Zed-King

GROUP OF INSTITUTIONS

ISO 9001 : 2015 Certified





Module: M4-R5.1

[Introduction to Internet of Things (IOT) and its Applications]

Practical Questions

Program1. Write a program to print even number in range of 100 to 200 on Serial monitor.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **USB Cable** to connect the Arduino to your computer
3. **Arduino IDE** installed on your computer for programming the Arduino

Arduino CODE:-

```
void setup() {
// Initialize the serial communication at 9600
baud

Serial.begin(9600);
}
void loop() {
// Print even numbers from 100 to 200

for (int num = 100; num <= 200;
num++) { if (num % 2 == 0) {
    Serial.println(num);
  }
```

```
}
```

// Stop the loop after printing the

Program2. Write a program to interface buzzer with Arduino board to buzz on/off with the delay of 1sec.

Components Needed:

1. **Arduino UNO**
2. **Buzzer**
3. **Jumper Wires**
4. **Breadboard**

Wiring:

- **Connect the buzzer's positive terminal to a digital pin on your Arduino board.** (Let's assume you connect it to pin 13 for this example).
- **Connect the buzzer's negative terminal to ground on the Arduino board.**

Arduino Code:

```
const int buzzerPin = 13;  // Define the buzzer pin

void setup() {
pinMode(buzzerPin, OUTPUT);  // Set the buzzer pin as output
}

void loop() {
```

```
  digitalWrite(buzzerPin, HIGH);  // Turn the buzzer ON
  delay(1000);              // Wait for 1 second
  digitalWrite(buzzerPin, LOW);  // Turn the buzzer OFF
  delay(1000);              // Wait for 1 second
}
```

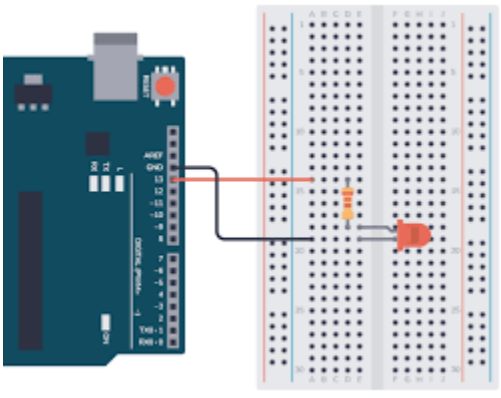
Alternate Code:-

```
const int buzzerPin = 13;  // Define the buzzer pin

void setup() {
pinMode(buzzerPin, OUTPUT);  // Set the buzzer pin as output
}

void loop() {
  tone(13,1000);  // buzzer beep
  delay(1000);    // Wait for 1 second
  noTone(13);  // buzzer OFF
  delay(1000); // Wait for 1 second Turn the
}
```

Program3. Write a program to Blink default Light Emitting Diode (LED) on Arduino board with the delay of 2 sec.



Components Needed:

1. Arduino Uno

2. **LED**
3. **Resistor**
4. **Breadboard and jumper wires**

Wiring:

Connect the Resistor and LED:

- ✓ Insert the longer leg (anode) of the LED into the breadboard.
- ✓ Insert one end of the resistor into the same row as the LED's anode.
- ✓ Insert the other end of the resistor into a GND (ground) row on the breadboard.

Connect to Arduino:

- ✓ Insert a jumper wire from the Arduino's digital pin 13 (or any other digital pin) to the same row as the LED's anode.
- ✓ Insert a jumper wire from the GND pin on the Arduino to a GND row on the breadboard.

Arduino Code:

```
// Blink an LED connected to pin 13, with a 2-second delay.

const int ledPin = 13; // the number of the LED pin

void setup() {

  // initialize the digital pin as an output.
```

```
  pinMode(ledPin, OUTPUT);
}
void loop() {
  digitalWrite(ledPin, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                  // wait for a second
  digitalWrite(ledPin, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                  // wait for a second
}
```

Program4. Write a program to interface Button and LED, so that LED blinks/glow when button is pressed.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **LED** (any standard 5mm LED)
3. **220-ohm Resistor** (to limit current and protect the LED)
4. **Pushbutton** (momentary pushbutton switch)
5. **10k-ohm Resistor** (for the pull-down resistor)
6. **Breadboard** (optional, for easy connections)
7. **Jumper Wires**

Circuit Diagram:

1. **Connect the LED**:
    - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
    - **Cathode (shorter leg)** of the LED to one end of the 220-ohm resistor.
    - **Other end of the resistor** to the Arduino GND (ground) pin.
2. **Connect the Pushbutton**:
    - One terminal of the pushbutton to a digital pin on the Arduino (e.g., pin 7).
    - The other terminal of the pushbutton to the Arduino GND (ground) pin.
    - Connect a 10k-ohm resistor between the button pin (pin 7) and

    - GND to act as a pull-down resistor.

