



PROJECT ***BASIC STARTER KIT***



A unit of Quad Store

UNO R4 LITE

WWW.QUADSTORE.IN



Table of Contents

Introduction

1. Introduction to UNO R4 Board
2. Warranty – Register Now!
3. Understanding Uno R4 board
4. Installing Arduino IDE
5. Connecting the UNO R4 Board
6. Uploading Your First Program (Blinking internal LED Test)

Projects

1. External LED Blink
2. Traffic Lights with 3 LEDs
3. LED control using Push Button
4. Passive Buzzer Sound
5. Active Buzzer Beep
6. RGB LED Color Mixing
7. Potentiometer-controlled LED Brightness (PWM)
8. LDR Night Lamp
9. IR Detector
10. Servo Motor Sweep
11. DHT11 Weather Monitor
12. I2C LCD Display
13. 4x4 Keypad

Mini Projects

1. Automated Gate System & Smart Dustbin
2. Digital Locker
3. Smart Weather Station
4. GAME: Stickman Jumping
5. GAME: Car Dodging

Introduction to Uno R4 boards (Must Read)

About the UNO R4 Lite/Minima Board

The **UNO R4** board is an upgraded version of the previous UNO R3. There are **2 versions** of the Uno R4 board available which is **Minima** and **Wifi**.

What is difference between UNO R4 Lite & Uno R4 minima?

The board included in Quad Store kit is a compatible Uno R4 Lite which is exact same as the Minima board. Our boards has the exact same processor, larger memory, and uses a USB Type-C connector and expandable I/O ports.

Despite improvements, it is still easy to use for beginners.

So, in short, **Quad Store's Uno R4 Lite = Arduino Uno R4 Minima**.

What is difference between compatible and original board?

- **Arduino boards are open-source**, so anyone can legally manufacture their own version that works exactly like the original.
- A **compatible board** is a board built to work exactly like the original Arduino, following the same pin layout and functionality.
- Compatible boards (like Quad Store's) offer the same functionality but with upgraded PCB quality and durability.
- They often include **extra expansion** options such as additional I/O pads or improved connectors.
- Compatible boards usually provide better value, with longer warranties, rust-proof coating, and stronger local support

Why Quad Store Uno R4 Lite board is better than original Arduino Uno R4 Minima board?

- **Superior PCB** construction for improved signal integrity and durability.
- Additional plated-through holes and **expansion pads** for extra analogue/digital/IO connections — ideal for complex DIY builds.
- Protective **rust-proof coating** (conformal finish) to safeguard against humidity and corrosion (especially useful in Indian environments).
- Standard 3-month **warranty** included; register the product on our website and get an additional 3 months – giving you a full **6-month** peace-of-mind.
- **Enhanced customer support**: we provide quicker responses, dedicated documentation, and local spare-parts availability — beyond the generic support offered by many brands

Warranty: (Register Now!)

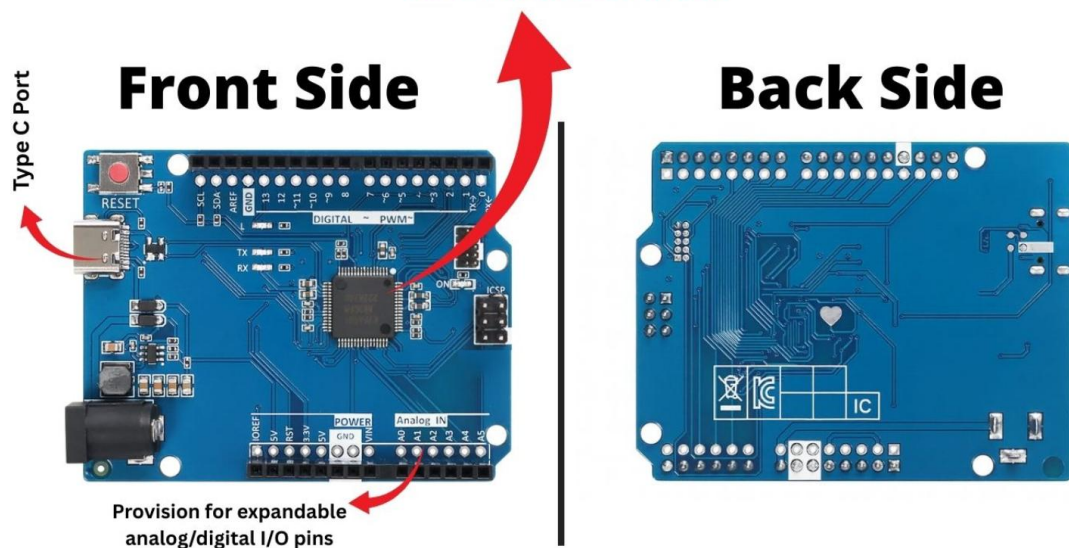
To **claim standard warranty** for Uno R4 Lite boards, **registration** on the Quad Store website is required. We provide a standard **3-month** warranty, and by registering your product online, you receive an additional 3 months—giving you a full **6 months** of peace of mind.

