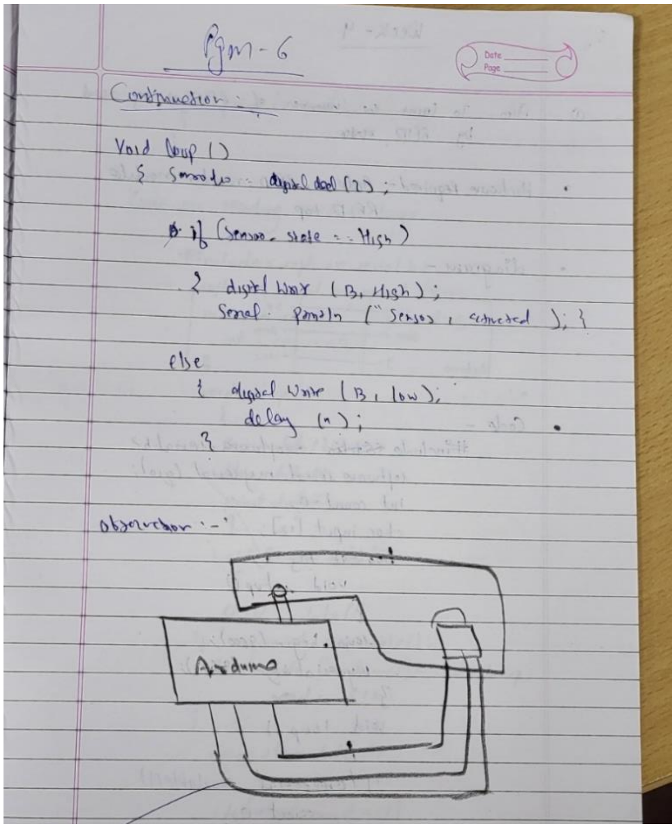
Pgm - 6 - Night Simulation (PIR)

Aim - Simulating a Night light using PIN

Hardware Required - 1 LED , 1 PIR , 1 Arduino
Breadboard.

```
Code :-   int Sensorstate = 0 ;
          Void Setup () ;
          {
             PinMode (12, input ) ;
             pin Mode (13, Output ) ;
             Serial . begin (9600 ) ;
          }
```

Continuation at last after 9



Pgm - 6

Continuation :-

```
Void loop ()
 { Sensorstate = digitalRead (12) ;

     ~~B.~~ if (Sensor_state == High )

      { digitalWrite (13, High ) ;
        Serial . println ("Sensor activated ); }

     else
      { digitalWrite (13, low );
          delay (n ) ;
      }
```

Observation :-

Handwritten code pic:

Observation: While lights are switched off in the room, LED should switch ON, when lights are switched on in the room, LED switches off.

Program no: **07** Program Title: **Ultrasound sensing** Date:07/12/2023

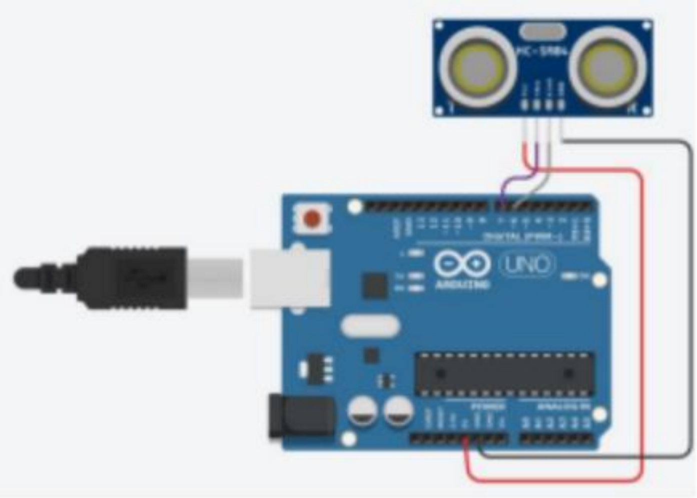
Aim: Simulating ultrasound with Arduino UNO and Ultrasonic sensor

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
Ultrasonic sensor-1
Jumper wires

Circuit Diagram / Pin connection

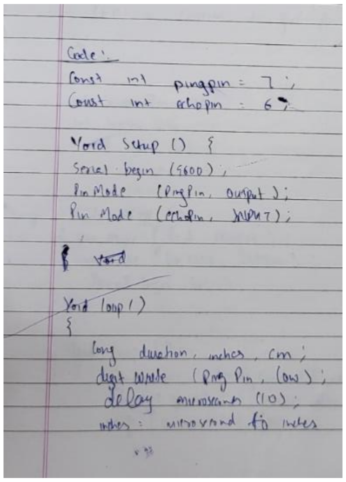
VCC-5V , GND-GND , pingpin-7 , echopin - 6



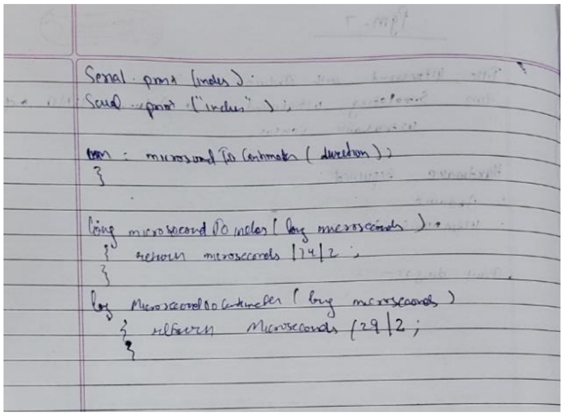
Code:

```
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;

 }
```

Handwritten code pic:



```
Code:
Const  int  pingpin = 7;
Const  int  echopin = 6;

Void Setup () {
Serial.begin (9600);
PinMode  (PingPin, Output);
Pin Mode  (echoPin, INPUT);

}  ~~Void~~

Void loop ()
{
   long  duration, inches, cm;
   digit Write  (Ping Pin, low);
   delay  microsecond (10);
   inches =  microsecond to inches
```



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

cm = microsecond To Centimeter ( duration );
}

long  microsecond To inches ( long  microseconds ).
 {  return  microseconds /74/2;
 }
long  Microsecond To Centimeter ( long  microseconds )
   {  return   Microseconds /29/2;
   }
```

Observation: Based on vibrations of sound,distance will be measured.

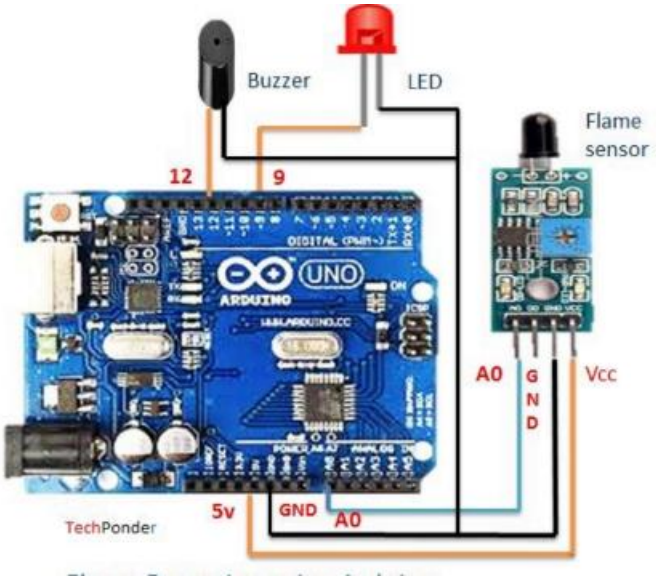
Program no: **08** Program Title: **Fire Alert** Date:07/12/2023

Aim: Fire alarm simulation

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

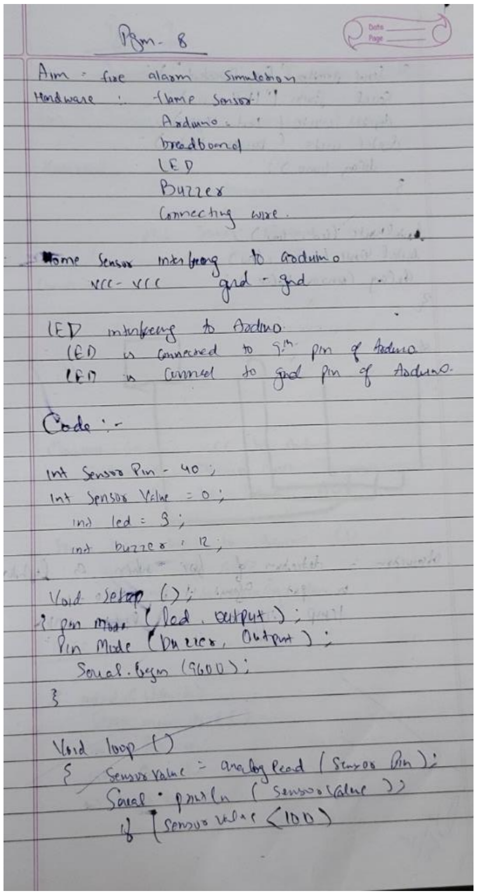
Buzzer interfacing to Arduino

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

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

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);
}
```