Arduino code:-

```
// Define the pin for the LED and
the button const int ledPin = 9;
const int buttonPin= 7;
void setup() {
 // Set the LED pin as an output
 pinMode(ledPin, OUTPUT);
 // Set the button pin as an input
 pinMode(buttonPin, INPUT);
}
```

```
void loop() {
  // Read the state of the button
  int buttonState = digitalRead(buttonPin);
  // Check if the button is pressed if (buttonState == HIGH)
  {
    // Turn the LED on
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
}
```

Program5. Write a program demonstrating the use of analog output (Pulse with Modulation- PWM) to fade LED.

Components Needed:

1. **Arduino Uno Board** - This will be the main controller.
2. **LED** - This is the component that will fade in and out.
3. **220Ω Resistor** - Protects the LED from too much current.
4. **Jumper Wires** - For making connections.
5. **Breadboard** (optional) - For making the connections in a neat and organized way.

Wiring the Components:

1. **LED:**
   - The **long leg** (anode) of the LED connects to the **PWM pin** on the Arduino (let's use pin **9** as an example).
   - The **short leg** (cathode) of the LED connects to one side of the **220Ω resistor**.
   - The other side of the **220Ω resistor** goes to **GND** (ground) on the Arduino.
2. **PWM Pin:**

   You can use any of the pins marked with a "~" symbol on the Arduino Uno. In this example, we'll use **pin 9** for PWM.

Arduino Code:

```
// Define the pin where the LED is connected
int ledPin = 9;  // PWM-capable pin


void setup() {
  // Set the LED pin as an output
  pinMode(ledPin, OUTPUT);
}


void loop() {
  // Fade in the LED (increase brightness)
  for (int brightness = 0; brightness <= 255; brightness++) {
```

```
    analogWrite(ledPin, brightness);  // Write the brightness to the LED
    delay(10);  // Wait for 10ms to see the fade effect
  }

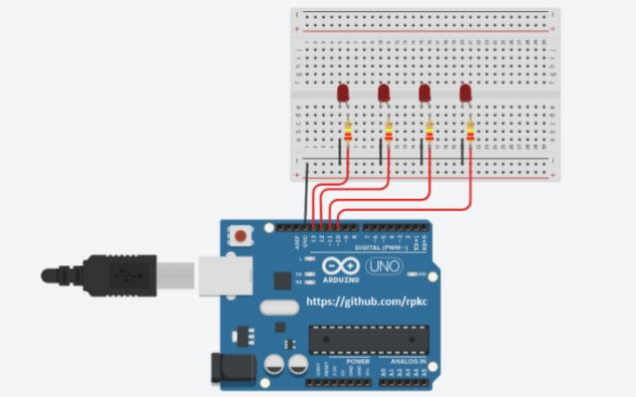
  // Fade out the LED (decrease brightness)
  for (int brightness = 255; brightness >= 0; brightness--) {

  analogWrite(ledPin, brightness);  // Write the brightness to the LED
    delay(10);  // Wait for 10ms to see the fade effect
  }
}
```

Program6. Write a program to interface LEDs on pin no. 10,11,12,13 and Blink alternatively at the delay of 1 Sec.

Components Needed:

1. Arduino Uno
2. LED
3. Resistor
4. Breadboard and jumper wires



Wiring:

Connect the LEDs:

- Connect the anode (longer leg) of each LED to the corresponding pin (10, 11, 12, 13) on the Arduino.
- Connect the cathode (shorter leg) of each LED to a 220-ohm resistor.
- Connect the other end of each resistor to ground on the Arduino.

Arduino Code:

```
void setup()

{

  pinMode(10, OUTPUT);

  pinMode(11, OUTPUT);

  pinMode(12, OUTPUT);

  pinMode(13, OUTPUT);

}

void loop()

{

 digitalWrite(10, HIGH);
```

```
digitalWrite(11, LOW);

digitalWrite(12, LOW);

digitalWrite(13, LOW);


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

digitalWrite(10, LOW);

digitalWrite(11, HIGH);

digitalWrite(12, LOW);

digitalWrite(13, LOW);

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

digitalWrite(10, LOW);

digitalWrite(11, LOW);

digitalWrite(12, HIGH);

digitalWrite(13, LOW);

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

digitalWrite(10, LOW);

digitalWrite(11, LOW);
```

```
  digitalWrite(12, LOW);

  digitalWrite(13, HIGH);

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

}
```

Program7. Write a program to interface LED and Buzzer With Arduino Board, so that buzzer is put on whenever LED is on and Buzzer is put off when LED is off.

