Programme Name and Semester: MCA 4th Semester
Course Name (Course Code): Internet of Things and Data Analytics Lab (**MCA492A**)

Class: MCA 4H

Academic Session: 2024-25

Laboratory Manual

(Internet of Things and Data Analytics Lab & MCA492A)

Table of Contents

Name of the Faculty: Dr.B.Vasumathi
Designation and Department: Associate Professor (Computational Sciences)
Brainware University, Kolkata

Assignment No 1 - 15

1. **Aim/Purpose of the Assignments:**

Assignment 1: Aim of this assignment is to demonstrate basics of Arduino, LED, Breadboard, Resistor and create a program code to implement traffic signals.

Assignment 2: Aim of this assignment is to provide simple practice to Make an LED connected to a specific pin on the Arduino blink on and off. Program code showcase the use of Arduino, LED, resistor (220Ω), breadboard, jumper wires.

Assignment 3: Aim of this assignment is to Create a digital raindrop sensor using Arduino typically involves using a raindrop sensor module that includes a digital output. These modules are equipped with a sensor that detects raindrops and outputs a digital signal to the Arduino.

Assignment 4: Aim of this assignment is to focus on a moisture sensor is a device used to measure the moisture level in soil or other materials. It is commonly used in gardening and agriculture to monitor soil moisture for plants.

Assignment 5: Aim of this assignment is to monitor the ambient temperature of a room over a cycle of 24 hours and display the current temperature, the maximum temperature recorded, and the minimum temperature recorded within that 24-hour cycle on an LCD panel.

Assignment 6: Aim of this assignment is to illustrate touch sensor is commonly used to control devices where touch is detected. It defines the touch sensor's signal pin that we have connected with the Arduino's digital pin. It helps to know about setup (), Loop (), digitalRead().

Assignment 7: Aim of this assignment is to demonstrate how to connect IR sensor to an Arduino Uno, read its output, and interface multiple sensors to the Arduino how Infrared Sensor Work.

Assignment 8: Aim of this assignment is to illustrate the connections between the ESP32 and the servo motor Servo Moto and to create a Servo Sweep programme that continuously oscillates between the extremes of left and right.

Assignment 9: Aim of this assignment is to track the frequency of usage of the Smart hand Sanitizer and to dispense the sanitizer without requiring physical contact.

Assignment 10: Aim of this assignment is to using an IR flame sensor, LCD, and speaker is to provide an effective and timely response to the presence of a fire or flame. The system is designed to detect the infrared radiation emitted by flames, process this information, and trigger visual and audible alerts to notify occupants or relevant authorities about the potential fire hazard.

Assignment 11: Aim of this assignment is to demonstrate Automatic Room Lighting System using Arduino Uno, PIR Motion Sensor, LDR (Light Dependent Resistor), LED Lights, Resistors, Breadboard, and jumper wires is to create an intelligent and energy-efficient lighting system that automatically adjusts the room lighting based on occupancy and ambient light levels.

Assignment 12: Aim of this assignment helps to understand the functionalities of Mapper and Reducer Classes to find out the frequency of each word.

Assignment 13: Aim of this assignment is to create and apply Linear and Logistic Regression Models for a simple linear regression model using Python and the popular machine learning library, scikit-learn. It helps to make predictions or understand the strength and nature of the association between the variables.

Assignment 14: Aim of this assignment is to performing data analysis with machine learning methods is to gain insights, make predictions, and automate decision-making processes using computational models.

Assignment 15: Aim of this assignment is to performing graphical data analysis using Python to visually explore, analyze, and communicate insights from data and Understand relationships between different variables or features.

Learning Outcomes

1: Understand and explain the concepts of IoT.

2: Analyze and evaluate various sensors with Arduino.

3: Create and evaluate different statistical models.

2. **Prerequisites:**
   Basic knowledge of computer network, Programming Language.

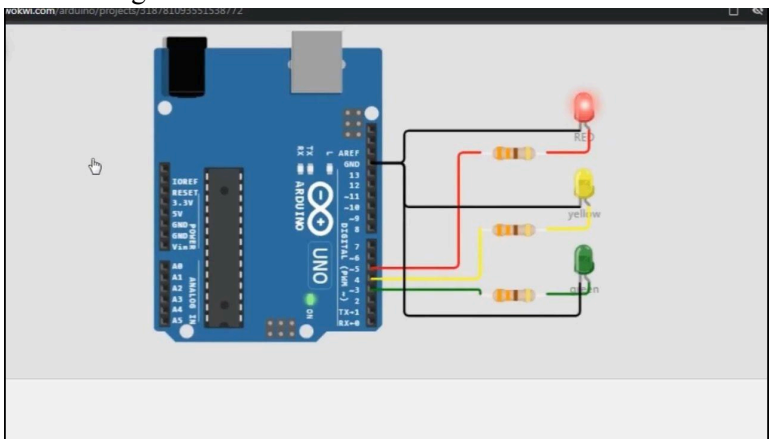
3. **Software required:**
   Wokwi Simulator

4. **Introduction and Theory**

Link: https://wokwi.com/projects/390330524135532545

Assignment 1: Using the data from the sensors, it will decide how dense the traffic is in each lane, and based on that, it will control the traffic signals, which will then take advantage of any traffic signals. LEDs were used in the creation of the system's traffic signals. Red, Yellow and green LEDs are present on each signal.

Circuit Diagram



Sample code:

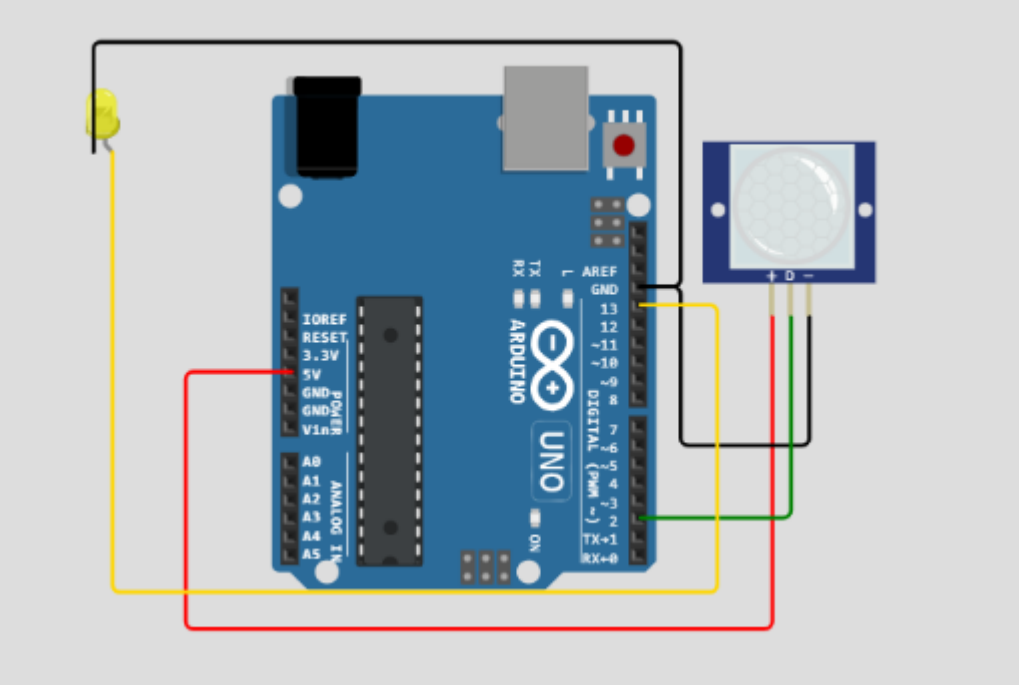
```
// Define the pins to which the traffic signal LEDs are connected
```

```
const int redPin = 10;    // Red LED connected to pin 10
const int yellowPin = 9;  // Yellow LED connected to pin 9
const int greenPin = 8;   // Green LED connected to pin 8

// Define the duration of each phase in milliseconds
const int redDuration = 5000;     // 5000 milliseconds (5 seconds)
const int yellowDuration = 2000;  // 2000 milliseconds (2 seconds)
const int greenDuration = 5000;   // 5000 milliseconds (5 seconds)

// Setup function runs once when the Arduino is powered on or restarted
void setup() {
  // Set the LED pins as outputs
  pinMode(redPin, OUTPUT);
  pinMode(yellowPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
}

// Loop function runs repeatedly as long as the Arduino is powered on
void loop() {
  // Red phase
  digitalWrite(redPin, HIGH);
  digitalWrite(yellowPin, LOW);
  digitalWrite(greenPin, LOW);
  delay(redDuration);

  // Yellow phase
  digitalWrite(redPin, LOW);
  digitalWrite(yellowPin, HIGH);
  digitalWrite(greenPin, LOW);
  delay(yellowDuration);

  // Green phase
  digitalWrite(redPin, LOW);
  digitalWrite(yellowPin, LOW);
  digitalWrite(greenPin, HIGH);
  delay(greenDuration);
}
```

https://wokwi.com/projects/390230500008796161

Assignment 2.1: It counts visitors in a room, as they enter through a narrow corridor containing one PIR. It provides simple practice to Make an LED connected to a specific pin on the Arduino blink on and off.