Pgm - 8

Aim : fire alarm Simulation

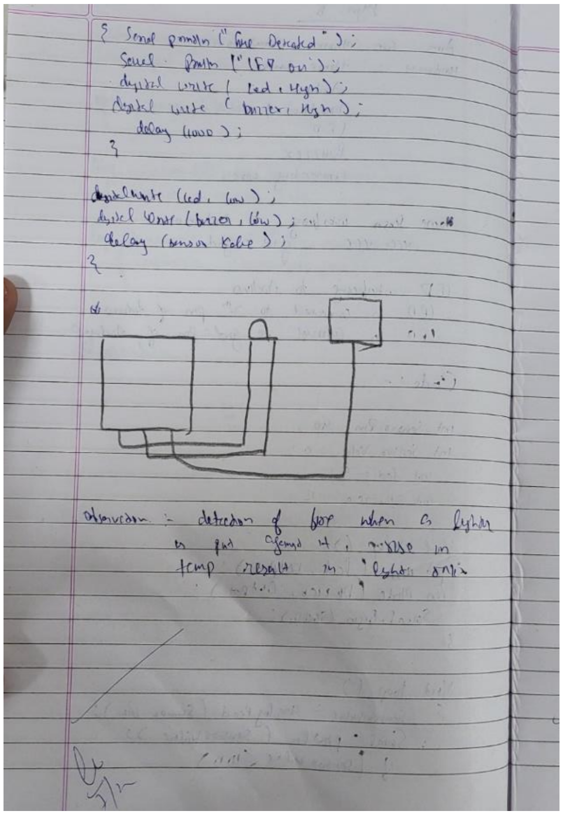
Hardware : flame Sensor
Arduino
breadboard
LED
Buzzer
Connecting wire.

flame Sensor interfacing to Arduino
vcc - vcc   gnd - gnd

LED interfacing to Arduino
LED is connected to 9th pin of Arduino
LED is connected to gnd pin of Arduino.

Code :-

```
int Sensor Pin - 40 ;
int Sensor Value = 0 ;
  int led = 9 ;
  int buzzer : 12 ;

Void Setup () ;
{ pin Mode (led, output) ;
 Pin Mode (buzzer, Output) ;
   Serial. begin (9600) ;
}

Void loop ()
 {  Sensor Value = analog Read ( Sensor Pin ) ;
    Serial . println ( Sensor Value ))
      if [ Sensor Value < 100 )
```



```
{ Serial println ("fire Detected") ;
  Serial . println ("LED on") ;
  digital write ( led , High ) ;
 digital write ( buzzer, High ) ;
   delay (1000 ) ;
 }

digital write (led, low ) ;
digital write (buzzer , low ) ;
 delay (sensor Value ) ;
}
```

Observation :- detection of fire when a lighter is put against it , raise in temp result in lights on

Observation: When flame is detected,LED and buzzer turns ON.

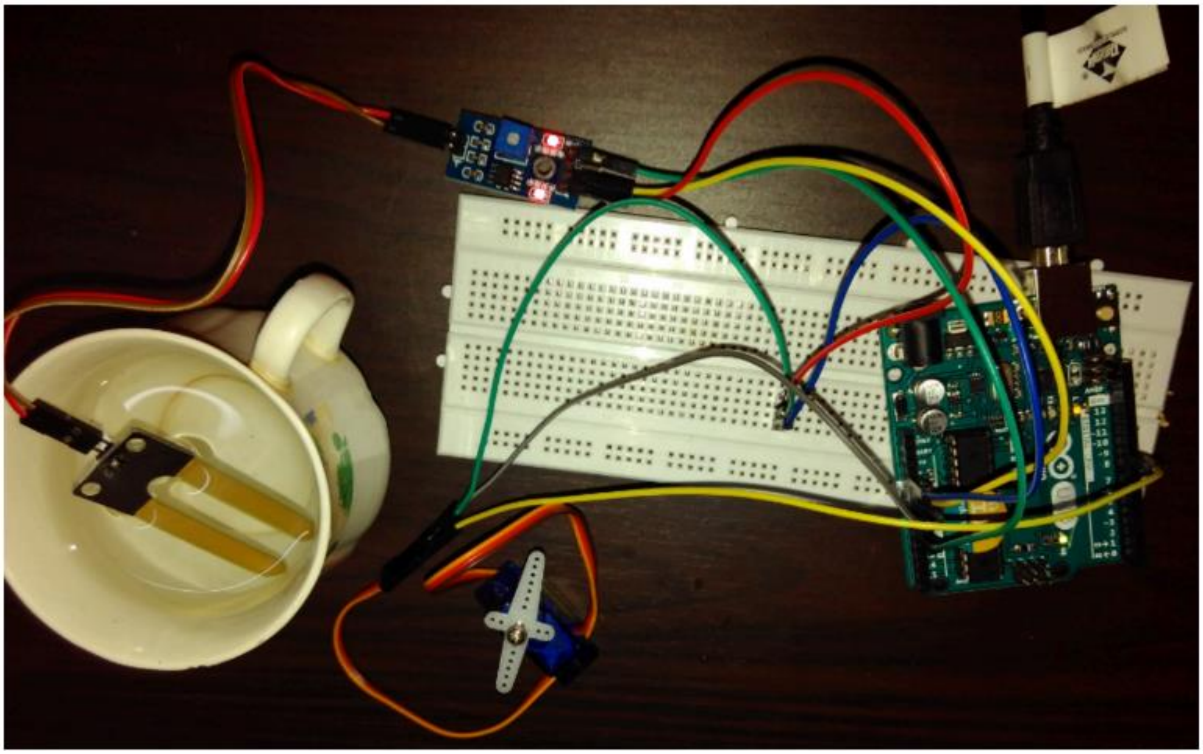
Program no: **09** Program Title: **Automatic Irrigation** Date:07/12/2023

Aim: Sensing the soil moisture and sprinkling the Water simulation

Hardware Required

Arduino
Moisture Sensor
Breadboard
Min servo motor

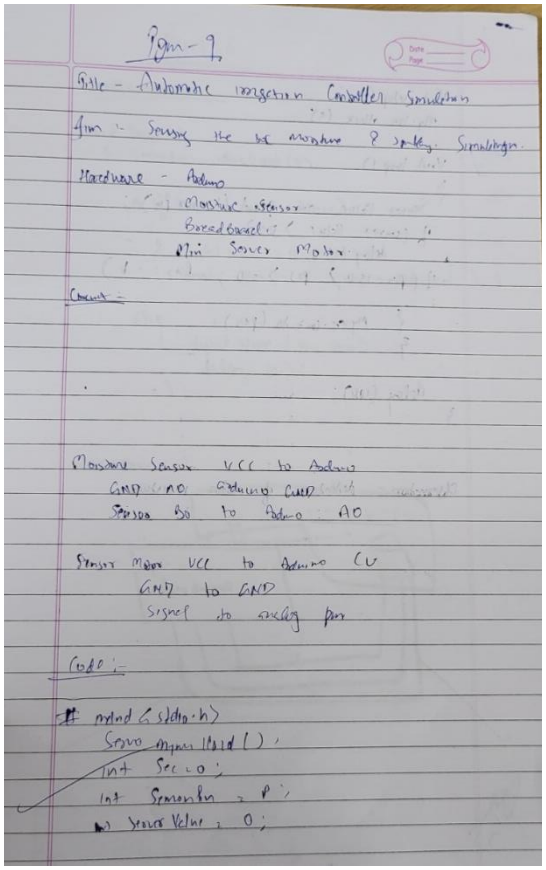
Circuit diagram



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); // tell servo to go to position in variable '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);
}
```



Pgm - 9

Title - Automatic irrigation Controller Simulation

Aim :- Sensing the so moisture & spiking Simulation.

Hardware - Arduino
Moisture Sensor
Bread board
Mini Servo Motor

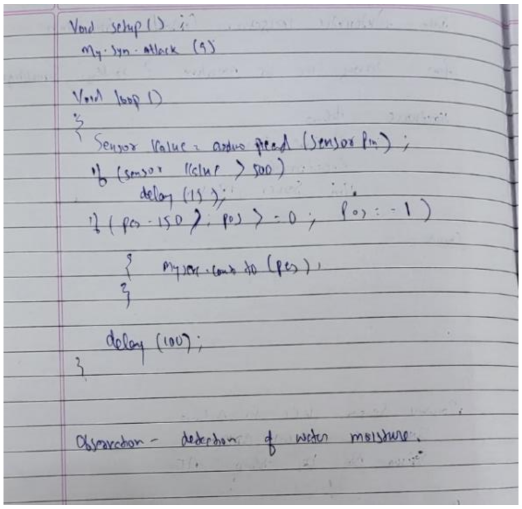
Circuit -

Moisture Sensor VCC to Arduino
GND AO Arduino GND
Sensor Bo to Arduino AO

Sensor Motor VCC to Arduino CV
GND to GND
Signal to analog pin

Code :-