Components Needed:

1. **Arduino board** (like Uno, Nano, etc.)
2. **LED**
3. **Resistor** (220Ω for the LED)
4. **Buzzer**
5. **Breadboard and jumper wires**

Circuit Diagram:

1. **LED**:
   - Connect the anode (longer leg) of the LED to a digital output pin (e.g., pin 9) through a 220Ω resistor.
   - Connect the cathode (shorter leg) of the LED to GND.
2. **Buzzer**:
   - Connect one pin of the buzzer to a digital output pin (e.g., pin 8).
   - Connect the other pin of the buzzer to GND.

Arduino code:-

```
// Define the pin numbers for the LED and
Buzzer

const int ledPin = 9;
const int buzzerPin
= 8;
void setup() {
// Set LED and Buzzer pins as
output
pinMode(ledPin, OUTPUT);
pinMode(buzzerPin,
OUTPUT);
}
void loop() {
// Turn on LED and Buzzer
digitalWrite(ledPin, HIGH);
digitalWrite(buzzerPin,
HIGH);
// Keep them on for 2
seconds
```

```
  delay(2000);
  // Turn off LED and Buzzer

  digitalWrite(ledPin, LOW);
  digitalWrite(buzzerPin, LOW);
  // Keep them off for 2
  seconds
  delay(2000);
}
```

Program8. Design an interface showing the street traffic signal programmed to keep red light for 30 secs and green light for 60 secs.

Components Needed:

Component	Quantity	Purpose
Arduino Uno	1	The brain of your project
Red LED	1	To show "STOP" signal
Yellow LED	1	To show "WAIT" signal (optional but common)
Green LED	1	To show "GO" signal
220Ω Resistors	3	One for each LED (to prevent burning out)
Breadboard	1	For clean and easy wiring
Jumper Wires	6-8	To connect everything
USB Cable	1	To upload code from your computer to

		Arduino

Wiring the LEDs to Arduino Uno:

LED Color	Arduino Pin	Resistor	Connect to GND?
Red	Pin 2	220Ω	Yes
Yellow	Pin 3	220Ω	Yes
Green	Pin 4	220Ω	Yes

Arduino Code:

```
int redLED = 2;
int greenLED = 3;

void setup() {
  pinMode(redLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
}

void loop() {
  // Green ON for 60s
  digitalWrite(greenLED, HIGH);
  digitalWrite(redLED, LOW);
  delay(60000);

  // Red ON for 30s
  digitalWrite(greenLED, LOW);
  digitalWrite(redLED, HIGH);
```

```
    delay(30000);
}
```

Program9. Write a program for Pedestrain Crossing System comprising 4 LED that is Red, Yellow, Green and Blue LED, 5 Resistors and push button interface with Arduino Uno. The Blue LED will represent the pedestrian light, while the Red, Yellow and Green LEDs will represent the traffic light.

Components Needed:

Component	Quantity	Purpose
Arduino Uno	1	The main controller
Red LED	1	Traffic light (Red - Stop)
Yellow LED	1	Traffic light (Yellow - Caution)
Green LED	1	Traffic light (Green - Go)
Blue LED	1	Pedestrian light (Blue - Cross)
220Ω Resistors	5	Protect LEDs from too much current
Push Button	1	To trigger pedestrian crossing
Breadboard	1	For connecting everything neatly
Jumper Wires	6-8	To make the connections
USB Cable	1	To upload the code to Arduino

Wiring the Components:

LEDs:

LED Color	Arduino Pin	Resistor	GND Pin
Red LED	Pin 2	220Ω	GND

Yellow LED	Pin 3	220Ω	GND
Green LED	Pin 4	220Ω	GND
Blue LED	Pin 5	220Ω	GND

Push Button:

Button Pin	Arduino Pin
One side	Pin 7
Other side	GND

Arduino Code:

```
// Define the pin numbers for LEDs and button
int redLED = 2;
int yellowLED = 3;
int greenLED = 4;
int blueLED = 5;
int buttonPin = 7;  // Push button pin

// Variable to store the button state
int buttonState = 0;

void setup() {
  // Set LED pins as OUTPUT
  pinMode(redLED, OUTPUT);
```

```
  pinMode(yellowLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(blueLED, OUTPUT);

  // Set button pin as INPUT
  pinMode(buttonPin, INPUT);
}

void loop() {
  // Read the push button state
  buttonState = digitalRead(buttonPin);

  // If button is pressed, change the light cycle
  if (buttonState == HIGH) {
    // Pedestrian crossing allowed (Blue LED ON)
    digitalWrite(blueLED, HIGH);   // Pedestrian light on (crossing)
    delay(3000);  // Pedestrian crossing for 3 seconds
```

```
  // Turn off the pedestrian light and start the traffic lights cycle
  digitalWrite(blueLED, LOW);

  // Red light ON (STOP) for 30 seconds
  digitalWrite(redLED, HIGH);
  digitalWrite(yellowLED, LOW);
  digitalWrite(greenLED, LOW);
  delay(30000);  // 30 seconds of red light

  // Yellow light ON (Caution) for 5 seconds
  digitalWrite(redLED, LOW);
  digitalWrite(yellowLED, HIGH);
  digitalWrite(greenLED, LOW);
  delay(5000);  // 5 seconds of yellow light

  // Green light ON (GO) for 60 seconds
  digitalWrite(redLED, LOW);
  digitalWrite(yellowLED, LOW);
```

```
  digitalWrite(greenLED, HIGH);
  delay(60000);  // 60 seconds of green light

  // Reset all lights off
  digitalWrite(redLED, LOW);
  digitalWrite(yellowLED, LOW);
  digitalWrite(greenLED, LOW);
 }
 else {
  // If the button is not pressed, no pedestrian crossing (only traffic lights)
  digitalWrite(blueLED, LOW);  // Pedestrian light off

  // Red light ON (STOP) for 30 seconds
  digitalWrite(redLED, HIGH);
  digitalWrite(yellowLED, LOW);
  digitalWrite(greenLED, LOW);
  delay(30000);
```

```
    // Yellow light ON (Caution) for 5 seconds
    digitalWrite(redLED, LOW);
    digitalWrite(yellowLED, HIGH);
    digitalWrite(greenLED, LOW);
    delay(5000);

    // Green light ON (GO) for 60 seconds
    digitalWrite(redLED, LOW);
    digitalWrite(yellowLED, LOW);
    digitalWrite(greenLED, HIGH);
    delay(60000);

    // Reset all lights off
    digitalWrite(redLED, LOW);
    digitalWrite(yellowLED, LOW);
    digitalWrite(greenLED, LOW);
  }
}
```

Program10. Write a program to interface Button, Buzzer and LED, Whenever the button is pressed the buzzer gives beep for 350ms and LED status is toggled.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **LED** (any standard 5mm LED)
3. **220-ohm Resistor** (for the LED, to limit current and protect the LED)
4. **Buzzer** (active buzzer preferred for simplicity)
5. **Pushbutton** (momentary pushbutton switch)
6. **10k-ohm Resistor** (for the pull-down resistor)
7. **Breadboard** (optional, for easy connections)
8. **Jumper Wires**

Circuit Diagram:

1. **Connect the LED**:
   - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
   - **Cathode (shorter leg)** of the LED to one end of a 220-ohm resistor.
   - **Other end of the resistor** to the Arduino GND (ground) pin.
2. **Connect the Buzzer**:
   - **Positive Terminal** of the buzzer to a digital pin on the Arduino (e.g., pin 8).
   - **Negative Terminal** of the buzzer to the Arduino GND (ground) pin.
3. **Connect the Pushbutton**:

- One terminal of the pushbutton to a digital pin on the Arduino (e.g., pin 7).

- The other terminal of the pushbutton to the Arduino GND (ground) pin.
- Connect a 10k-ohm resistor between the button pin (pin 7) and GND to act as a pull-down resistor.

Arduino code:-

```
// Define the pins for the LED, buzzer, and
button

const int ledPin = 9;
const int buzzerPin = 8;
const int buttonPin = 7;
// Variable to keep track of the
LED state
 bool ledState = true;
void setup() {

// Set the LED and buzzer pins as outputs

pinMode(ledPin, OUTPUT);

pinMode(buzzerPin, OUTPUT);


// Set the button pin as an input
pinMode(buttonPin, INPUT);
 }

 void loop() {

// Read the state of the button
```

```
  int buttonState = digitalRead(buttonPin);

  // Check if the button is pressed
  if (buttonState == HIGH) {

    // Beep the buzzer for 350 milliseconds
    digitalWrite(buzzerPin, HIGH);

    delay(350);
    digitalWrite(buzzerPin, LOW);

    // Toggle the LED state

    ledState = !ledState;
    digitalWrite(ledPin, ledState ? HIGH : LOW);
// Debounce delay to avoid multiple toggles from a single button

    press

    delay(200);
   }

  }
```

Program11. Write a program will read input from the Serial Monitor and echo it back to you.

Components Needed:

1. **Arduino Uno** board
2. **USB cable** to connect Arduino to your computer
3. **Computer** with the Arduino IDE installed
4. **Arduino IDE** (Software for coding and uploading programs)

Wiring:

There's no additional wiring needed, other than the connection between the Arduino Uno and your computer via the USB cable. The USB cable will both power the Arduino and allow it to communicate with the computer for uploading the code and interacting with the Serial Monitor.