Circuit Diagram:



Sample code:

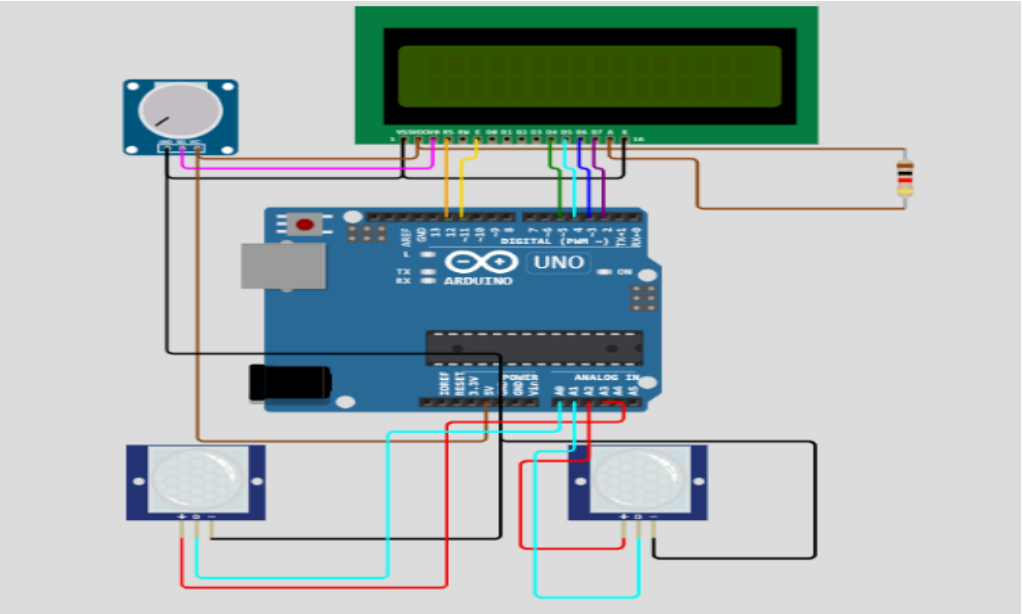
```
int val=0;
int pirstate= LOW;
int c=0;
void setup() {
  // put your setup code here, to run once:
pinMode(13,OUTPUT);
pinMode(2, INPUT);
Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
val=digitalRead(2);
if(val==HIGH)
{
  digitalWrite(13, HIGH);
  if(pirstate==LOW)
  {
    Serial.println("Motion detected");
    c++;
    Serial.println(c);
```

```
    pirstate=HIGH;
  }
}
  else
  {
  digitalWrite(13, LOW);
  if(pirstate==HIGH)
  {
    Serial.println("Motion Ended");
    pirstate=LOW;
  }
  }
}
```

2.2 Bidirectional visitors counter: It counts visitors in a room, as they enter through a narrow corridor containing two PIRs, passing from PIR1 to PIR2 and decreases them when exiting backwards. It provides simple practice to Make an LCD connected to a specific pin on the Arduino blink.

https://wokwi.com/projects/390590775847013377

Circuit Diagram:



Sample code:

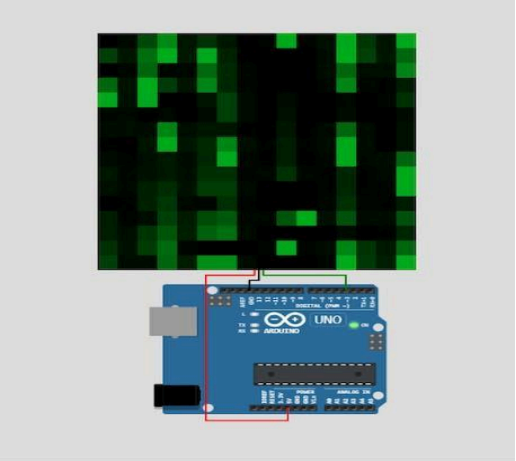
```
#include <LiquidCrystal.h>
int in = 15;
int inpr = 16;
int out = 14;
int outpr = 17;
```

```
int ppl = 0;
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
bool pi = 0;
bool po = 0;
void setup() {
  // put your setup code here, to run once:
   pinMode(15, INPUT);
  pinMode(14, INPUT);
  pinMode(16, OUTPUT);
  pinMode(17, OUTPUT);
  lcd.begin(16, 2);

}

void loop() {
  // put your main code here, to run repeatedly:
  lcd.clear();
  digitalWrite(outpr, HIGH);
  digitalWrite(inpr, HIGH);
  pi = digitalRead(in);
  po = digitalRead(out);
  if (pi == 1){
    ppl--;
    delay(500);
  }
  else if (po == 1){
    ppl++ ;
    delay(500);
  }
  ppl = constrain(ppl, 0, 50);
  lcd.setCursor(0, 0);
  lcd.print("PEOPLE IN:");
  lcd.setCursor(11, 0);
  lcd.print(ppl);
  if (ppl >= 20){
     lcd.setCursor(0, 1);
     lcd.print("PLEASE WAIT");
     delay(1000);
  }
  if (ppl <= 19){
     lcd.setCursor(0, 1);
     lcd.print("PLEASE VISIT");
     delay(1000);
  }

}
```

https://wokwi.com/projects/390330524135532545

Assignment 3: The raindrop sensor measures the moisture via analog output pins and it provides a digital output when a threshold of moisture exceeds. The modules are equipped with a sensor that detects raindrops and outputs a digital signal to the Arduino.

Circuit Diagram



Sample code:

Program Code:

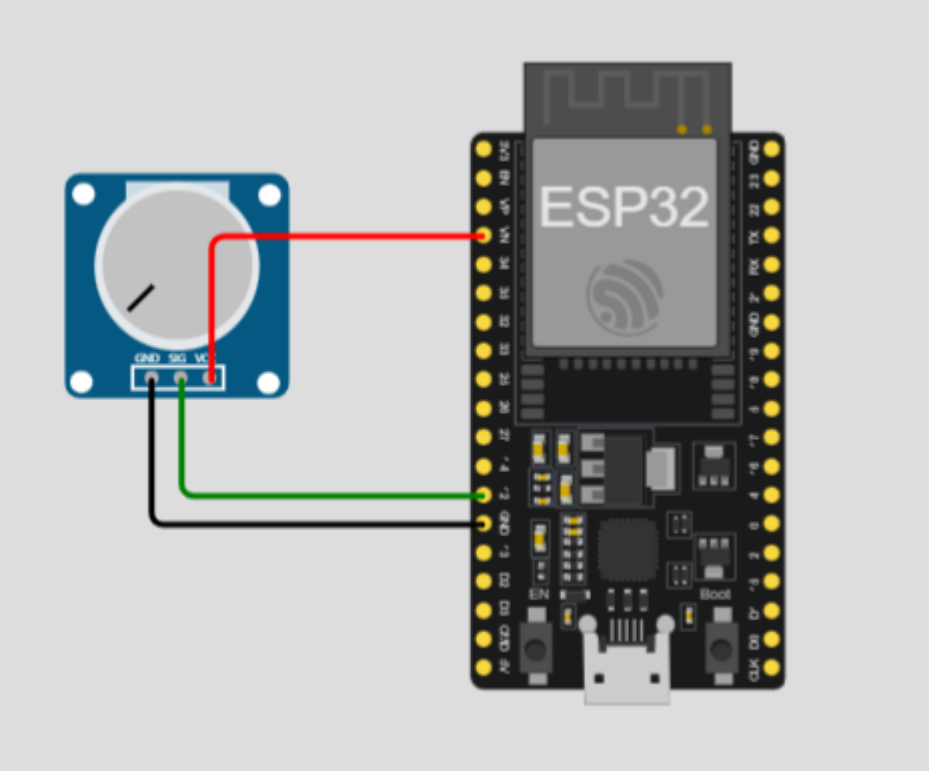
```
//Digital Rain implementation
//fastled 16x16 matrix demo
//Yaroslaw Turbin 24.08.2020
//https://vk.com/ldirko
//https://www.reddit.com/user/ldirko/

#include "FastLED.h"

// Matrix size
#define NUM_ROWS 16
#define NUM_COLS 16
// LEDs pin
#define DATA_PIN 3
// LED brightness
#define BRIGHTNESS 255
#define NUM_LEDS NUM_ROWS * NUM_COLS
// Define the array of leds
CRGB leds[NUM_LEDS];
byte rain[NUM_LEDS];
byte counter = 1;
int speed = 1;

void setup() {
  FastLED.addLeds<NEOPIXEL, DATA_PIN>(leds, NUM_LEDS);
  FastLED.setBrightness(BRIGHTNESS);
  raininit();
}
```

```
void loop() {
  EVERY_N_MILLISECONDS(100) {
    updaterain();
    FastLED.show();
  }
  EVERY_N_MILLISECONDS(30) {
    changepattern();
  }
} //loop

void changepattern () {
  int rand1 = random16 (NUM_LEDS);
  int rand2 = random16 (NUM_LEDS);
  if ((rain[rand1] == 1) && (rain[rand2] == 0) )   //simple get two random dot 1 and 0 and
swap it,
  {
    rain[rand1] = 0;  //this will not change total number of dots
    rain[rand2] = 1;
  }
} //changepattern

void raininit() {                                 //init array of dots. run once
  for (int i = 0; i < NUM_LEDS; i++) {
    if (random8(20) == 0) {
      rain[i] = 1;  //random8(20) number of dots. decrease for more dots
    }
    else {
      rain[i] = 0;
    }
  }
} //raininit

void updaterain() {
  for (byte i = 0; i < NUM_COLS; i++) {
    for (byte j = 0; j < NUM_ROWS; j++) {
      byte layer = rain[XY(i, ((j + speed + random8(2) + NUM_ROWS) % NUM_ROWS))];   //fake
scroll based on shift coordinate
      // random8(2) add glitchy look
      if (layer) {
        leds[XY((NUM_COLS - 1) - i, (NUM_ROWS - 1) - j)] = CHSV(110, 255, BRIGHTNESS);
      }
    }
  }

  speed ++;
  fadeToBlackBy(leds, NUM_LEDS, 40);
  blurRows(leds, NUM_COLS, NUM_ROWS, 16);      //if you want
} //updaterain

uint16_t XY (uint8_t x, uint8_t y) {
  return (y * NUM_COLS + x);
}
```

https://wokwi.com/projects/391219930220646401

Assignment 4.1: Soil Moisture can measure the moisture content in the soil based on the change in resistance between the two conducting plates. The resistance between the two conducting plates varies in an inverse manner with the amount of moisture present in the soil. To focus on a moisture sensor is a device used to measure the moisture level in soil or other materials.