Visit below link and register within **7 days** of purchase of product. **You may also choose to complete the warranty registration after verifying that Uno R4 board is functioning properly by running your 1st program which is Blinking LED Project which is covered in subsequent session.**

<https://quadstore.in/warranty/>

Quad Store R4-Lite board

Comes with Renesas RA4M1 series microcontroller

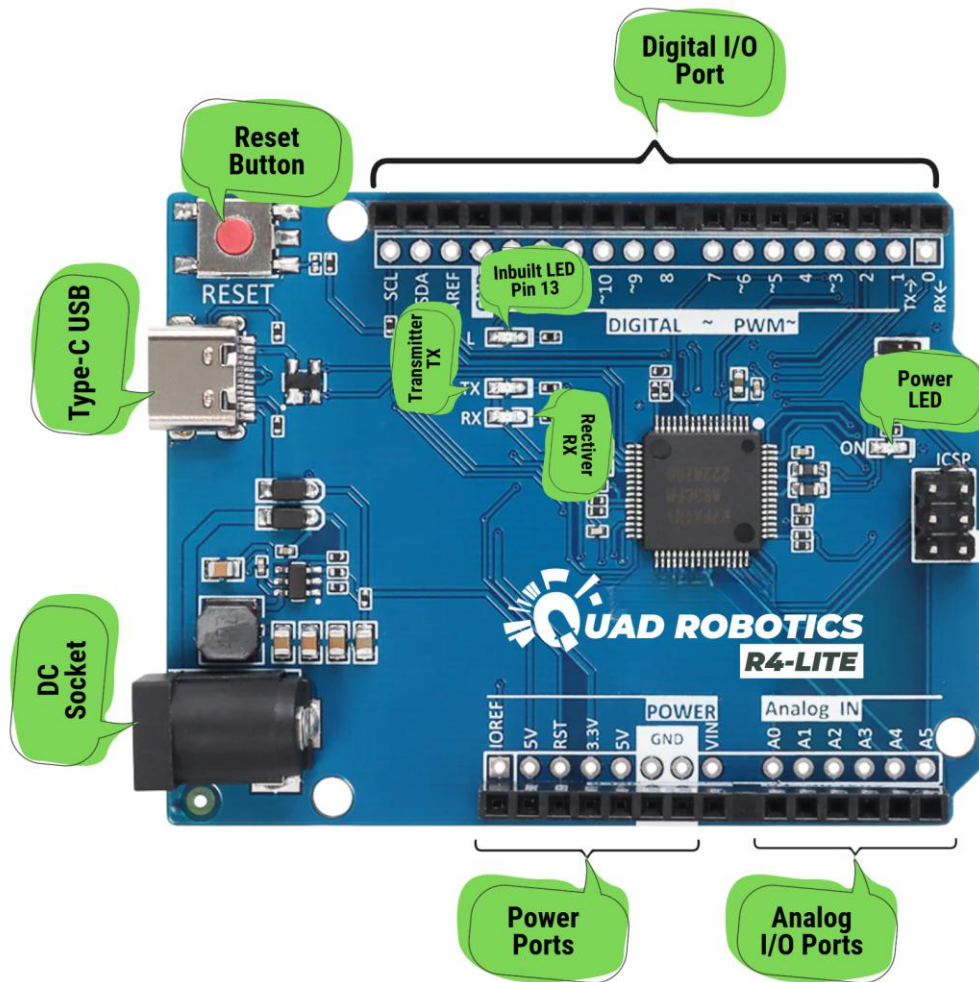


Our boards are fully compatible with Arduino IDE

- Model: Renesas RA4M1 (RA4M1F44LBA)