Arduino code:

```
void setup() {
  // Start the serial communication at 9600 baud rate
  Serial.begin(9600);
}

void loop() {
  // Check if data is available to read from the Serial Monitor
  if (Serial.available() > 0) {
    // Read the incoming data as a character
    char incomingByte = Serial.read();

    // Echo the incoming character back to the Serial Monitor
    Serial.print("You typed: ");
    Serial.println(incomingByte);
  }
```

}

Program12. Write a program to interface LCD with Arduino board and display “Hello World” on it.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **USB Cable** to connect the Arduino to your computer
3. **Arduino IDE** installed on your computer for programming the Arduino

Arduino code:-

```
// Define the incoming message
const String targetMessage = "hello
world";

void setup() {
 // Initialize the serial communication at 9600
 baud
 Serial.begin(9600);
}
void loop() {
 // Check if data is available to
 read

  if (Serial.available() > 0) {
  // Read the incoming message
```

```
    String incomingMessage = Serial.readStringUntil('\n');
    // Remove any leading or trailing
    whitespace
    incomingMessage.trim();
    // Compare the incoming message with the target
    message
    if (incomingMessage == targetMessage) {
     // Print "hello world" to the Serial Monitor
     Serial.println("hello world");
    }
  }
}
```

Program13. **Write a program to interface Light Dependent Resistor (LDR) and display the values read on the Serial monitor after delay of 2 seconds each.**

Components Needed:

1. **Arduino board** (like Uno, Nano, etc.)
2. **LDR (Light Dependent Resistor)**
3. **Resistor** (10kΩ is commonly used for the voltage divider with the LDR)
4. **Breadboard and jumper wires**

Arduino code:-

```
// Define the analog pin where the LDR is
int ldrPin = A0; // Analog pin A0

void setup() {
// Start serial communication at 9600 baud
rate
Serial.begin(9600);
}

void loop() {
// Read the value from the LDR
int ldrValue =analogRead(ldrPin);

// Print the value to the Serial Monitor
Serial.print("LDR Value: ");
Serial.println(ldrValue);

// Wait for 2 seconds
delay(2000);
}
```

Program14. Write a program in Arduino to interface a temperature sensor (LM35) to detect the temperature of a room.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **Temperature Sensor**:
   - **LM35** (analog temperature sensor)

- **DHT11** or **DHT22** (digital temperature and humidity sensor)

3. **Breadboard** (optional, for easy connections)
4. **Jumper Wires**

Using LM35 (Analog Temperature Sensor)

LM35 provides an analog voltage output that is directly proportional to the temperature. The output voltage increases by 10mV per degree Celsius.

Arduino code:-

```
// Define the pin for the LM35 sensor

const int lm35Pin = A0;

void setup() {

 // Initialize serial communication at 9600

 baud

 Serial.begin(9600);

}

void loop() {

 // Read the analog value from the

 LM35

 int sensorValue =

 analogRead(lm35Pin);

 // Convert the analog value to temperature in Celsius
```

```
  float voltage = sensorValue * (5.0 / 1023.0); // Convert to voltage
  float temperatureC = voltage * 100; // Convert voltage to temperature in C
  // Print the temperature to the Serial
  Monitor
  Serial.print("Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" C");
  // Wait for 2 seconds
  delay(2000);
}
```

Program15. Write a program to interface DHT sensor with LED. The program will switch ON LED once temperature rises above 25 degrees?

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **DHT11 Temperature and Humidity Sensor** (or DHT22)
3. **LED** (any standard 5mm LED)
4. **220-ohm Resistor** (for the LED, to limit current and protect the LED)
5. **Breadboard** (optional, for easy connections)
6. **Jumper Wires**

Circuit Diagram:

1. **Connect the DHT11/DHT22**:
   - **VCC** of DHT11/DHT22 to **5V** on Arduino.
   - **GND** of DHT11/DHT22 to **GND** on Arduino.
   - **DATA** of DHT11/DHT22 to a digital pin on Arduino (e.g., pin 7).
   - **Pull-up resistor** (10k ohm) between **VCC** and **DATA** pin (optional but recommended).
2. **Connect the LED**:
   - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
   - **Cathode (shorter leg)** of the LED to one end of a 220-ohm resistor.
   - **Other end of the resistor** to the Arduino GND (ground) pin.

Arduino code:-

```
#include <DHT.h>

// Define the type of DHT sensor
#define DHTTYPE DHT11 // Change to DHT22 if you are using DHT22

DHT dht(7, DHTTYPE);   // Initialize DHT sensor on digital pin 7

const int ledPin = 9;      // Define the pin for the LED

void setup() {

// Initialize serial communication at 9600

baud


Serial.begin(9600);
```

```
// Initialize the DHT sensor
dht.begin();
// Set the LED pin as an
output
pinMode(ledPin, OUTPUT);
 }
 void loop() {
// Wait a few seconds between
measurements

 delay(2000);
// Read temperature as Celsius
float temperatureC = dht.readTemperature();
// Check if any reads failed and exit early

if (isnan(temperatureC)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
// Print the temperature to the Serial
Monitor
```

```
Serial.print("Temperature: ");
Serial.print(temperatureC);
Serial.println(" C");
// Turn the LED on if the temperature is above 25 degrees Celsius

if (temperatureC > 25) {
    digitalWrite(ledPin, HIGH);
} else {
    digitalWrite(ledPin, LOW);
  }
 }
```

Program16. Write a program to interface LCD and DHT11, displaying the value read for sensor DHT on LCD.