```
# include <stdio.h>
  Servo myservo();
  int Sec = 0;
  int SensorPin = P;
  int Sensor Value = 0;
```



```
Void setup () {
  My. syn. attack (9)

Void loop ()
{
  Sensor Value = ardue read (Sensor Pin);
  if (sensor value > 500)
      delay (15);
  if ( pos = 150 ); pos > = 0 ; pos : -1 )
      {   myser. count to (pes);
      }
    delay (100);
}
```

Observation - detection of water moisture.

Observation: Soil moisture sensor continuously detects the soil moisture and servo motor would turn ON when there is a low moisture level.

Program no: **10** Program Title: **READING RFID TAG** Date:21/12/2023

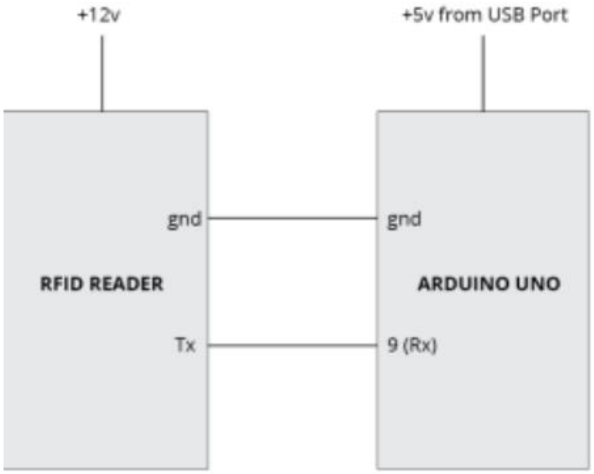
Aim: To read the code present on RFID tag and print it in serial monitor.

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
RFID tag
Jumper wires

Circuit Diagram / Pin connection

5V-Arduino 5V GND-Arduino GND Tx-pin 9

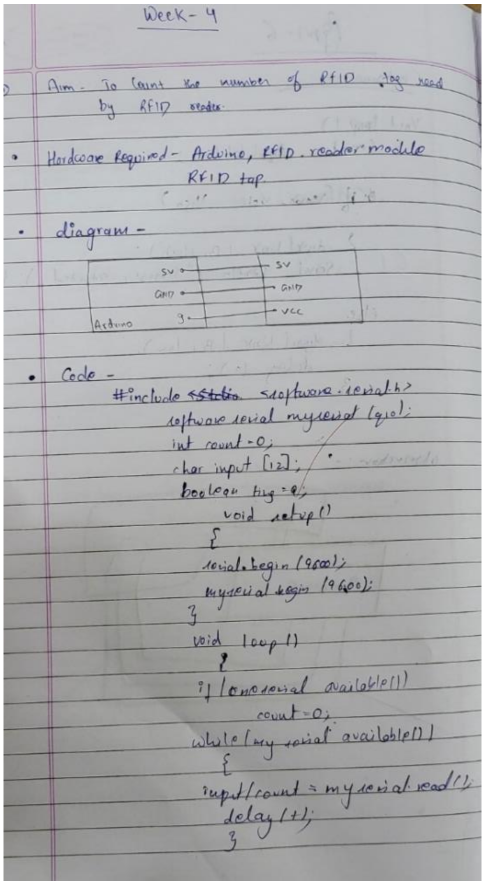


Interfacing RFID Reader to Arduino

C ode:

```
#include<SoftwareSerial.h>


 SoftwareSerial mySerial(9, 10);


int count = 0; // count = 0
 char input[12]; // character array of size 12




boolean flag = 0; // flag =0
void setup()
 {
   Serial.begin(9600); // begin serial port with baud rate 9600bps mySerial.begin(9600);
 }
void loop()
 {
   if(mySerial.available())
    {
      count = 0;
      while(mySerial.available() && count < 12) // Read 12 characters and store them in input
array
       {
         input[count] =mySerial.read();
          count++;
          delay(5);
       }
      Serial.print(input); // Print RFID tag number

          }
 }
```

Handwritten code pic:



Week- 4

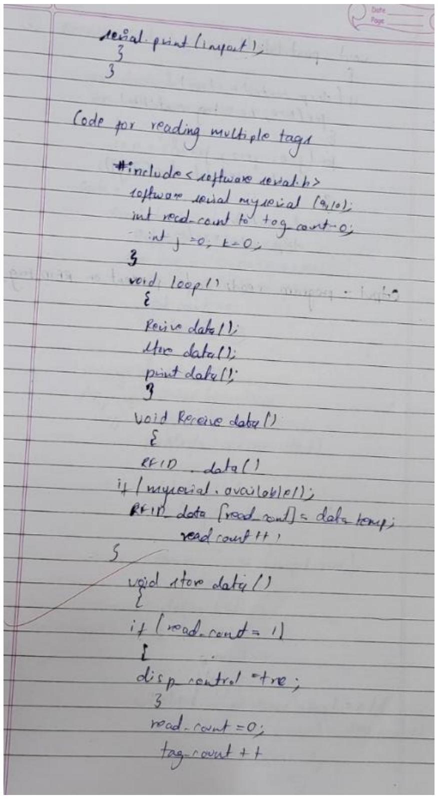
Aim - To Count the number of RFID tag read by RFID reader.

- Hardware Required - Arduino, RFID reader module RFID tap.
- diagram -

5V
GND
9
Arduino
5V
GND
VCC

- Code -

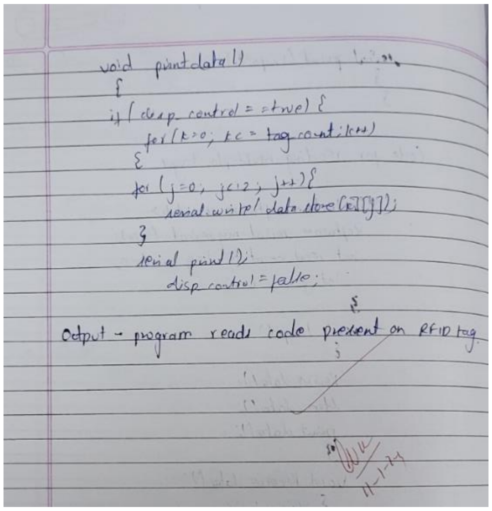
```
#include ~~<Stdio~~ <software.serial.h>
        software serial myserial (9,10);
        int count=0;
        char input [12];
        boolean flag=0;
            void setup()
          {
          serial.begin (9600);
          myserial begin (9600);
        }
        void loop()
          {
         if (mereserial available())
              count=0;
        while(my serial available())
          {
         input/count = my serial read();
          delay(+);
          }
```



```
serial.print(input);
  }
  }
```

Code for reading multiple tags

```
#include<software serial.h>
 software serial myserial (9,10);
 int read-count to tag count-0;
   int j =0, k=0;
 }
 void loop()
   {
   Recive data();
   store data();
   print data();
   }
  void Receive data()
   {
   RFID . data()
 if (myserial.available());
  RFID_data [read_count] = data temp;
        read count ++,
 }
  void store data()
  {
  if (read-count= 1)
   {
   disp control =tre;
    }
   read-count =0;
    tag-count ++
```