- Core: Arm® Cortex® -M4, 32-bit
- Clock speed: 48 MHz
- Flash memory: 256 KB
- SRAM: 32 KB
- Operating voltage: 5V

Understanding Uno R4 Board:

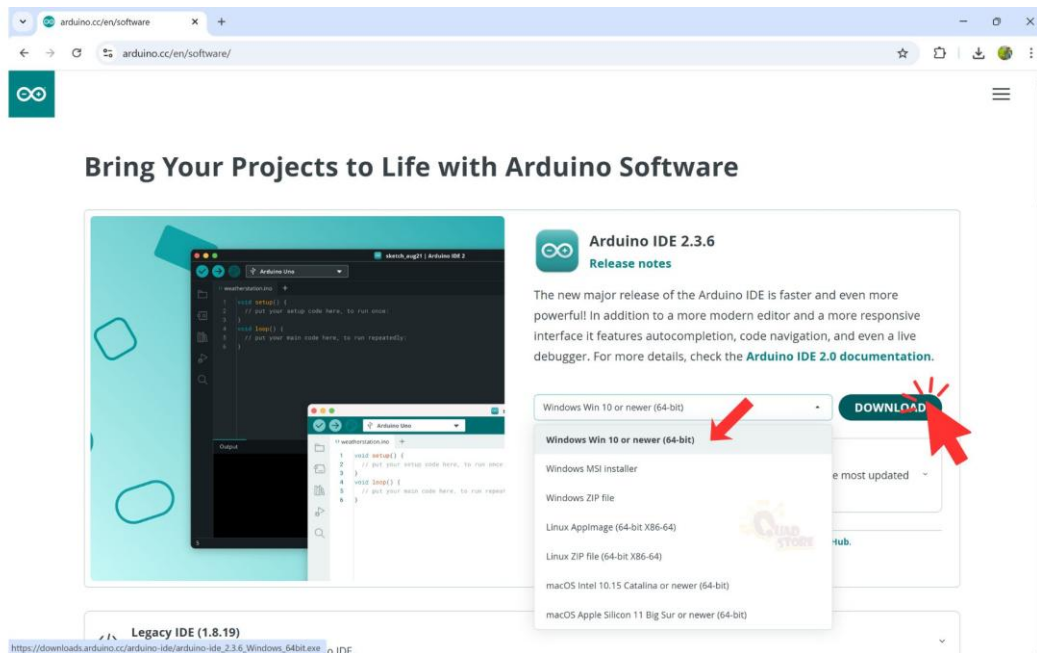
The reference board used in this book is **Quad Store's UNO R4 Lite board**, an Arduino-compatible Uno R4 minima model. A diagram of the Quad Store UNO board is shown below. The board included in your kit **may or may not** carry the **Quad Store / Quad Robotics logo**, as it depends on availability. However, all boards function identically.