Components Needed:

1. **Arduino Uno** board
2. **DHT11 Temperature and Humidity Sensor** (Sensor for temperature and humidity)
3. **16x2 LCD Display** (This is a common LCD display used with Arduino)
4. **Potentiometer** (for controlling contrast of the LCD)
5. **Jumper wires**
6. **Breadboard** (optional, for easy wiring)

Wiring the Components:

1. **DHT11 Sensor Wiring:**
   - **VCC pin (DHT11)** → **5V** pin on Arduino
   - **GND pin (DHT11)** → **GND** pin on Arduino
   - **Data pin (DHT11)** → **Pin 2** on Arduino (can be changed if needed)
2. **LCD Wiring:**
   - **VSS** → **GND** on Arduino
   - **VDD** → **5V** on Arduino
   - **V0 (contrast control)** → **Wiper pin of a potentiometer** (the other two pins of the potentiometer go to 5V and GND)
   - **RS** → **Pin 12** on Arduino
   - **RW** → **GND**
   - **EN** → **Pin 11** on Arduino
   - **D4** → **Pin 5** on Arduino
   - **D5** → **Pin 4** on Arduino
   - **D6** → **Pin 3** on Arduino
   - **D7** → **Pin 6** on Arduino
   - **A (Anode)** → **5V** (optional, for LCD backlight)
   - **K (Cathode)** → **GND**

Libraries Needed:

1. **DHT Sensor Library** (for reading from the DHT11 sensor)
2. **LiquidCrystal Library** (for controlling the LCD)

Arduino Code:

```
#include <DHT.h>

#include <LiquidCrystal.h>
```

```
// Pin setup
#define DHTPIN 2       // Pin connected to the DHT11 data pin
#define DHTTYPE DHT11   // Define sensor type (DHT11)

// Initialize DHT sensor
DHT dht(DHTPIN, DHTTYPE);

// Initialize LCD (pin 12 = RS, pin 11 = EN, pins 5-2 = D4-D7)
LiquidCrystal lcd(12, 11, 5, 4, 3, 6);

void setup() {
  // Start serial communication (for debugging if needed)
  Serial.begin(9600);

  // Initialize the DHT sensor
  dht.begin();

  // Initialize the LCD and set up the screen
  lcd.begin(16, 2);  // 16 columns, 2 rows
  lcd.print("Initializing...");
  delay(2000);  // Wait for 2 seconds before starting
}
```

```
void loop() {
  // Wait a few seconds between readings
  delay(2000);  // DHT11 requires a delay of 2 seconds between readings

  // Read temperature and humidity
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature();

  // Check if any reading failed and exit the loop
  if (isnan(humidity) || isnan(temperature)) {
    lcd.clear();
    lcd.print("Failed to read sensor");
    return;
  }

  // Print the temperature and humidity to the LCD
  lcd.clear();  // Clear the previous reading
  lcd.setCursor(0, 0);  // Set cursor to top-left corner
  lcd.print("Temp: ");
  lcd.print(temperature);
  lcd.print(" C");
```

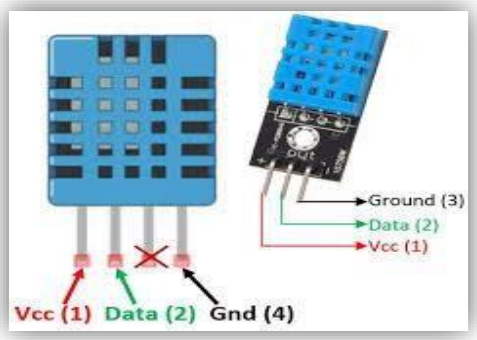
```
    lcd.setCursor(0, 1);  // Move cursor to second row
    lcd.print("Hum: ");
    lcd.print(humidity);
    lcd.print(" %");

    // Optional: print to serial monitor for debugging
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.print(" °C ");
    Serial.print("Humidity: ");
    Serial.print(humidity);
    Serial.println(" %");
}
```

Program17. Write an Arduino program to interface a temperature monitoring system in a public environment which will display information on 16*2 LCD. The display should show "Fever" if temp. is greater than 99.6 degrees and "No Fever" if the temp. is below 99.6 degrees. (Use of DHT11 or LM35 temperature sensor is permissible).