```
void  printdata()
 {
 if ( disp_control = =true) {
    for(k=0; k<= tag count;k++)
   {
   for (j=0; j<12; j++){
       serial.write( data store [k][j]);
   }
   serial print();
       disp control = false;
                    }
```

Output - program reads code present on RFID tag.

11-1-24

Observation: The output consists of 12 character ASCII data, where first 10 bits will be the tag number and last 2 bits will be the XOR result of the tag number which can be used for error correction.

Program no: **11** Program Title: **ACCESS CONTROL via RFID TAG** Date:21/12/2023

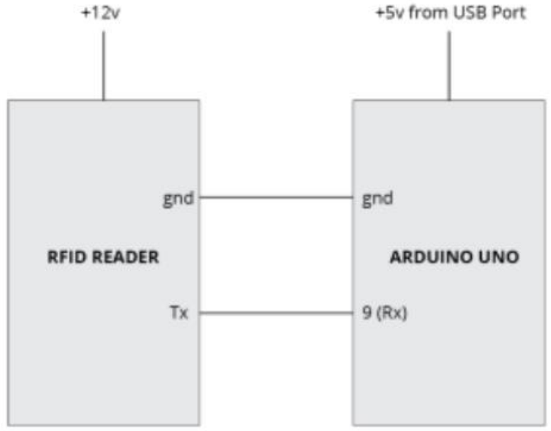
Aim: To read the code present on RFID tag tapped. If the code matches with the previously known tag(configured in the code), it will grant access(here LED will glow), otherwise access will be denied.

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
RFID tag
Jumper wires

Circuit Diagram / Pin connection

5V-Arduino 5V GND-Arduino GND Tx-pin 9



Interfacing RFID Reader to Arduino

Code:

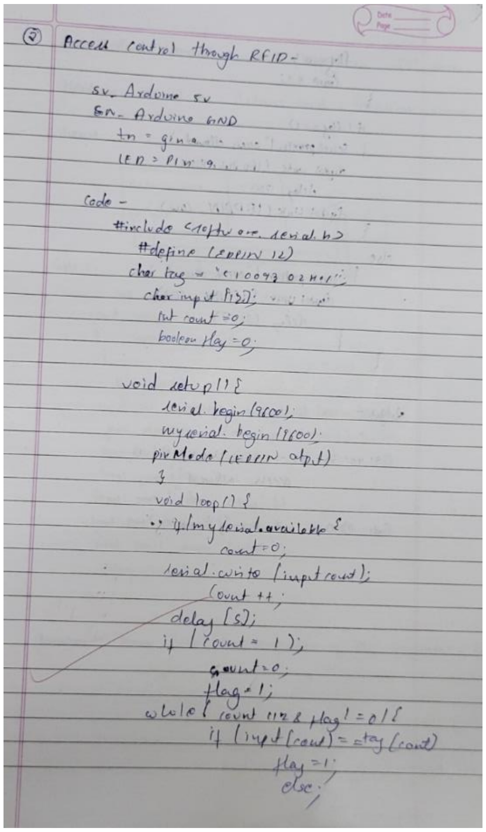
```
#include<SoftwareSerial.h>


 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); // Initialise Serial Communication with the Serial Monitor
             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; // Reset the counter to zero
                        /* Keep reading Byte by Byte from the Buffer till the RFID Reader Buffer is
                empty
                          or till 12 Bytes (the ID size of our Tag) is read */
                        while(mySerial.available() && count < 12)
                         {
                                input[count] = mySerial.read();
// Read 1 Byte of data and store it in the input[] variable
                                      Serial.write(input[count]);
                                count++; // increment counter
                                delay(5);
                         }
                        /* When the counter reaches 12 (the size of the ID) we stop and compare each
        value
                           of the input[] to the corresponding stored value */
                        if(count == 12) //
                         {
                                count =0; // reset counter varibale to 0
                                flag = 1;
/* Iterate through each value and compare till either the 12 values are
all matching or till the first mistmatch occurs */
                                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;
/* if the ID values don't match, set flag variable to 0 and
                               stop comparing by exiting the while loop */
                                        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);
                                        }
/* Fill the input variable array with a fixed value 'F' to overwrite
                        all values getting it empty for the next read cycle */
                        for(count=0; count<12; count++)
                         {
```

```
                        input[count]= 'F';
                    }
                    count = 0; // Reset counter variable
            }
    }
```

Handwritten code pic:

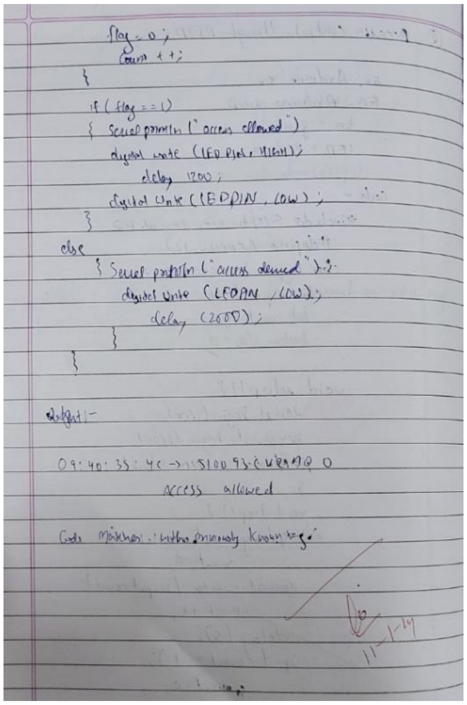


② Access control through RFID-

5v- Arduino 5v
Gn- Arduino GND
tn = pin 9
LED = Pin 9

Code -

```
    #include <softwore. serial. h>
        #define (LEDPIN 12)
      char tag = "c100?3 02H01";
        char input [12];
          int count =0;
          boolean flag =0;

      void setup() {
          serial. begin (9600);
          myserial. begin (9600).
        pinMode (LEDPIN output)
        }
        void loop() {
        if (myserial.available {
                count=0;
          serial.write (input count);
                count ++;
            delay [5];
          if (count = 1);
                count=0;
                flag=1;
        while( count <12 & flag! =0) {
              if (input(count)= =tag (count)
                        flag =1;
                        else;
```



```
        flag = 0 ;
          Count ++;
      }
      if ( flag = = 1)
      { Serial println ("access allowed")
        digital write (LED PIN, HIGH);
            delay 1200 ;
        digital Write (LEDPIN , LOW) ;
      }
  else
      { Serial println ("access denied");
          digital write (LEDPIN , LOW);
              delay (2000) ;
          }
    }
```

Output :-

```
09: 40: 35 : 4C -> : 510093 CVR9AQ 0
              Access allowed
```

Code Matches : with the previously Known tag.

11-1-14

Observation: 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.

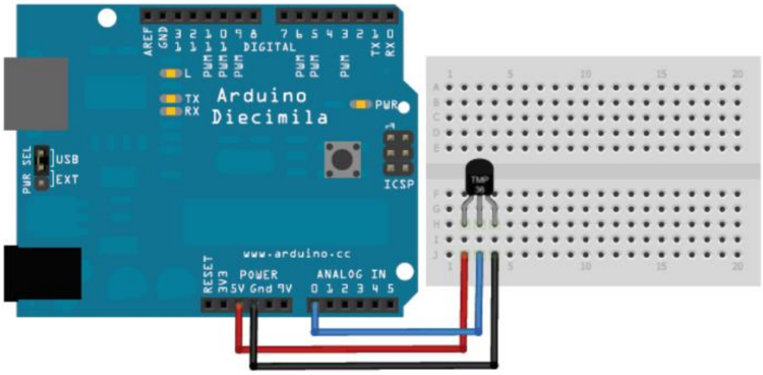
Program no: **12** Program Title: **TEMPERATURE SENSING** Date:21/12/2023

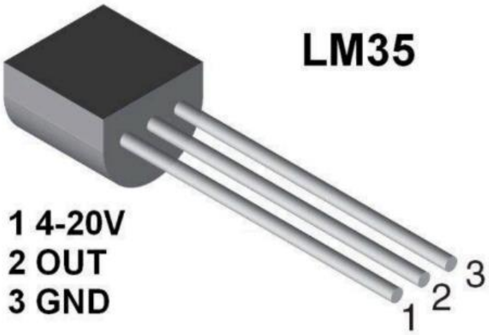
Aim: To monitor the temperature using LM35.

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
Temperature sensor LM35
Jumper wires

Circuit Diagram / Pin connection

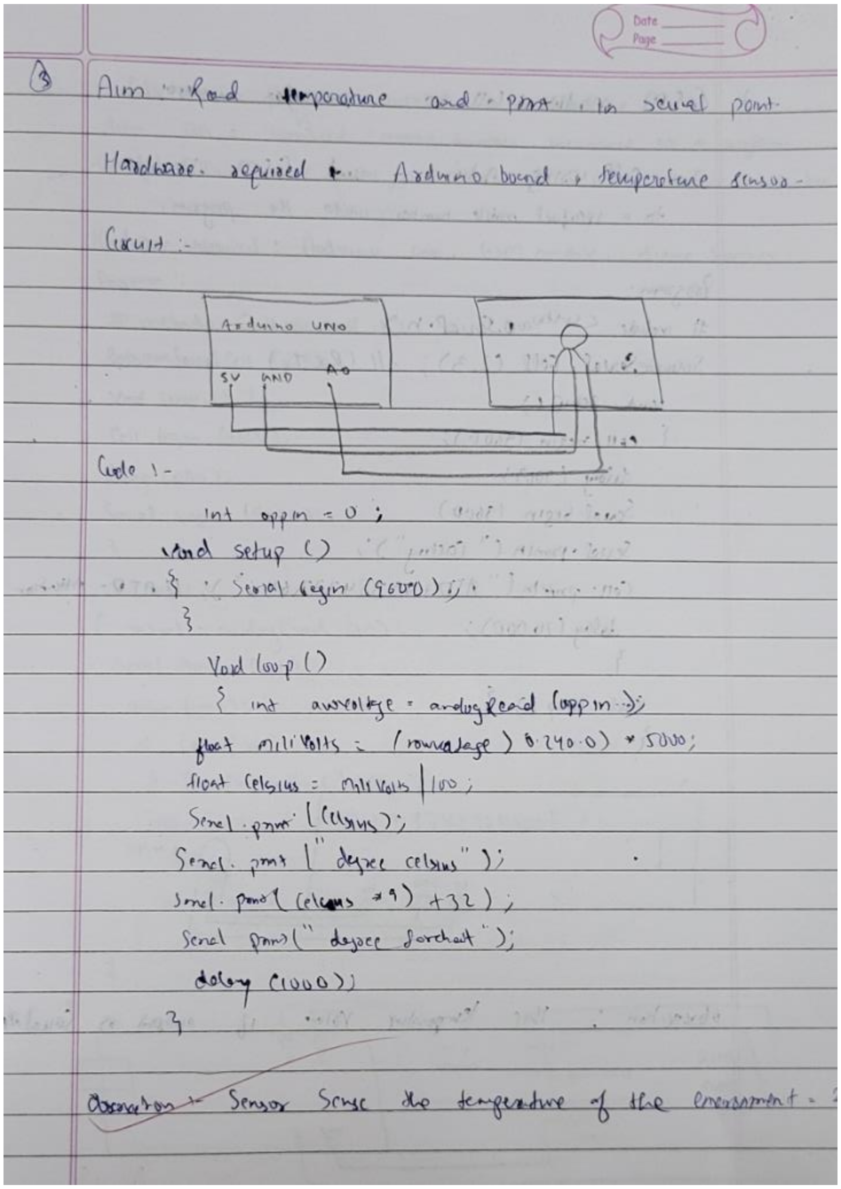




Code:

```
int sensorPin = 0; //the analog pin the TMP36's Vout (sense) pin is connected to
                        //the resolution is 10 mV / degree centigrade with a
                         //500 mV offset to allow for negative temperatures


/*
 * setup() - this function runs once when you turn your Arduino on
 * We initialize the serial connection with the computer
 */
void setup()
```

```
{
  Serial.begin(9600); //Start the serial connection with the computer
                       //to view the result open the serial monitor
}

void loop() // run over and over again
{
 //getting the voltage reading from the temperature sensor
 int reading = analogRead(sensorPin);

 // converting that reading to voltage, for 3.3v arduino use 3.3
 float voltage = reading * 5.0 / 1024;

 // print out the voltage
 Serial.print(voltage); Serial.println(" volts");

 // now print out the temperature
 float temperatureC = (voltage - 0.5) * 100 ; //converting from 10 mv per degree wit 500 mV offset
                                               //to degrees ((volatge - 500mV) times 100)


 Serial.print(temperatureC); Serial.println(" degress C");

 // now convert to Fahrenheight
 float temperatureF = (temperatureC * 9 / 5) + 32;
 Serial.print(temperatureF); Serial.println(" degress F");

 delay(1000); //waiting a second
}
```



③ Aim : Rad temperature and print in serial point.

Hardware required :- Arduino board , temperature sensor -

Circuit :-

Arduino UNO
5V
GND
A0

Code :-

```
        int oppin = 0 ;
    void setup ()
     { Serial begin (9600) ;
      }

        Void loop ()
        { int awvoltage = andogRead (oppin ));
      float miliVolts = (rowvalage ) 0.240.0) * 5000;
      float Celsius = miliVolts | 100 ;
      Serel . print [ (celsius ) ;
     Serel . print [ " degree celsius " ) ;
     Serel . print ( (celcius *9) +32 ) ;
     Serel print ( " degree farchait " ) ;
      delay (1000 ))
     }
```

Observation :- Sensor Sense the temperature of the environment .

Handwritten code pic:

Observation: Sensor senses the temperature of the surroundings as 21C

Program no: **13** Program Title: **GSM CALLING** Date:28/12/2023

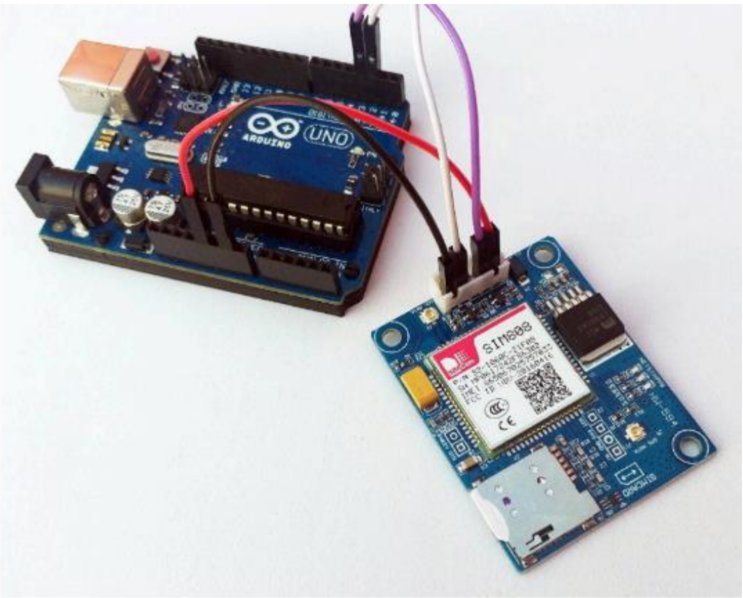
Aim: Call using Arduino and GSM Module – to a specified mobile number inside the program.

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
GSM module
SIM slot
Jumper wires

Circuit Diagram / Pin connection:

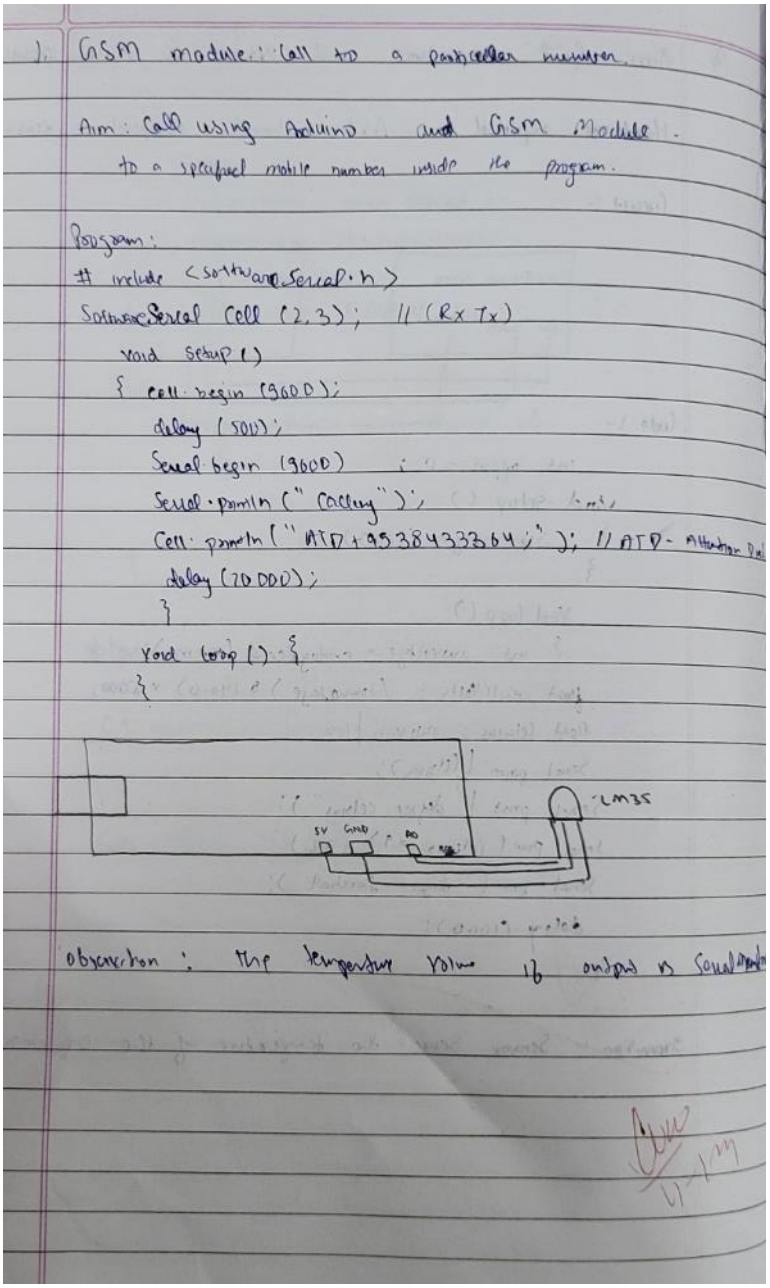
GSM Tx –>Arduino Rx (Here pin 2)
GSM Rx –>ArduinoTx. (Here pin 3)
Make the ground common between Arduino and GSM modem.



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;"); // ATD – Attention Dial
 delay(20000);
 }
void loop() {
```