Circuit Diagram:



Program:

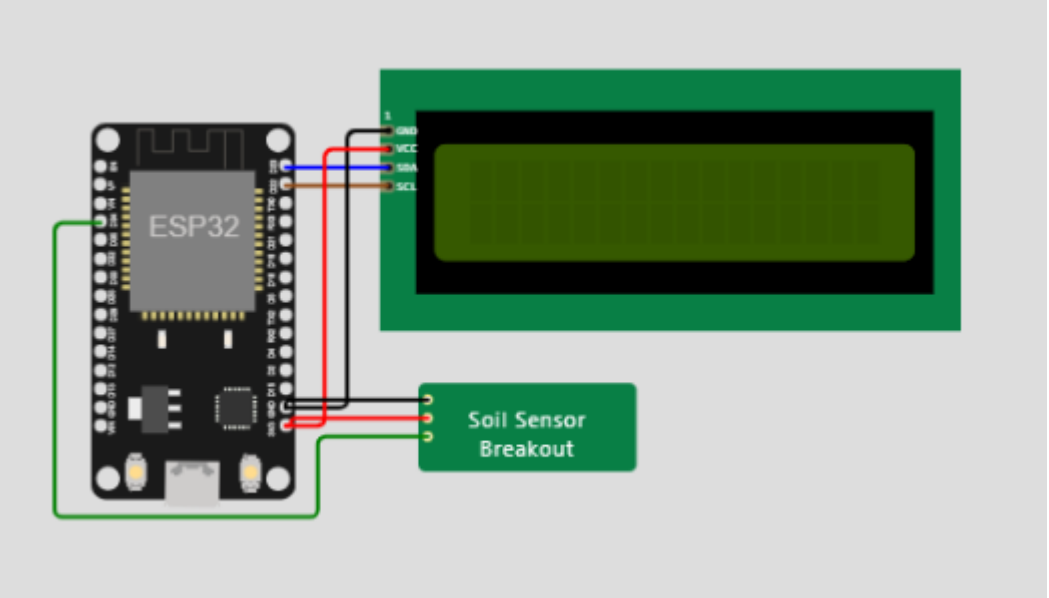
```
#define MoisensorPin 12
int MoisensorValue = 0;
int limit = 55;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.println("Hello, ESP32!");
}

void loop() {
  // put your main code here, to run repeatedly:
  //Read Soil moisture Sensor value
```

```
  MoisensorValue = analogRead(MoisensorPin);                    //sprintf(buffer,
"Soil Moisture Raw: %d", MoisensorValue);Serial.println(buffer);
  MoisensorValue = map(MoisensorValue, 0, 4095, 0, 100);        // 0 to 100%
  MoisensorValue = 100 - MoisensorValue;
  delay(10); // this speeds up the simulation

  Serial.print("MoisensorValue : ");Serial.print(MoisensorValue);Serial.println("");
}
```

https://wokwi.com/projects/355385730506292225

Assignment 4.2: Soil Moisture can measure the moisture content in the soil based on the change in resistance between the two conducting plates. The resistance between the two conducting plates varies in an inverse manner with the amount of moisture present in the soil. To focus on a moisture sensor is a device used to measure the moisture level in soil or other materials.

Circuit Diagram:



Program:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);

void setup()
{
  Wire.begin(23, 22);
  Serial.begin(9600);
  lcd.init();
```

```
  lcd.backlight();
}

void loop()
{
  int16_t i = analogRead(34);
  String msg = i < 2165 ? "WET" : i > 3135 ? "DRY" : "OK";
  lcd.clear();
  lcd.print("Soil: ");
  lcd.print(msg);
  delay(500);
}
```

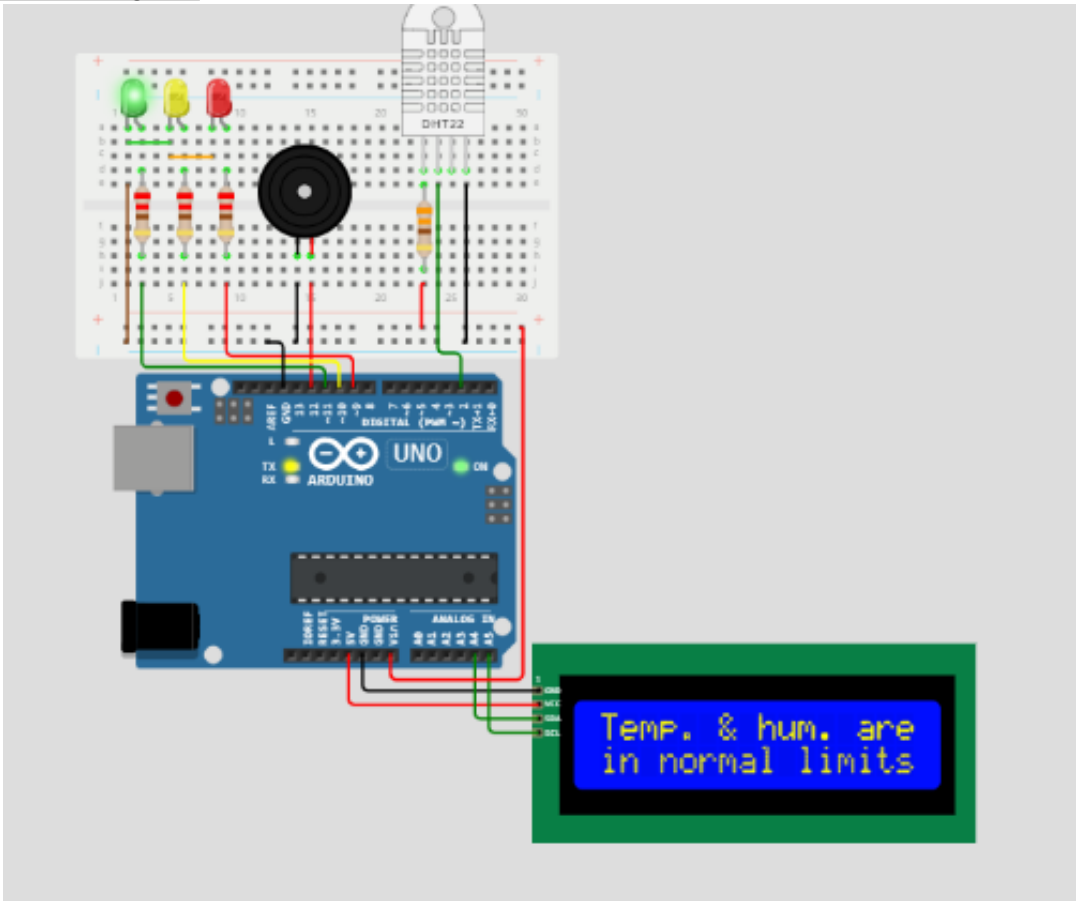
Soil_sensor.chip.c

```
#include "wokwi-api.h"
#include <stdio.h>
#include <stdlib.h>

typedef struct {
  pin_t pin;
  float moisture;
} chip_data_t;

void chip_timer_callback(void *data)
{
  chip_data_t *chip_data = (chip_data_t*)data;
  float moisture = attr_read(chip_data->moisture);
  float volts = 5 * (moisture / 4096.0);
  pin_dac_write(chip_data->pin, volts);
}

void chip_init()
{
  chip_data_t *chip_data = (chip_data_t*)malloc(sizeof(chip_data_t));
  chip_data->moisture = attr_init("moisture", 2910.0);
  chip_data->pin = pin_init("A0", ANALOG);

  const timer_config_t config =
  {
    .callback = chip_timer_callback,
    .user_data = chip_data,
  };

  timer_t timer_id = timer_init(&config);
  timer_start(timer_id, 1000, true);
}
```

Reference: https://youtu.be/oz5OrfTWpNM
Reference: https://www.tinkercad.com/things/6BWhOjG4R1j-

Assignment 5: Temperature sensor LM35 and Arduino Uno are the hardware used interfaced with computer, and the temperature is controlled in the room. Temperature is displayed on LCD display employing A1 pin of hardware with the help of analog pin utilizing pulse width modulation (PWM).

Circuit Diagram:



Sample code:

```
/* How to use the DHT-22 sensor with Arduino uno.
   Is a temperature and humidity sensor!
   See it in original form:

 https://create.arduino.cc/projecthub/mafzal/temperature-monitoring-with-dht22-arduino-15b013
*/
//LCD I2C library:
#include <LiquidCrystal_I2C.h>
//DHT22 sensor library:
#include <DHT.h>;
```

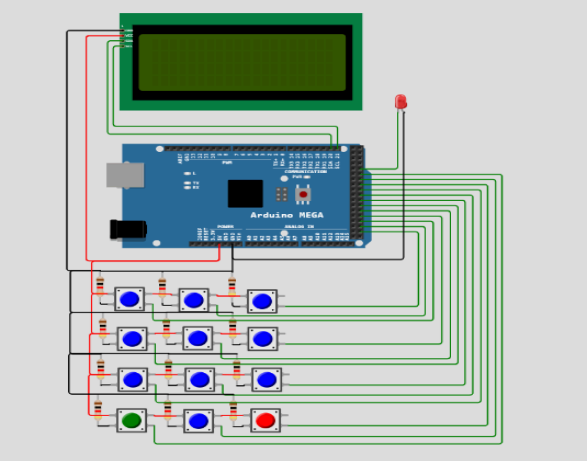
```
//LCD I2C address 0x27, 16 column and 2 rows!
LiquidCrystal_I2C lcd(0x27, 16, 2);

//Constants:
#define DHTPIN 2          //what pin we're connected to
#define DHTTYPE DHT22     //DHT 22  (AM2302)
DHT dht(DHTPIN, DHTTYPE); //Initialize DHT sensor for normal 16mhz Arduino
//Variables:
float H; //Humidity value
float T; //Temperature value
int buzzer = 12;

//Initialize LCD, DHT22 sensor and buzzer:
void setup(){
  lcd.init(); lcd.backlight();  dht.begin();  pinMode(buzzer, OUTPUT);
  //Print some text in Serial Monitor
  Serial.begin(9600);  Serial.println("DHT22 sensor with Arduino Uno R3!");
  pinMode(9, OUTPUT);  pinMode(10, OUTPUT);  pinMode(11, OUTPUT);
}

void loop(){
  delay(2000);
  //Read data and store it to variables hum and temp
  H = dht.readHumidity();  T = dht.readTemperature();

  //Print temp and humidity values to serial monitor
  Serial.print("Humidity: ");
  Serial.print(H);
  Serial.println(" %; ");
  Serial.print("Temperature: ");
  Serial.print(T);
  Serial.println(" Celsius.\n");