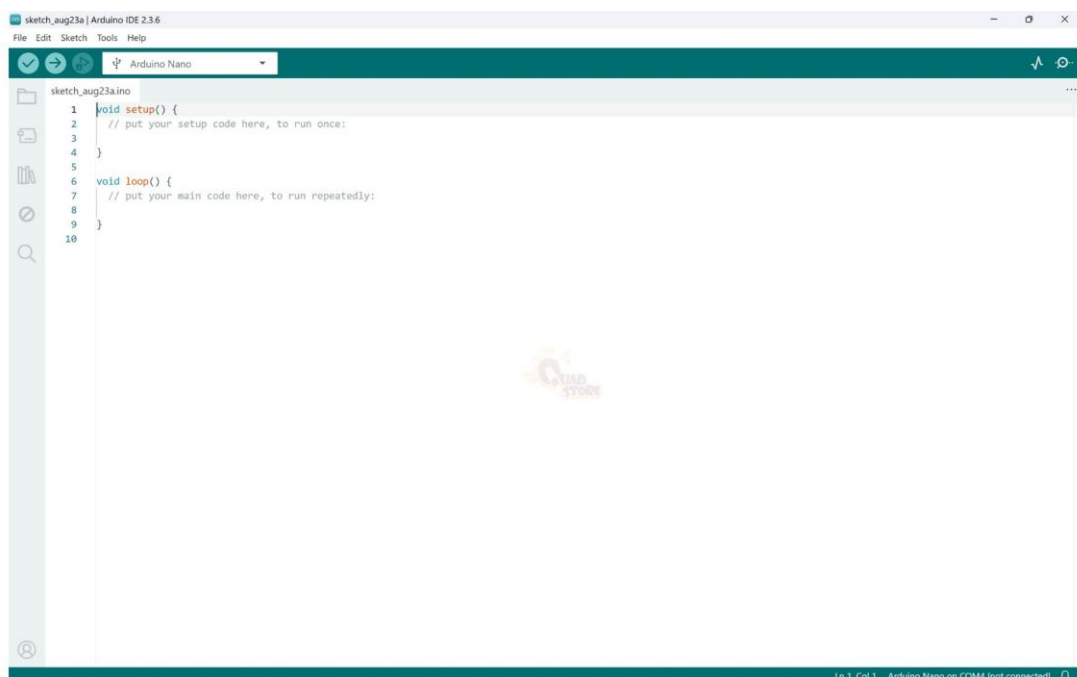
- **Type-C USB:** Used to power the board and upload programs from your computer.
- **Reset Button:** Restarts the board and reloads the running program.
- **DC Socket:** Allows powering the board using an external 7–12V adapter.
- **Digital I/O Port:** Pins used for digital input/output operations, including PWM.
- **Inbuilt LED Pin 13:** Built-in test LED connected to digital pin 13 for quick debugging.
- **Transmitter (TX):** Sends serial data from the board to other devices.
- **Receiver (RX):** Receives serial data into the board from other devices.
- **Power LED:** Indicates that the board is powered ON.
- **Power Ports:** Provide various voltage outputs (5V, 3.3V, GND, VIN) for sensors and modules.
- **Analog I/O Ports:** Pins A0–A5 used for reading analog sensor values.

Installing Arduino IDE

1. Visit the official Arduino website: <https://www.arduino.cc/en/software>
2. Download and install the Arduino IDE for Windows, Mac, or Linux.