}



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 () {
   }
```

5V
GND
A0
LM35

Observation: The temperature value is output in Serial Monitor

Handwritten code pic:

Observation: Calling to GSM module , you’ll get beep sound .

Program no: **14** Program Title: **GSM CALLING FIRE ALERT** Date:11/01/2024

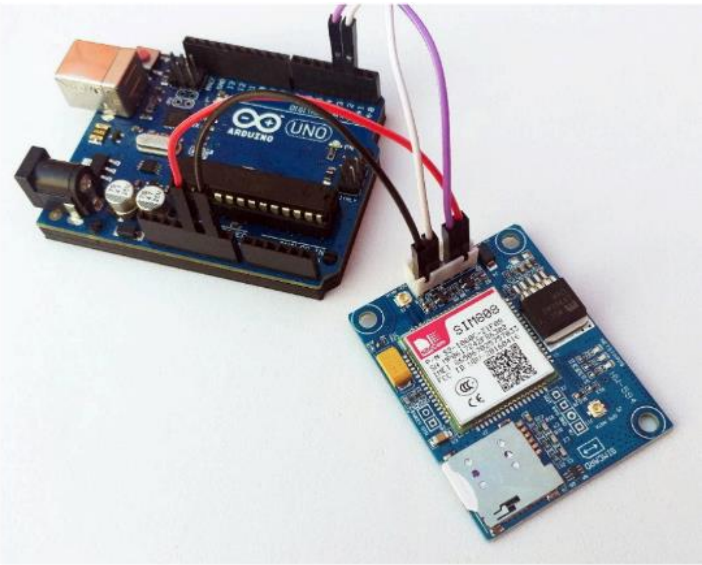
Aim: Call a specified mobile number mentioned in the program using Arduino and GSM Module when a flame sensor detects “fire”.

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
GSM module
SIM slot
Flame sensor
Jumper wires

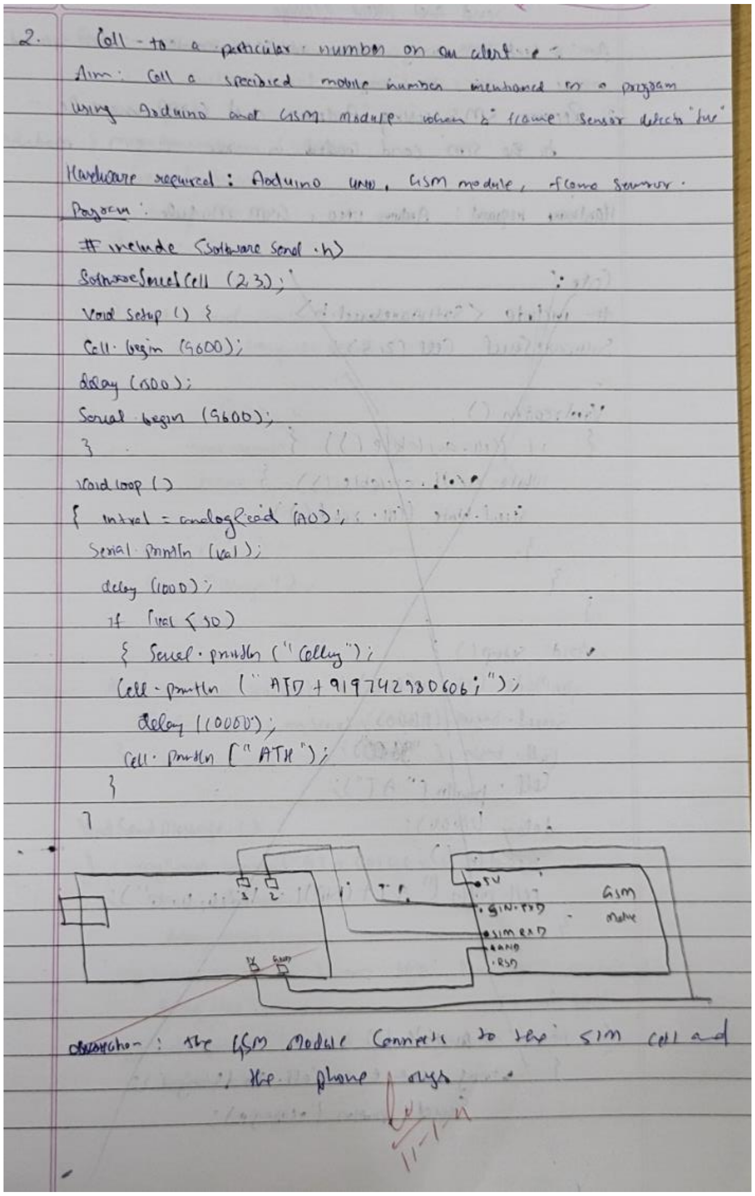
Circuit Diagram / Pin connection:

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



Code:

```
#include <SoftwareSerial.h>
 SoftwareSerialcell(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
   }
}
```



2. Call to a particular number on an alert :

Aim: Call a specified mobile number mentioned in a program using Arduino and GSM module when a flame sensor detects "fire"

Hardware required : Arduino uno, GSM module, flame sensor.

Program:

```
#include <Software Serial.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+919742980606;");
     delay (10000);
    Cell.println ("ATH");
   }
 }
```

Observation : The GSM Module Connects to the SIM call and the phone rings.

11-1-

Handwritten code pic:

Observation: When there is a flame,a particular specified number will get a call as an alert.

Program no: **15** Program Title: **SMS SERVICE USING GSM** Date:11/01/2024

Aim:

1) Send SMS using Arduino and GSM Module – to a specified mobile number inside the program

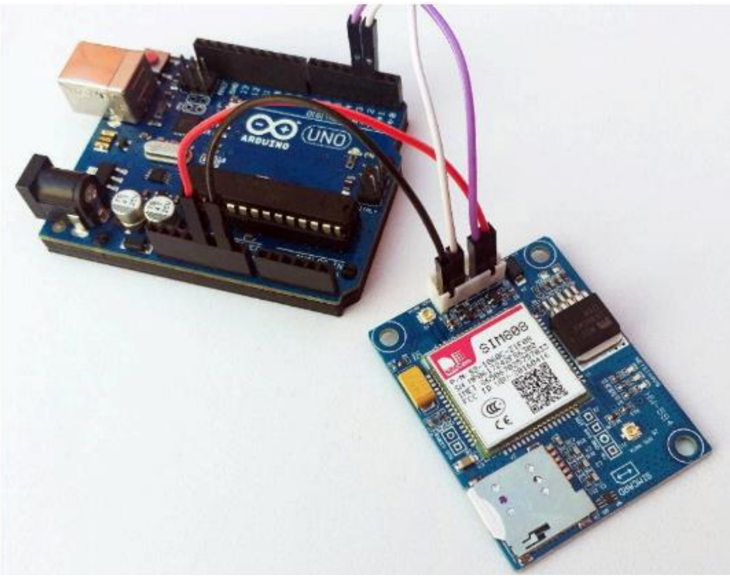
2) Receive SMS using Arduino and GSM Module – to the SIM card loaded in the GSM Module.

Hardware/components Required

Arduino Uno board - 1
USB Cable - 1
GSM module
SIM slot
Jumper wires

Circuit Diagram / Pin connection:

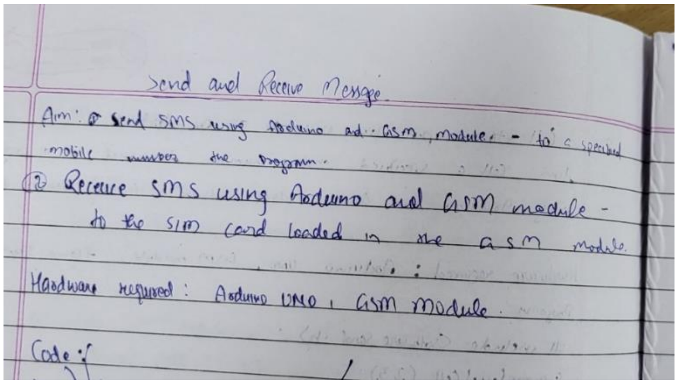
GSM Tx –>Arduino Rx (Here pin 2)
GSM Rx –>ArduinoTx. (Here pin 3)
Make the ground common between Arduino and GSM modem.



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());
 }

voidSendMessage()
 {
 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 withyour 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);// ASCII code of CTRL+Z , to terminate the
 message delay(1000);



 }

voidRecieveMessage()
 {
 mySerial.println("AT+CNMI=2,2,0,0,0"); // AT+CNMI, New Message
Indications // AT Command to recieve a live SMS
 delay(1000);
  }
```