  /*If humidity is higher than 70% &
  temperature is higher than 30 degrees Celsius
  then it will show on LCD „Too warm! Cool down!”*/
  if(H >= 70.00 && T >= 30.00){
    digitalWrite(9, HIGH);  digitalWrite(10, LOW);  digitalWrite(11, LOW);

    lcd.println("   Too warm!    ");
    lcd.setCursor(0, 1);
    lcd.println("   Cool down!   ");
    lcd.setCursor(0, 0);

    digitalWrite(buzzer, 1); tone(buzzer, 900, 100);
    delay(400);
    digitalWrite(buzzer, 0); tone(buzzer, 900, 100);
```

```
    delay(400);
    digitalWrite(buzzer, 1); tone(buzzer, 900, 100);
    delay(400);
    digitalWrite(buzzer, 0); tone(buzzer, 900, 100);
    delay(400);
  }else{
  /*If humidity is lower than 70% &
  temperature is lower than 30 degrees Celsius
  then it will show on LCD „Temp. & hum. are in normal limits”*/
    digitalWrite(9, LOW);  digitalWrite(10, LOW);  digitalWrite(11, HIGH);
    lcd.println("Temp. & hum. are");  lcd.setCursor(0, 1);
    lcd.println("in normal limits");  lcd.setCursor(0, 0);
    digitalWrite(buzzer, 0);
  }

  /*If either humidity is lower than 70%, but
  temperature is higher than 30 degrees Celsius,
  then it will show on LCD „Be ware! Temp. too high” or
  humidity is higher than 70%, but
  temperature is lower than 30 degrees Celsius, then
  it will show on LCD „Be ware! Hum. too high”*/
  if(H < 70.00 && T >= 30.00){
    digitalWrite(9, LOW);  digitalWrite(10, HIGH);  digitalWrite(11, LOW);
    lcd.println("Be ware!        ");  lcd.setCursor(0, 1);
    lcd.println("Temp. too high! ");  lcd.setCursor(0, 0);
    digitalWrite(buzzer, 1); tone(buzzer, 400, 400);  delay(400);
    digitalWrite(buzzer, 0); tone(buzzer, 400, 400);  delay(400);
  }
  if(H >= 70.00 && T < 30.00){
    digitalWrite(9, LOW);  digitalWrite(10, HIGH);  digitalWrite(11, LOW);
    lcd.println("Be ware!        ");  lcd.setCursor(0, 1);
    lcd.println("Hum. too high!  ");  lcd.setCursor(0, 0);
    digitalWrite(buzzer, 1); tone(buzzer, 400, 400);  delay(400);
    digitalWrite(buzzer, 0); tone(buzzer, 400, 400);  delay(400);
  }
}
```

https://wokwi.com/projects/373753338411104257

- **Assignment 6:** Touch Sensor enables the user to replace the press with touch. It can detect the change in capacitance when a finger is nearby. That means no matter your finger directly touches the pad or just stays close to the pad. When we touch the things that we have kept or connected to the Arduino, it will be used to blink the LED.

Circuit Diagram



Sample code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

/*

Things needed are:
1. Arduino Mega
2. 9 TTP223 capacitive touch sensors
(https://hobbycomponents.com/sensors/901-ttp223-capacitive-touch-sensor)
3. 20x4 I2C LCD
4. Some wires/cabling as needed
5. An LED
*/

/*
The board is Arduino Mega
LCD 16x2 located at I2C addr: 0x27
*/

LiquidCrystal_I2C screen(0x27, 20, 4);

// LED connected to 29
int ledPin = 29;

enum keypad_pin {
  KEYPAD_NUM0,
  KEYPAD_NUM1,
  KEYPAD_NUM2,
  KEYPAD_NUM3,
  KEYPAD_NUM4,
  KEYPAD_NUM5,
  KEYPAD_NUM6,
  KEYPAD_NUM7,
```

```
  KEYPAD_NUM8,
  KEYPAD_NUM9,
  KEYPAD_CLEAR,
  KEYPAD_CHANGE_PIN,
  KEYPAD_COUNT
};

// The button should be TTP223 but it
// doesnt exist on simulator

// Pins where the buttons connected to
static int buttonToDigitalPin[KEYPAD_COUNT] = {
  [KEYPAD_NUM0] = 33,
  [KEYPAD_NUM1] = 37,
  [KEYPAD_NUM2] = 39,
  [KEYPAD_NUM3] = 41,
  [KEYPAD_NUM4] = 43,
  [KEYPAD_NUM5] = 45,
  [KEYPAD_NUM6] = 47,
  [KEYPAD_NUM7] = 49,
  [KEYPAD_NUM8] = 51,
  [KEYPAD_NUM9] = 53,
  [KEYPAD_CLEAR] = 31,
  [KEYPAD_CHANGE_PIN] = 35
};

enum global_state {
  GLOBAL_STATE_NOP,
  GLOBAL_STATE_ASKING_PIN,
  GLOBAL_STATE_CHECK_PIN,
  GLOBAL_STATE_WAIT_ANY_KEY
};

enum button_state {
  BUTTON_PRESSED,
  BUTTON_RELEASED
};

bool buttonStatePrev[KEYPAD_COUNT] = {false};
bool buttonStateCurrent[KEYPAD_COUNT] = {false};
enum button_state buttonState[KEYPAD_COUNT] = {};
enum global_state globalState = GLOBAL_STATE_NOP;

// 6 PIN code
#define PIN_SIZE 6

int currentPIN[PIN_SIZE] = {5,5,7, 7,5,5};
```

```
int currentPos = 0;
int pinBuffer[PIN_SIZE] = {};

void lockTheThing() {
  // digitalWrite(ledPin, LOW);
}

void unlockTheThing() {
  digitalWrite(ledPin, HIGH);
}

void setup() {
  for (int i = 0; i < KEYPAD_COUNT; i++)
    pinMode(buttonToDigitalPin[i], INPUT);

  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);

  screen.init();
  screen.backlight();
  screen.setCursor(0, 0);
  screen.print("Hello There UwU");
  screen.setCursor(0, 1);
  screen.blink();

  lockTheThing();
}

void clearLine(int y) {
  screen.setCursor(0, y);
  screen.print("                    ");
  screen.setCursor(0, y);
}

void clearScreen() {
  screen.clear();
  screen.setCursor(0, 3);
  screen.print("           By Fox <3");
}

#define EINVAL 1
#define ENODATA 2

// -ENODATA: No button pressed
// -EINVAL: non numeric buttons
int getNumberAndPrint() {
```

```
  int ret = -ENODATA;
  for (int i = 0; i < KEYPAD_COUNT; i++) {
    if (buttonState[i] == BUTTON_PRESSED) {
      ret = i - KEYPAD_NUM0;
      break;
    }
  }

  if (ret > 9)
    ret = -EINVAL;
  if (ret >= 0)
    screen.print(ret);
  return ret;
}

void loop() {
  for (int i = 0; i < KEYPAD_COUNT; i++)
    buttonStateCurrent[i] = digitalRead(buttonToDigitalPin[i]) == HIGH;

  for (int i = 0; i < KEYPAD_COUNT; i++) {
    if (buttonStatePrev[i] == false && buttonStateCurrent[i] == true)
      buttonState[i] = BUTTON_PRESSED;
    else
      buttonState[i] = BUTTON_RELEASED;
  }

  // button_manager_poll();

  int buttonActive = 0;
  for (int i = 0; i < KEYPAD_COUNT; i++)
    buttonActive += buttonStateCurrent[i] ? 1 : 0;

  // We only care if there one button active
  if (buttonActive != 1)
    goto dont_care;

  screen.noBlink();

  switch (globalState) {
    case GLOBAL_STATE_WAIT_ANY_KEY:
      // No button was recently pressed
      if (getNumberAndPrint() == -ENODATA)
        break;

      globalState = GLOBAL_STATE_NOP;
      lockTheThing();
      clearScreen();
```

```
      break;
    case GLOBAL_STATE_NOP:
      clearScreen();

      screen.setCursor(0, 0);
      screen.print("Enter PIN UwU:");
      screen.setCursor(3, 1);
      screen.print("-");
      screen.setCursor(0, 1);
      currentPos = 0;
      globalState = GLOBAL_STATE_ASKING_PIN;
    case GLOBAL_STATE_ASKING_PIN:
      // Always place the seperator
      screen.setCursor(3, 1);
      screen.print("-");
      screen.setCursor(currentPos + (currentPos / 3), 1);

      int num = getNumberAndPrint();
      if (num < 0 && buttonState[KEYPAD_CLEAR] == BUTTON_PRESSED) {
        clearLine(1);
        currentPos = 0;
        goto processing_done;
      }

      if (num >= 0 && num <= 9) {
        pinBuffer[currentPos] = num;
        currentPos++;
      }

      if (currentPos == PIN_SIZE)
        globalState = GLOBAL_STATE_CHECK_PIN;
      else
        break;
    case GLOBAL_STATE_CHECK_PIN:
      clearScreen();
      screen.setCursor(0, 0);
      currentPos = 0;

      // Check pin
      if (memcmp(pinBuffer, currentPIN, sizeof(pinBuffer)) != 0) {
        screen.print("Incorrect PIN! :(");
        goto wrong_pin_wait_for_any_key;
      }

      screen.print("Correct PIN! :)");

      // At here maybe open something
```