Components Needed:

- **Arduino Uno**
- **DHT11 Humidity Sensor**
- **Breadboard and jumper wires**
- **10k ohm resistor (usually needed for some sensors, but not always required for the DHT11)**



Wiring the DHT11 Sensor

1. DHT11 Pinout:

- **VCC**: Connects to 5V on the Arduino.
- **GND**: Connects to GND on the Arduino.
- **DATA**: Connects to a digital pin on the Arduino (commonly used pin is D2).
- **NC**: Not connected (usually not used).

2. Connections:

- Connect the **VCC** pin of the DHT11 to the **5V** pin on the Arduino.
- Connect the **GND** pin of the DHT11 to a **GND** pin on the Arduino.
- Connect the **DATA** pin of the DHT11 to **digital pin 2** on the Arduino (you can use another digital pin if you change the pin number in the code).

Arduino Code:

```
#include <DHT.h>

#include <LiquidCrystal.h>
```

```
// Pin configurations
#define DHTPIN 7        // Pin where DHT11 data pin is connected
#define DHTTYPE DHT11     // Define sensor type (DHT11)
#define LM35PIN A0       // If using LM35, connect to analog pin A0
DHT dht(DHTPIN, DHTTYPE);  // Initialize DHT sensor
// LCD pin setup (adjust as per your connection)
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
  // Start serial communication
  Serial.begin(9600);

  // Initialize DHT sensor
  dht.begin();
    // Set up LCD
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("Temp. Monitoring");
  delay(2000); // Wait for 2 seconds
}
void loop() {
  // Reading temperature from DHT11 sensor
  float temperature = dht.readTemperature();
```

```
// Uncomment the following lines if you use LM35 sensor instead of DHT11
// float voltage = analogRead(LM35PIN) * (5.0 / 1023.0);
// float temperature = voltage * 100;  // LM35 outputs 10mV per degree
// Check if the reading is valid
if (isnan(temperature)) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Failed to read");
  lcd.setCursor(0, 1);
  lcd.print("sensor data!");
  delay(2000);
  return;
}
  // Print temperature on serial monitor for debugging
Serial.print("Temperature: ");
Serial.print(temperature);
Serial.println(" C");
// Clear the display before showing new info
lcd.clear();
// Display temperature status on LCD
lcd.setCursor(0, 0);
if (temperature > 60.6) {
```

```
    lcd.print("Fever");
  } else {
    lcd.print("No Fever");
  }
  // Wait for 2 seconds before reading the temperature again
  delay(2000);
}
```

Program18. Write a program for Interfacing Air Quality Sensor-pollution (e.g. MQ135) to Arduino board and display dots generated by sensor on LCD.

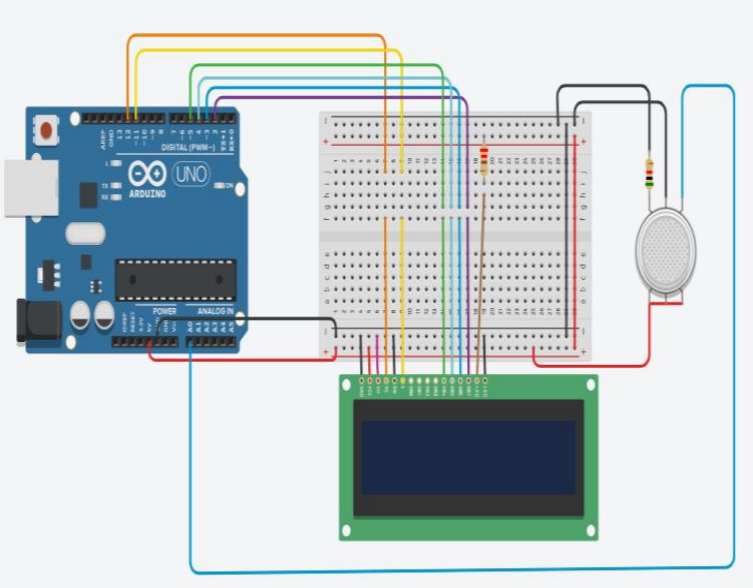
Components Needed

1. **Arduino** (e.g., Arduino Uno)
2. **MQ135 Gas Sensor**
3. **16x2 LCD with I2C Interface**
4. **Breadboard and Connecting Wires**

Wiring:-

1. **MQ135 Gas Sensor:**
   - **B1,B2,H2** to Arduino 5V
   - **A2** to Arduino A0
   - **H1** to Arduino GND
   - **A1** to Arduino GND through register
2. **LCD :**

- RS pin to digital pin 12
- Enable pin to digital pin 11
- D4 pin to digital pin 5
- D5 pin to digital pin 4
- D6 pin to digital pin 3
- D7 pin to digital pin 2
- R/W pin to ground
- VSS pin to ground
- VCC pin to 5V
- Contrast pin to ground