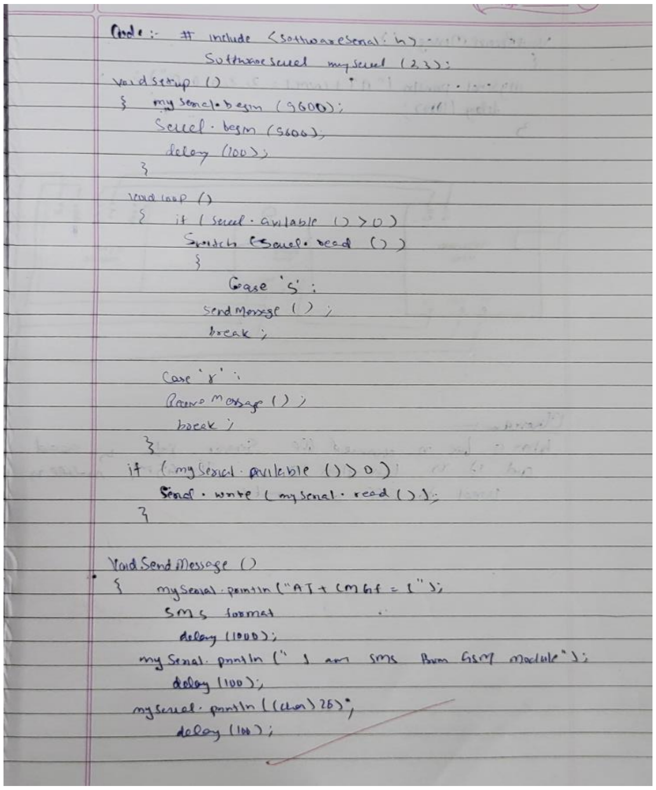
Send and Receive Message.

Aim: ① Send SMS using Arduino and GSM module - to a specified mobile number the program.

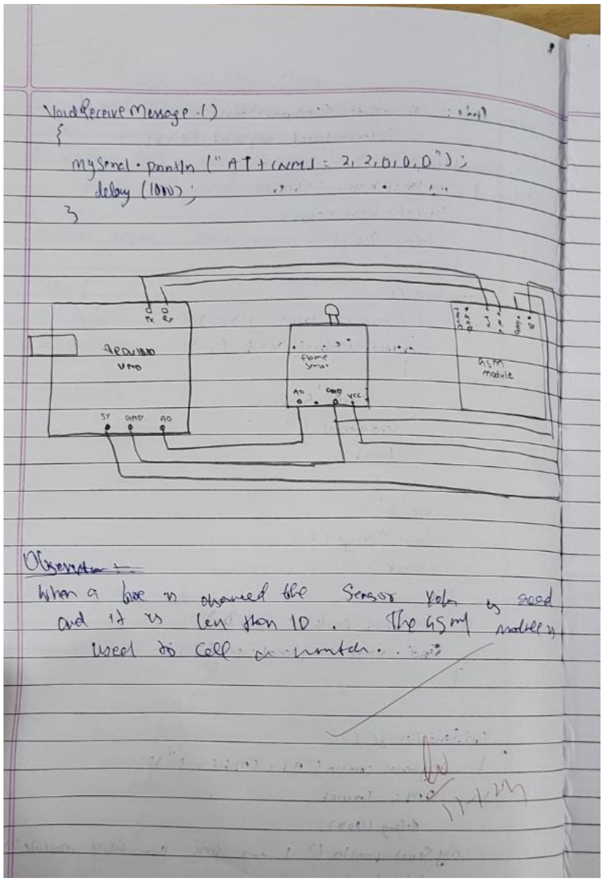
② Receive SMS using Arduino and GSM module - to the SIM card loaded in the GSM module.

Hardware required: Arduino UNO, GSM Module.

Code:-



```
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 's' :
          SendMessage () ;
           break ;

        Case 'r' :
        ReceiveMessage () ;
         break ;
    }
  if (mySerial.available () > 0)
     Serial.write (mySerial.read ());
   }

Void SendMessage ()
 {    mySerial.println ("AT+CMGF = 1");
      SMS format
        delay (1000);
    mySerial.println ("I am SMS from GSM module");
        delay (100);
   mySerial.println ((char)26);
        delay (100);
```



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

Observation :-

When a fire is observed the Sensor value is read and if it is less than 10. The GSM module is used to call a number.

Handwritten code pic:

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

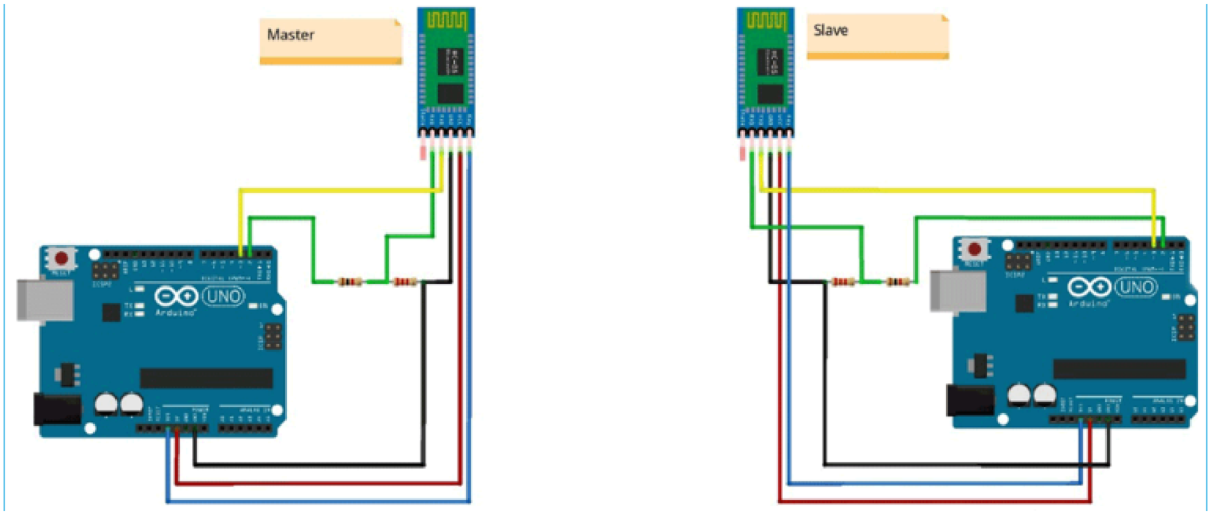
Program no: **16** Program Title: **BLUETOOTH MASTER SLAVE** Date:18/01/2024

Aim: To control the LED in the master device by client device.

Hardware/components Required

Arduino Uno board - 2
USB Cable - 1
Jumper wires
LED-1
HC-05 bluetooth module-2

Circuit Diagram / Pin connection:



Slave Mode:

The HC-05 bluetooth module can also act as a slave. There are fewer commands to set this up:

AT+ORGL Reset to defaults
AT+RMAAD Clear any paired devices
AT+ROLE=0 Set mode to SLAVE
AT+ADDR Display SLAVE address //+ADDR:98d3:33:807822
Master Mode:

To configure the module as Bluetooth Master and to pair with another bluetooth module follow these steps. First we need to put the module into command mode Enter these commands in order:

AT+RMAAD Clear any paired devices
AT+ADCN
AT+ROLE=1 Set mode to Master
AT+CMODE=0 Allow master to ONLY connect to bound address (slave). This allows the master to automatically connect to the slave when switched on AT+PSWD=1234 Set

PIN. Should be same as slave device

AT+BIND=<address> Set bind address to the slave address
AT+LINK=<address> Connect to slave.
AT+INIT

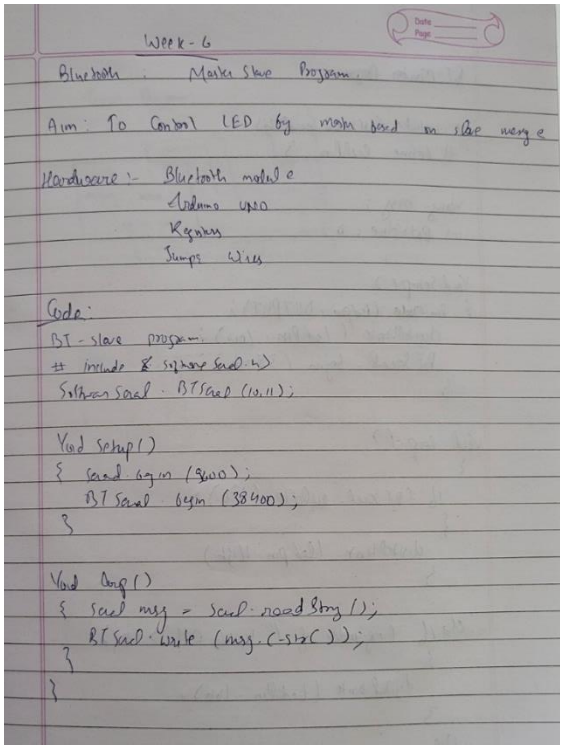
Note: If it shows any Error, then check if both the bluetooth modules are blinking in sync. If so then both the bluetooth modules are synchronized.