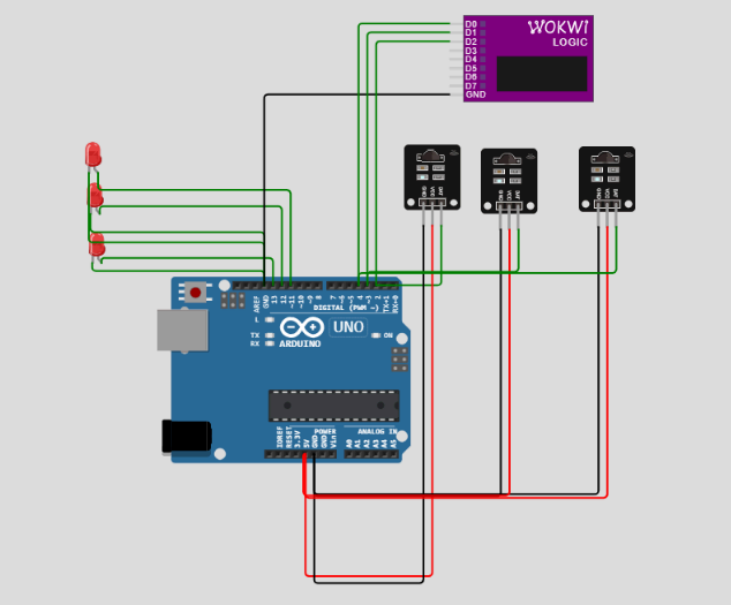
```
      // protected
      unlockTheThing();
wrong_pin_wait_for_any_key:
      // Wait for any key
      screen.setCursor(0, 1);
      screen.print("Press any key :3");
      globalState = GLOBAL_STATE_WAIT_ANY_KEY;
      break;
  }

processing_done:
  screen.blink();
dont_care:
  memcpy(buttonStatePrev, buttonStateCurrent, sizeof(buttonStateCurrent));
  delay(10);
}
```

```
public class MyClass {
   public void method1() {
      System.out.println("Method 1 called.");
   }

   public void method2() {
      System.out.println("Method 2 called.");
   }

   public static void main(String[] args) {
      MyClass obj = new MyClass();
      obj.method1();
      obj.method2();
   }
}
```

https://wokwi.com/projects/353576792228920321

Assignment 7: Connecting the IR sensor to any microcontroller is simple. This sensor outputs a digital signal and processing this signal is very easy. To check the port in an infinite loop to see when the port changes its state from high to low. Power the IR with 5V or 3.3V and connect ground to ground. Then connect the output to a LED.

Circuit Diagram:



Sample code:

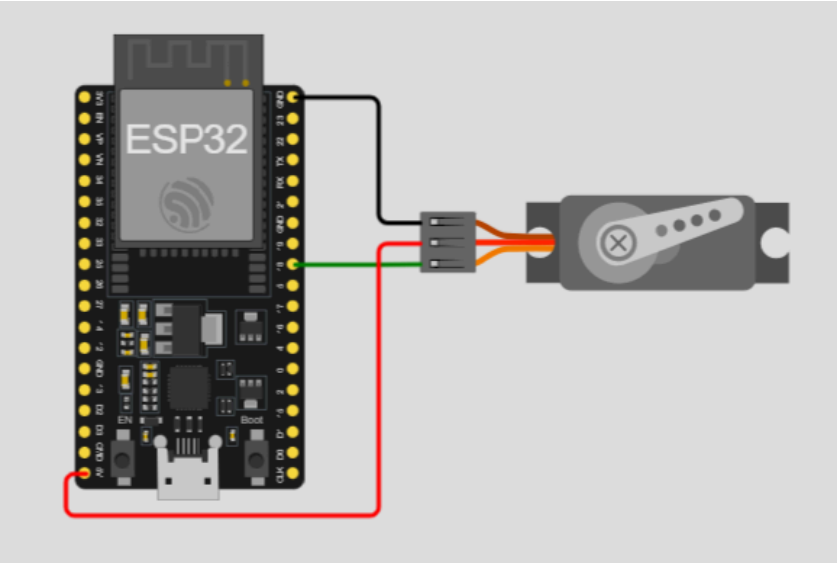
```
int LED1 = 13;
int LED2 = 12;
int LED3 = 11;
int IR1 = 2;
int IR2 = 3;
int IR3 = 4;
int val1 = 0;
int val2 = 0;
int val3 = 0;

void setup() {
  pinMode(IR1, INPUT);
  pinMode(IR2, INPUT);
  pinMode(IR3, INPUT);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  Serial.begin(9600);

}

void loop() {
  val1 = digitalRead(IR1);
  val2 = digitalRead(IR2);
  val3 = digitalRead(IR3);

  if (val1 == 0) {
```

```
    digitalWrite(LED1, HIGH);
  }
  else {
    digitalWrite(LED1, LOW);
  }
  if (val2 == 0) {
    digitalWrite(LED2, HIGH);
  }
  else {
    digitalWrite(LED2, LOW);
  }
  if (val3 == 0) {
    digitalWrite(LED3, HIGH);
  }
  else {
    digitalWrite(LED3, LOW);
}
delay(200);

}
```

https://wokwi.com/projects/323706614646309460

Assignment 8: Utilizing an ESP32 Development Board to drive a servo motor. Create a sweeping program that causes the servo to oscillate back and forth in order to show how the ESP32 Servo Control works. Then, it Demonstrates how to use a potentiometer to control the servo.

Circuit Diagram:



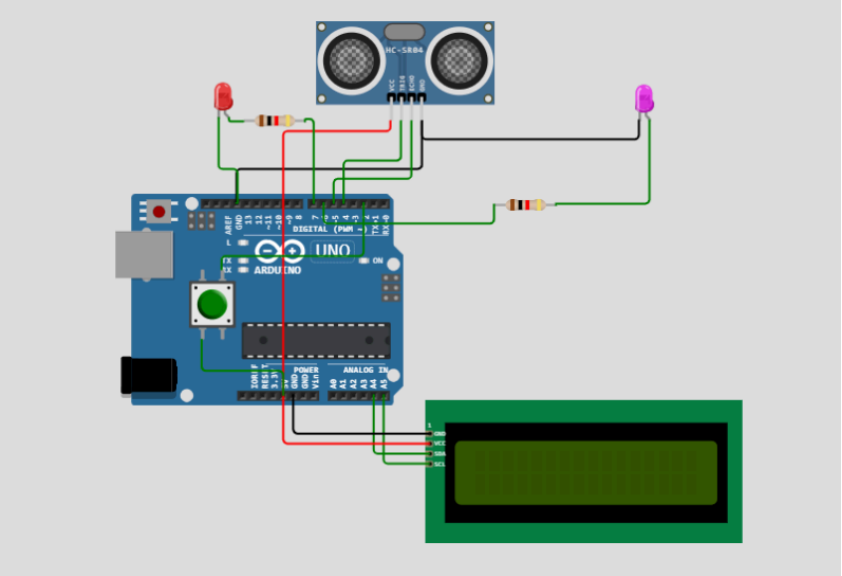
Sample code:

```
// Servo Sweep example for the ESP32
// https://wokwi.com/arduino/projects/323706614646309460

#include <ESP32Servo.h>

const int servoPin = 18;

Servo servo;

void setup() {
  servo.attach(servoPin, 500, 2400);
}

int pos = 0;

void loop() {
  for (pos = 0; pos <= 180; pos += 1) {
    servo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    servo.write(pos);
    delay(15);
  }
}
```

https://wokwi.com/projects/374689642187872257

Assignment 9: Smart hand sanitizers come equipped with sensors or monitoring systems to track the frequency of usage. This touchless technology to dispense the sanitizer without requiring physical contact.

Circuit Diagram:



Sample code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Define pins
const int trigPin = 4;
const int echoPin = 5;
const int irPin = 2;
const int motorPin = 6;
const int buzzer = 7;
bool d_patch = false;

unsigned long update_time = 0;

// Ultrasonic sensor variables
long duration;
int distance, fullDistance = 4, emptyDistance = 20, percentage;

// Initialize LCD with I2C address 0x27
LiquidCrystal_I2C lcd(0x27, 16, 2);  // If the address is different from 0x27, replace
it accordingly.

void setup() {
  // Initialize pins
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(irPin, INPUT);
  pinMode(motorPin, OUTPUT);
  pinMode(buzzer, OUTPUT);
```

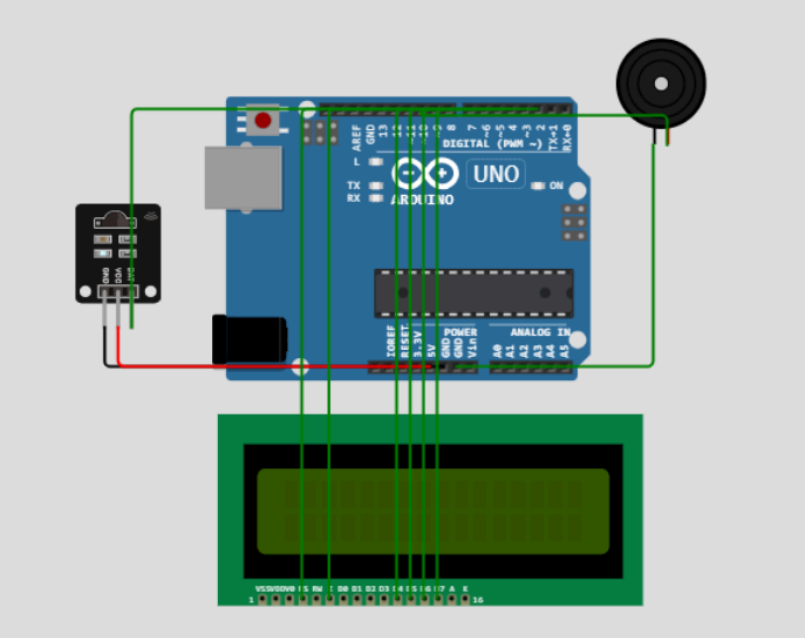
```
  Serial.begin(9600);

  // Initialize LCD
  lcd.init();
  lcd.backlight();

  attachInterrupt( digitalPinToInterrupt(irPin), dispatch, RISING);
}

void loop() {


  if (millis() - update_time > 1000)
  {
    // Check the sanitizer level
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);
    distance = duration * 0.034 / 2;
    // Calculate percentage and display on LCD
    percentage = map(distance, fullDistance, emptyDistance, 100, 0);

    // If the bottle is empty, sound the buzzer
    if (percentage > 0) {
      digitalWrite(buzzer, LOW);
      displayPercentage(percentage);
    } else {
      digitalWrite(buzzer, HIGH);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("  Please ");
      lcd.setCursor(0, 1);
      lcd.print("   Refill ");
    }
    update_time = millis();
  }

  // Check for hand and dispense sanitizer if detected
  if (d_patch == true) {
    activateMotor();
    d_patch = false;
  }