Arduino Code:

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins

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

int sensor = A0;

void setup() {
  // set up the LCD's number of columns and rows:

 lcd.begin(16, 2);

 // Print a message to the LCD.
```

```
  lcd.print("Air Monitoring");
  lcd.setCursor(0, 1);
  lcd.print("System");
  delay(1000);
  lcd.clear();
}
void loop() {

 int airquality = analogRead(sensor);  // Read sensor value
 int dot = map(airquality, 0, 1023, 0, 16);  //Map Sensor value to no. of dots

lcd.clear(); // clear the display before printing new data

for (int i=0;i<dot;i++) {

  lcd.print("."); //Print dots based on air quality
}
lcd.print("Air is clear");
delay(5000); // Wait 5 seconds before the next reading

}
```

Program19. Write a program to sense Obstacle closer than 1 meter near an IR sensor interfaced With Arduino Uno.

Components Needed:

1. **Arduino Uno board**
2. **IR Distance Sensor** (e.g., Sharp GP2Y0A21YK0F)
3. **Breadboard and jumper wires**

Circuit Diagram:

1. **IR Distance Sensor**:
   - Connect the **VCC** pin of the IR sensor to the **5V** pin of the Arduino.
   - Connect the **GND** pin of the IR sensor to the **GND** pin of the Arduino.
   - Connect the **OUT** pin of the IR sensor to an analog input pin on the Arduino (e.g., **A0**).

Arduino Sketch:

```
// Define the analog pin where the IR sensor is
connected

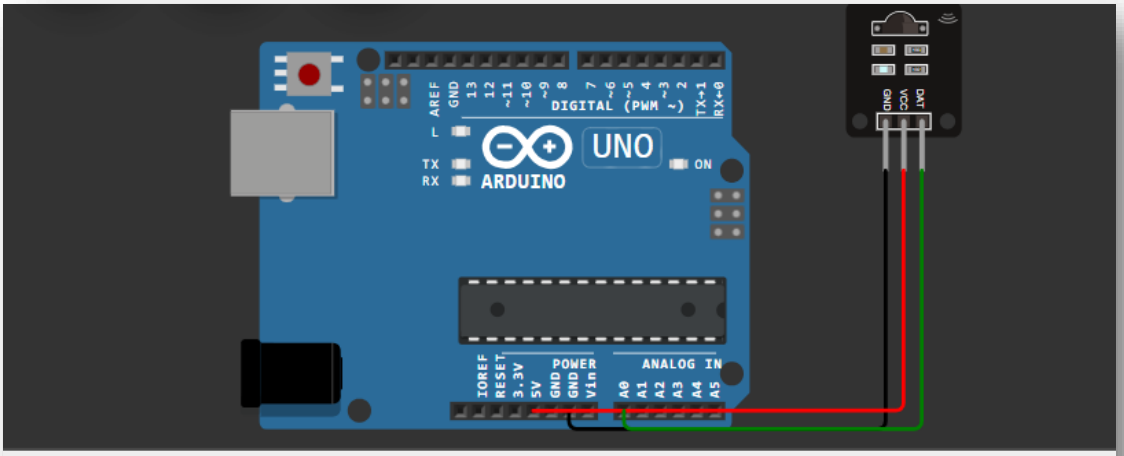
const int sensorPin = A0;
// Define the threshold distance (in centimeters) for obstacle
detection
const int thresholdDistance = 100; // 1 meter = 100 centimeters
void setup() {
// Start serial communication at 9600 baud
rate
```

```
Serial.begin(9600);
 }
 void loop() {
// Read the value from the IR sensor
int sensorValue = analogRead(sensorPin);
// Convert the sensor value to distance
  // Note: The conversion factor needs to be adjusted based on your
 specific // sensor
// For the Sharp GP2Y0A21YK0F, a typical conversion might look like
this:
// Distance (cm) = 27.86 / (sensorValue / 4.8 - 11.8)
// This is just an example and may need calibration for your sensor
float distance = 27.86 / (sensorValue / 4.8 - 11.8);
// Print the distance to the Serial
Monitor
Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");
// Check if the distance is less than the
threshold
```

```
if (distance < thresholdDistance) {
    Serial.println("Obstacle detected within 1 meter!");
  }
// Wait for 500 milliseconds before taking
another

reading

delay(500);
 }
```




```
Distance: 0.14 cm
Obstacle detected within 1 meter!
```

O-Level M1-R5.1, M2-R5.1, M3-R5.1, M4-R5.1

Full Video Playlist Available on YouTube Channel

Zed-King Computer Institute Kaithal

Free Online Computer Classes on YouTube Channel

https://www.youtube.com/@zedkinginstitute

For free PDF Notes

Our Telegram Channel:

t.me/boost/Zedking_Institute