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 more }
void loop() {
  // Reading the button
 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 more }


void loop() {
 if(BTSerial.available() > 0){
 // Checks whether data is comming from the serial port //
    Reads the data from the serial port
     message = BTSerial.readString();
    // Controlling the LED
    if(message.indexOf("SWITCH ON")>=0)
    {
    digitalWrite(ledPin, HIGH); // LED ON
    }
   else if(message.indexOf("SWITCH OFF")>=0)
    {
     digitalWrite(ledPin, LOW); // LED OFF
    }
   else
    {
     Serial.println("Noting to do");
    }
  delay(100);

 }
 delay(10);
}
```



Week-6

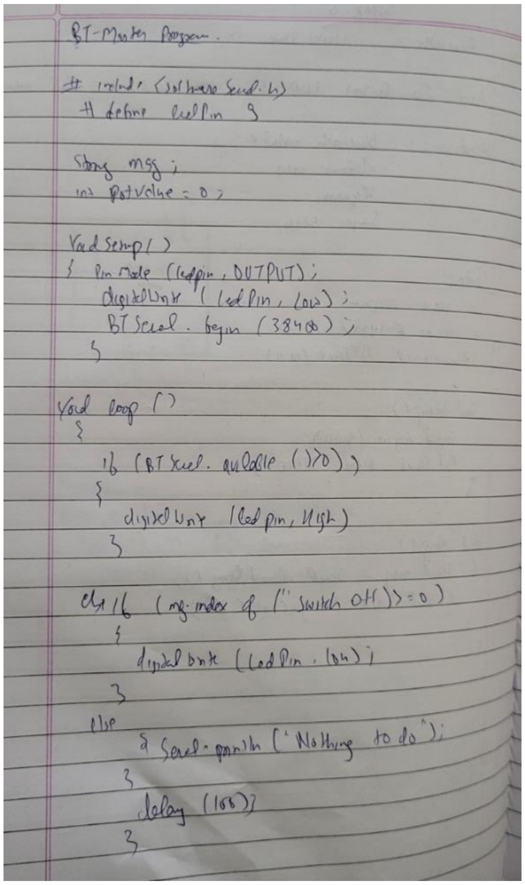
Bluetooth : Master Slave Program.

Aim: To Control LED by master based on slave message

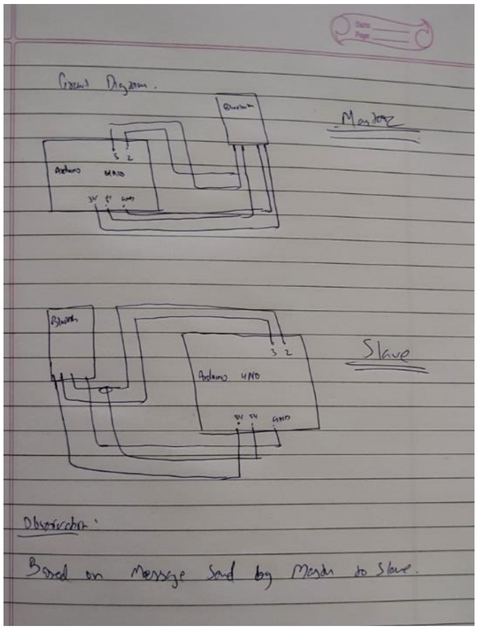
Hardware:- Bluetooth module
Arduino UNO
Registers
Jumper Wires

Code:

```
BT-slave program.
# include < Software Serial.h>
Software Serial BTSerial (10,11);

Void setup()
{ Serial.begin (9600);
  BT Serial begin (38400);
 }

Void loop()
 { Serial msg = Serial.readString();
   BT Serial.write (msg.c_str());
 }
}
```



```
BT-Master Program.

# include <Software Serial.h>
 # define ledPin 9

String msg;
int potValue = 0;

Void Setup()
{ Pin Mode (ledpin, OUTPUT);
   digitalWrite (ledPin, LOW);
   BT Serial. begin (38400);
  }

Void loop ()
 {
   if (BT Serial. available ()>0))
   {
     digitalWrite (ledpin, High)
    }

  else if (msg.index of ("switch off")>=0)
     {
       digitalWrite (ledPin, low);
     }
  else
       { Serial.println ("Nothing to do");
      }
      delay (100);
     }
```



Bluetooth
3 2
Arduino UNO
3V 5V GND
Bluetooth
3 2
Arduino UNO
3V 5V GND

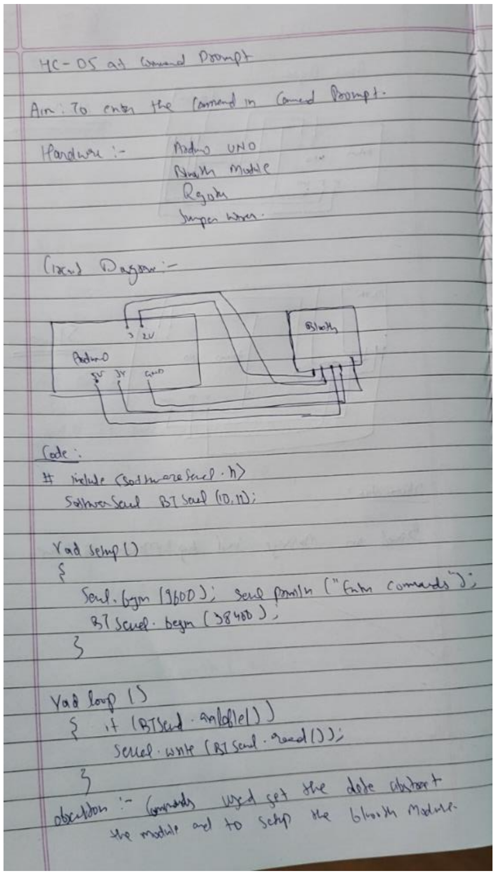
Circuit Diagram.

Master

Slave

Observation:

Based on Message Send by Master to Slave.



3 2V
Arduino
5V 3V GND
Bluetooth

HC-05 at Command Prompt

Aim: To enter the command in Command Prompt.

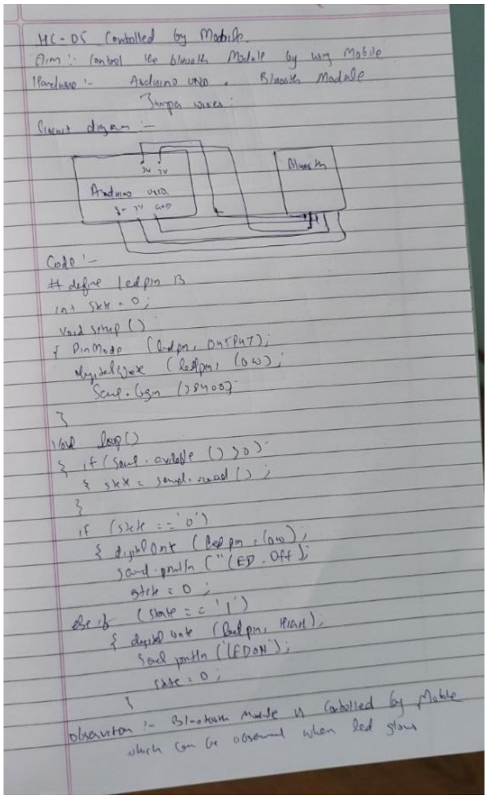
Hardware :- Arduino UNO
Bluetooth Module
Register
Jumper Wires.

Circuit Diagram :-

Code :

```
# include <SoftwareSerial.h>
  SoftwareSerial BTSerial (10,11);

 Void setup ()
 {
   Serial.begin (9600); Serial.println ("Enter commands");
   BTSerial.begin (38400);
  }

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

observation :- Commands used set the data abstract the module and to setup the bluetooth Module.



3V 2V
Arduino UNO
5V 3V GND
Bluetooth

HC-05 Controlled by Mobile.

Aim :- Control the bluetooth Module by using Mobile

Hardware :- Arduino UNO , Bluetooth Module
Jumper wires.

Circuit diagram :-

Code :-

```
# define ledpin 13
 int state = 0;
 Void setup ()
 { PinMode (ledpin, OUTPUT);
   digitalWrite (ledpin, LOW);
    Serial.begin (38400)
 }
Void loop ()
 { if (Serial.available () > 0)
    { state = serial.read () ;
   }
   if (state == '0')
     { digitalWrite (ledpin, LOW);
       Serial.println ("LED . Off");
        state = 0 ;
  else if (state == '1')
      { digitalWrite (ledpin, HIGH);
        Serial println ("LED ON");
         state = 0 ;
      }
```

observation :- Bluetooth Module is Controlled by Mobile which Can be observed when led glows

Handwritten code pic:

Observation: Whenever Client device sends the message “SWITCH ON”,LED turns ON and turns OFF if the message is “SWITCH OFF” otherwise it prints “Nothing to do” in the serial monitor.