}
```

```
void dispatch ()
{
  d_patch = true;
}


void displayPercentage(int percentage) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Sanitizer Level:");
  lcd.setCursor(0, 1);
  lcd.print(percentage);
  lcd.print("%     ");  // To clear any previous characters
}

void activateMotor() {

  digitalWrite(motorPin, HIGH);
  delay(1000);  // Motor runs for 1 second
  digitalWrite(motorPin, LOW);
}
```

https://wokwi.com/projects/393582668188173313

Assignment 10: The fire alarm system using an IR flame sensor, LCD, and speaker operates by detecting infrared radiation emitted by flames, processing the sensor output, and providing visual and audible alerts to notify occupants of a potential fire hazard.

Circuit Diagram:



Sample code:

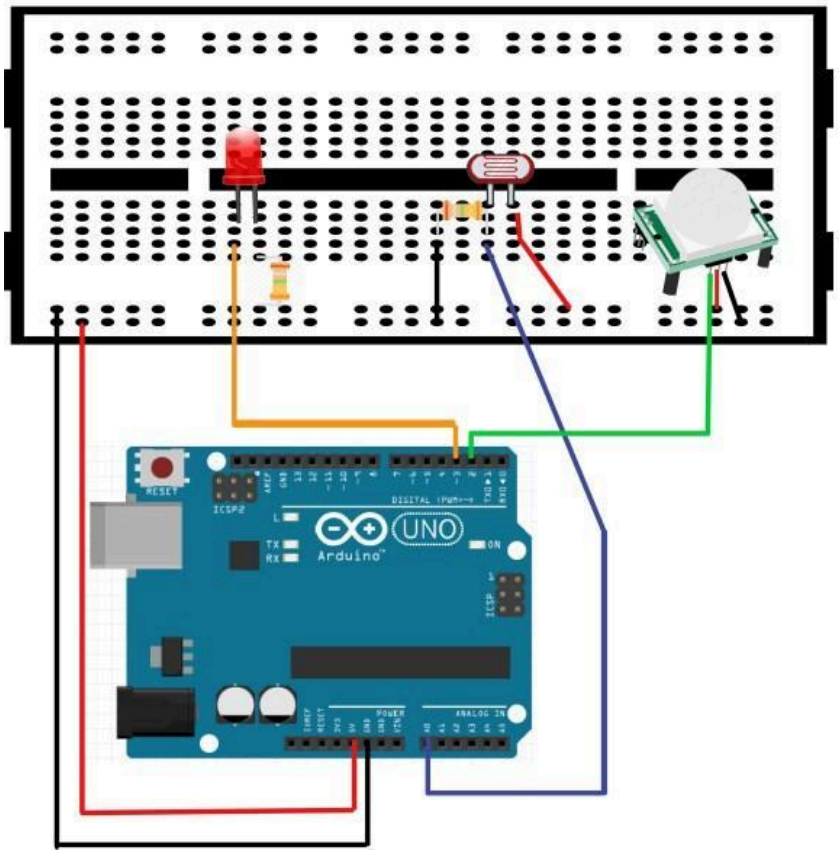
```
#include <LiquidCrystal.h>

// Pin configuration for the IR flame sensor
const int irSensorPin = 2; // Connect the IR sensor to digital pin 2
const int thresholdValue = 800; // Adjust this value to set the sensitivity of the
sensor

// Pin configuration for the LCD
LiquidCrystal lcd(8, 9, 4, 5, 6, 7); // RS, EN, D4, D5, D6, D7

// Pin for the speaker or piezo element
const int speakerPin = 10; // Connect the speaker or piezo to digital pin 10

void setup() {
  // Initialize the LCD
  lcd.begin(16, 2);

  // Initialize the IR sensor pin
  pinMode(irSensorPin, INPUT);

  // Set up the speaker pin
  pinMode(speakerPin, OUTPUT);

  // Set up the initial display message
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Fire Detection");
```

```
}

void loop() {
  // Read the value from the IR sensor
  int irValue = analogRead(irSensorPin);

  // Check if the IR sensor reading is above the threshold
  if (irValue < thresholdValue) {
    // Clear the LCD and print the "Fire detected" message
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Fire detected");

    // Generate a tone to create a sound
    tone(speakerPin, 1000); // You can adjust the frequency as needed

    // You can add additional actions here, like sending a notification.
  } else {
    // Clear the LCD and display a standby message
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Standby");

    // Turn off the tone
    noTone(speakerPin);
  }

  // Delay for a moment to avoid rapid display updates
  delay(1000);
}
```

https://www.sensingthecity.com/motion-light-detection-sensing-with-arduino/

Assignment 11: Creating an Automatic Room Lighting System using an Arduino Uno involves using sensors to detect occupancy and ambient light levels, and controlling the lighting based on the input received to demonstrate Automatic Room Lighting System. It's using a PIR (Passive Infrared) motion sensor for occupancy detection and an LDR (Light Dependent Resistor) for ambient light sensing.

Circuit Diagram:



Sample code:

// Before getting started you should ensure that your wiring with your parts and the arduino match the code below (you may change the pins but you then have to modidfy your code)
// Some things to note: "High" indicated the LED light is on, "LOW' indicates the LED if off. Now let's get started!

// First, initate the Pins. THese pins should correlate to your wiring with your part and yoru arduino.

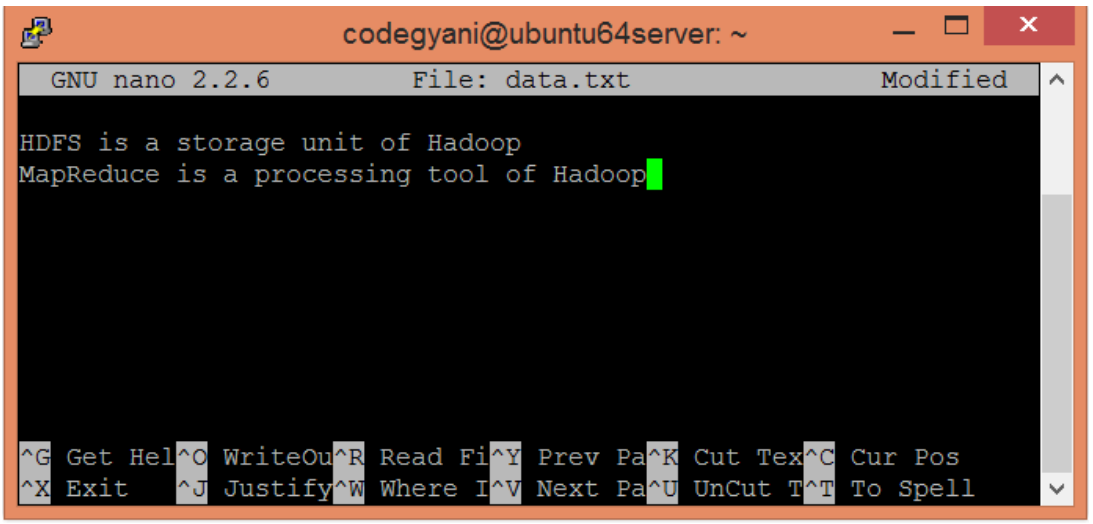
```
#define LDR 0 // Photoresistor to Arduino Pin A0
#define PIR 2 // PIR sensor to Arduino Pin 2
#define LED 3 // LED to Arduino Pin 3

// initate global variables or varibles that can have differnt values inside of the loop
int pirState; // store value for PIR sensor
int ldrValue;  // store value for Photoresistor

void setup() {
  //Serial.begin(9600); // Initialize serial communications with the PC
  pinMode(LED, OUTPUT); //  If LED is activated wil turn on as an ouput
  pinMode(PIR, INPUT); // PIR sensor detects if there is motion
  digitalWrite(LED, LOW); // LED turns off if there is light and or no motion
}
```

```
void loop(){
  ldrValue = analogRead(LDR);  // initialize Photoresistor output
  //Serial.print("Analog reading = ");
  //Serial.println(ldrValue);

  if (ldrValue <= 512) { // if photoresitor detects ldr value less than or equal to 512,
    digitalWrite(LED, HIGH); // then the LED light will turn on.
  }
  else { // ldrValue > 512  // or if the photoresistor detects the ldr to be higher then 512,
    pirState = digitalRead(PIR); //
    if (pirState == HIGH) { // but if the PIR still detetects movement
      digitalWrite(LED, HIGH);  // then the LED light will turn on
      delay(1000); // wait 1000 milieseconds
      digitalWrite(LED, LOW); // LED light will then be turned off
      delay(1000);
    }
    else { // pirState == LOW // if PIR detects no movemt
      digitalWrite(LED, LOW); // then LED light will turn off
    }
  }
  // Note from sourced code: "The processing in the Arduino occurs faster
  // than the response from the PIR, and adding this delay
  // eliminated a flickering on the LED"
  delay(1000);
}
```

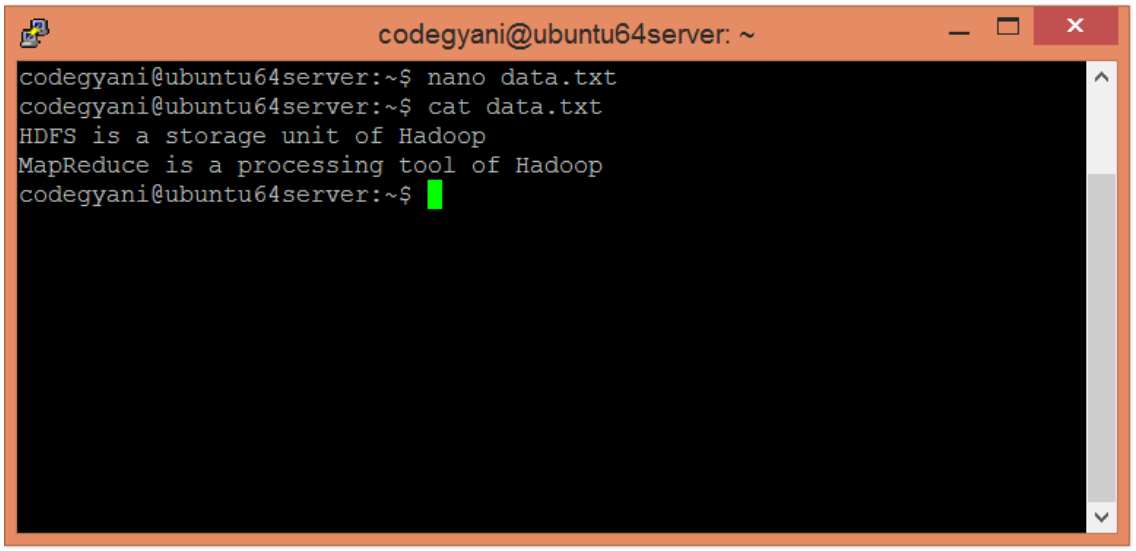
Assignment 12: In MapReduce word count example, to find out the frequency of each word. Here, the role of Mapper is to map the keys to the existing values and the role of Reducer is to aggregate the keys of common values. So, everything is represented in the form of Key-value pair.

Sample code : -

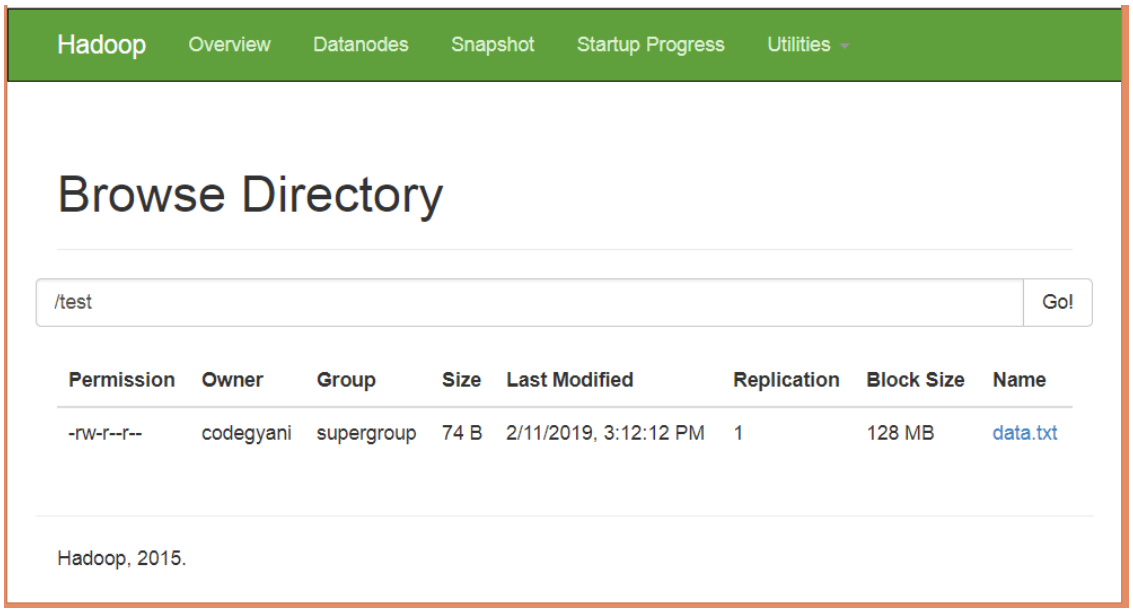
- Create a text file in your local machine and write some text into it.
  $ nano data.txt



- Check the text written in the data.txt file.
  $ cat data.txt



- Create a directory in HDFS, where to kept text file.
  $ hdfs dfs -mkdir /test
- Upload the data.txt file on HDFS in the specific directory.
  $ hdfs dfs -put /home/codegyani/data.txt /test



- Write the MapReduce program using eclipse.

File: WC_Mapper.java

```
package com.javatpoint;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
            Reporter reporter) throws IOException{
        String line = value.toString();
        StringTokenizer  tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()){
            word.set(tokenizer.nextToken());
            output.collect(word, one);
        }
    }

}
```

File: WC_Reducer.java

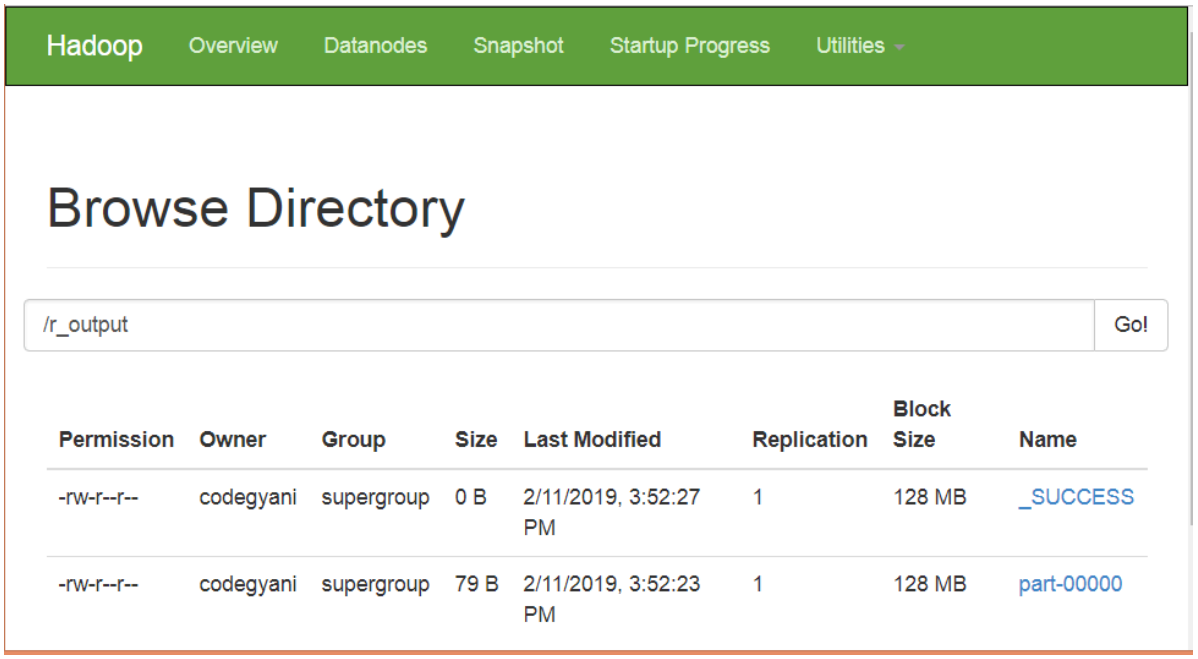
```
package com.javatpoint;
    import java.io.IOException;
    import java.util.Iterator;
    import org.apache.hadoop.io.IntWritable;
    import org.apache.hadoop.io.Text;
    import org.apache.hadoop.mapred.MapReduceBase;
    import org.apache.hadoop.mapred.OutputCollector;
    import org.apache.hadoop.mapred.Reducer;
    import org.apache.hadoop.mapred.Reporter;

    public class WC_Reducer  extends MapReduceBase implements Reducer<Text,IntWritable,Text,IntWritable> {

    public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> output,
     Reporter reporter) throws IOException {
    int sum=0;
    while (values.hasNext()) {
```

```
    sum+=values.next().get();
    }
    output.collect(key,new IntWritable(sum));
    }
    }
```

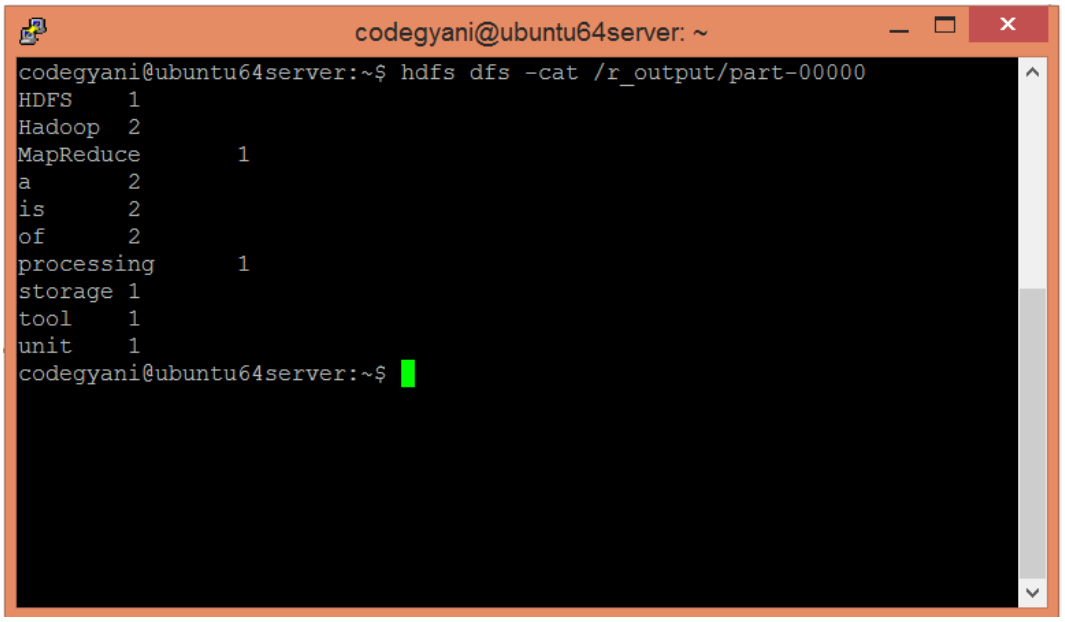
File: WC_Runner.java

```
package com.javatpoint;

    import java.io.IOException;
    import org.apache.hadoop.fs.Path;
    import org.apache.hadoop.io.IntWritable;
    import org.apache.hadoop.io.Text;
    import org.apache.hadoop.mapred.FileInputFormat;
    import org.apache.hadoop.mapred.FileOutputFormat;
    import org.apache.hadoop.mapred.JobClient;
    import org.apache.hadoop.mapred.JobConf;
    import org.apache.hadoop.mapred.TextInputFormat;
    import org.apache.hadoop.mapred.TextOutputFormat;
    public class WC_Runner {
       public static void main(String[] args) throws IOException{
          JobConf conf = new JobConf(WC_Runner.class);
          conf.setJobName("WordCount");
          conf.setOutputKeyClass(Text.class);
          conf.setOutputValueClass(IntWritable.class);
          conf.setMapperClass(WC_Mapper.class);
          conf.setCombinerClass(WC_Reducer.class);
          conf.setReducerClass(WC_Reducer.class);
          conf.setInputFormat(TextInputFormat.class);
          conf.setOutputFormat(TextOutputFormat.class);
          FileInputFormat.setInputPaths(conf,new Path(args[0]));
          FileOutputFormat.setOutputPath(conf,new Path(args[1]));
          JobClient.runJob(conf);
       }
    }
```

- Create the jar file of this program and name it countworddemo.jar.
- Run the jar file
  hadoop jar /home/codegyani/wordcountdemo.jar com.javatpoint.WC_Runner /test/data.txt /r_output
- The output is stored in /r_output/part-00000



- Now execute the command to see the output.
  hdfs dfs -cat /r_output/part-00000



Assignment 13: Creating a simple program for a simple Linear and Logistic Regression model using Python and the popular machine learning library, scikit-learn. Logistic Regression model using the famous Iris dataset for binary classification.

Sample code for program no (i): -

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Generate example data
np.random.seed(42)
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
print(f'Mean Squared Error: {mse}')

# Plot the results
plt.scatter(X_test, y_test, color='black', label='Actual')
plt.plot(X_test, y_pred, color='blue', linewidth=3, label='Predicted')
plt.xlabel('X')
plt.ylabel('y')
plt.title('Linear Regression Model')
plt.legend()
plt.show()
```

1. To generate synthetic data with a linear relationship between X and y, adding some random noise to simulate real-world scenarios.
2. The data is split into training and testing sets using train_test_split from scikit-learn.
3. A linear regression model is created and trained using the training data.
4. Predictions are made on the test set.
5. The mean squared error is calculated to evaluate the model's performance.
6. Finally, the results are plotted using matplotlib.

Sample code for program no (ii): -

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.datasets import load_iris

# Load Iris dataset
iris = load_iris()
X = iris.data[:, :2]  # Use only the first two features for simplicity
y = (iris.target != 0).astype(int)  # Binary classification: setosa (0) vs. versicolor/virginica (1)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')

# Plot the decision boundary
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01), np.arange(y_min, y_max, 0.01))
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, cmap=plt.cm.RdBu, alpha=0.8)
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', cmap=plt.cm.RdBu, marker='o', s=50)
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.title('Logistic Regression Decision Boundary')
plt.show()
```

1. Use the Iris dataset, focusing on the first two features (sepal length and sepal width) for simplicity.
2. The target variable y is modified to be binary: 0 for setosa and 1 for versicolor or virginica.
3. The data is split into training and testing sets.
4. A logistic regression model is trained on the training data.
5. Predictions are made on the test set, and the model is evaluated using accuracy and a confusion matrix.
6. The decision boundary is plotted to visualize the model's classification.

Assignment 14: Creating a simple program to performing data analysis with machine learning methods involves data loading, exploration, preprocessing, model training, evaluation, and visualization. The following program using Python and popular machine learning libraries (NumPy, Pandas, Matplotlib, and scikit-learn) with the famous Iris dataset.

Sample code for program:

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.datasets import load_iris

# Load Iris dataset
iris = load_iris()
X = iris.data[:, :2]  # Use only the first two features for simplicity
y = (iris.target != 0).astype(int)  # Binary classification: setosa (0) vs. versicolor/virginica (1)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')

# Plot the decision boundary
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01), np.arange(y_min, y_max, 0.01))
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, cmap=plt.cm.RdBu, alpha=0.8)
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolors='k', cmap=plt.cm.RdBu, marker='o', s=50)
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.title('Logistic Regression Decision Boundary')
plt.show()
```

1. The Iris dataset is loaded and explored using Pandas, including displaying the head, info, and summary statistics.
2. Visualizations are created using Matplotlib to illustrate the relationships between different features.
3. The dataset is split into training and testing sets.
4. A Random Forest Classifier is trained on the training data.
5. Predictions are made on the test set, and the model is evaluated using accuracy and a classification report.

Assignment 15: Creating a simple program to performing graphical data analysis involves creating visualizations to explore and understand relationships within the data. The following Python program using the popular libraries Matplotlib and Seaborn for graphical data analysis. Use a sample Iris dataset and create various types of plots.

Sample code for program:

```
# Import necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load a sample dataset (Iris dataset)
iris = sns.load_dataset('iris')

# Display the first few rows of the dataset
print("Sample Dataset:")
print(iris.head())

# Pairplot to visualize relationships between numerical features
sns.pairplot(iris, hue='species', markers=['o', 's', 'D'])
plt.title('Pairplot of Iris Dataset')
plt.show()

# Boxplot to visualize distribution and identify outliers
plt.figure(figsize=(10, 6))
sns.boxplot(x='species', y='sepal_length', data=iris)
plt.title('Boxplot of Sepal Length by Species')
plt.show()

# Heatmap to visualize the correlation matrix
correlation_matrix = iris.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Correlation Heatmap of Iris Dataset')
plt.show()

# Violin plot to compare the distribution of petal length for each species
plt.figure(figsize=(10, 6))
sns.violinplot(x='species', y='petal_length', data=iris)
```

```
plt.title('Violin Plot of Petal Length by Species')
plt.show()
```

1. To load a sample dataset (Iris dataset) using Seaborn's load_dataset function.
2. Display the first few rows of the dataset for a quick overview.
3. Create a pairplot to visualize relationships between numerical features, with different marker styles for each species.
4. Use a boxplot to visualize the distribution of sepal length for each species and identify potential outliers.
5. Generate a heatmap to visualize the correlation matrix of numerical features.
6. Create a violin plot to compare the distribution of petal length for each species.

5. Operating Procedure:

Operating an Arduino with sensors involves hardware setup, connecting sensors to the Arduino, programming, and monitoring sensor data. Operating procedures for data analytics using Python and Hadoop involve a combination of tasks related to data processing, analysis, and visualization. Setup Hadoop Cluster, Prepare Data in HDFS, Utilize Python libraries such as scikit-learn, for data analysis and machine learning.

6. Precautions and/or Troubleshooting

To ensure the connection on the components and Debugging while running the program

7. Observations

Take different possible inputs and observe the output

8. Calculations & Analysis

Assignment 1-15: Compile and Run time execution.

9. Result & Interpretation

The result or output have to be shown in computer and write it in the assignment copy. Motivate the students to display the different models using Arduino with Sensors.

10. Follow-up Questions

Assignment 1:

i) Explain the main components of an Arduino board.
ii) What is an LED, and how does it work?
iii) In the context of electronics, what is a register?

Assignment 2:

i) How can a PIR sensor be used for counting visitors in a space?
ii) Describe the basic principle behind using a PIR sensor as a people counter.
iii) What challenges might you encounter when using a PIR sensor for counting people, and how could you address them?

Assignment 3:

i) How does a digital raindrop sensor work to detect rain or water?
ii) Explain the concept of digital signals in the context of a raindrop sensor.
iii) What are the key components of a digital raindrop sensor?

iv) Assignment 4:

i) What is a moisture sensor, and what is its primary function?
ii) How does a moisture sensor work to measure the moisture level in soil or other materials?
iii) Explain the basic principle behind analog and digital moisture sensors.

Assignment 5: -

i) What is the importance of monitoring room temperature?
ii) How is room temperature typically measured in electronic systems?
iii) Describe the basic principles behind temperature sensors.

Assignment 6: -

i) What is a touch sensor, and how does it work?
ii) Explain the basic principle behind capacitive touch sensors.
iii) What are the common applications of touch sensors?

Assignment 7: -

i) What is an infrared (IR) sensor, and how does it work?
ii) Explain the basic principle behind infrared sensing technology.
iii) What are the common applications of infrared sensors?

Assignment 8: -

i) What is a servo motor, and how does it work?
ii) Explain the basic principle behind the operation of a servo motor.
iii) What are the common applications of servo motors?

Assignment 9: -

i) What is a smart hand sanitizer, and how does it differ from a traditional hand sanitizer dispenser?
ii) Explain the basic components and features of a smart hand sanitizer system.
iii) What are the advantages of using a smart hand sanitizer in comparison to manual dispensers?

Assignment 10: -

i) What is a fire alarm system, and what is its primary purpose?
ii) Explain the basic components of a fire alarm system.
iii) How does a fire alarm system differ from a smoke detector?

Assignment 11: -

i) What is an Automatic Room Lighting System, and what is its primary purpose?

ii) Describe the basic components of an Automatic Room Lighting System.
iii) How does it differ from traditional manual lighting systems?

Assignment 12: -

i) Explain the core components of the Hadoop ecosystem.
ii) What is HDFS, and how does it store data across a cluster of machines?
iii) What is MapReduce, and how does it function in the Hadoop framework?
iv) Explain the concepts of Mapper and Reducer in a MapReduce job.

Assignment 13: -

i. What is linear regression, and what is its primary objective?
ii. How do you import the necessary libraries for logistic regression in Python?
iii. Can you explain the basic steps involved in building a logistic regression model?

Assignment 14: -

i) How do you split a dataset into training and testing sets using Python for machine learning?
ii) What Python libraries are commonly used for building machine learning models?
iii) Can you provide an example of how to build a machine learning model using scikit-learn in Python?

Assignment 15: -

i. What Python libraries are commonly used for creating static data visualizations?
ii. Can you compare and contrast Matplotlib and Seaborn for data visualization in Python?
iii. How does Plotly differ from other Python visualization libraries, and what advantages does it offer?

11. Extension and Follow-up Activities (if applicable)

Not applicable

12. Assessments

As per the assessment and evaluation policy of University

13. Suggested readings

1. Analytics for the Internet of Things (IoT): Intelligent analytics for your intelligent devices, Andrew Minteer, Packt Publishing, 2017.
2. Big-Data Analytics for Cloud, IoT and Cognitive Computing, Kai Hwang and Min Chen, Wiley, 1st Edition, 2017.

14. Assignment copy format:

1. Write in A4 page; No channel file will be accepted.
2. Front page and Index page format will be provided by department
3. Following points must be included while writing assignment copy
    a) Problem definition
    b) Algorithm
    c) Program in corresponding language
    d) Output