3. Open the IDE by clicking on the desktop icon after installation. IDE should open as shown.

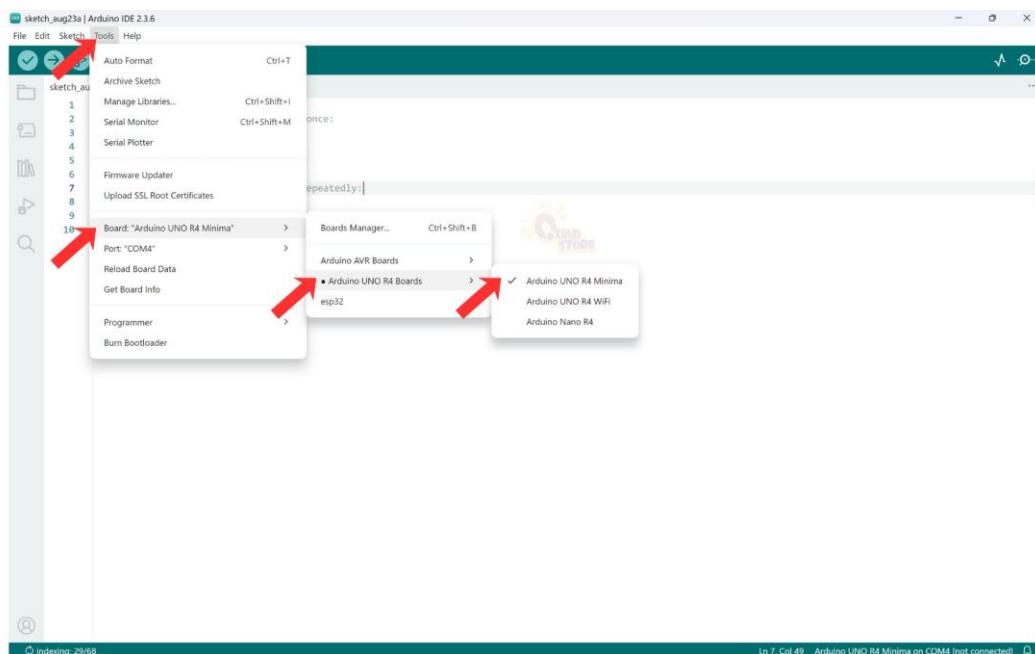


Connecting the UNO R4 Board

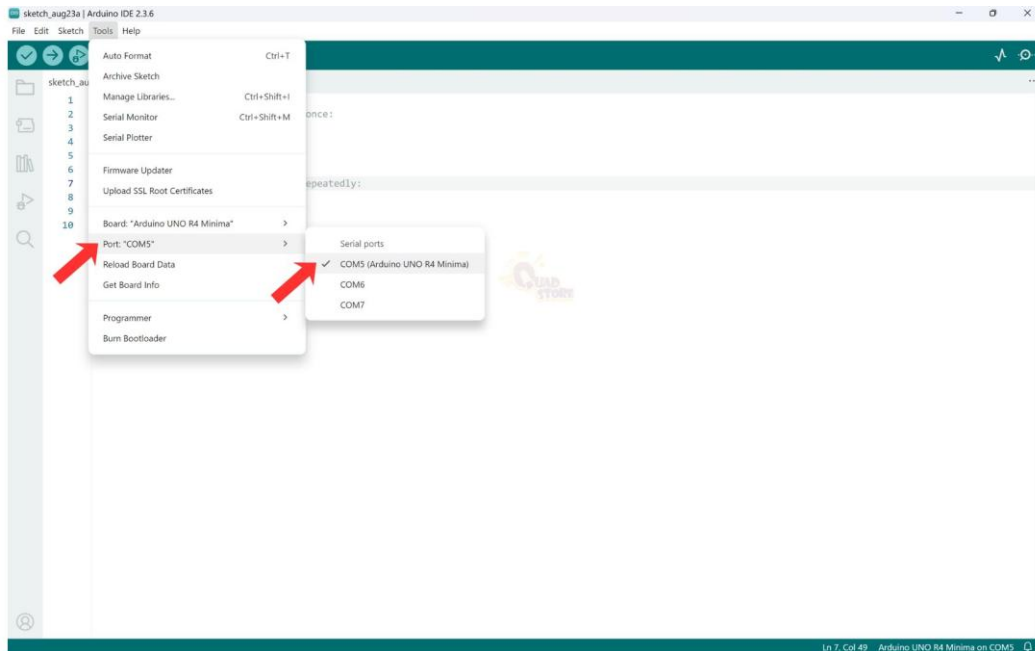
1. Use the Type-C to USB-A cable provided in the kit.
2. Plug the Type-C end into the UNO R4 and the USB-A end into your computer. If your computer or laptop has only a Type-C port, please purchase a Type-C to Type-A USB adapter and use it, or you may use a Type-C to Type-C USB cable instead.



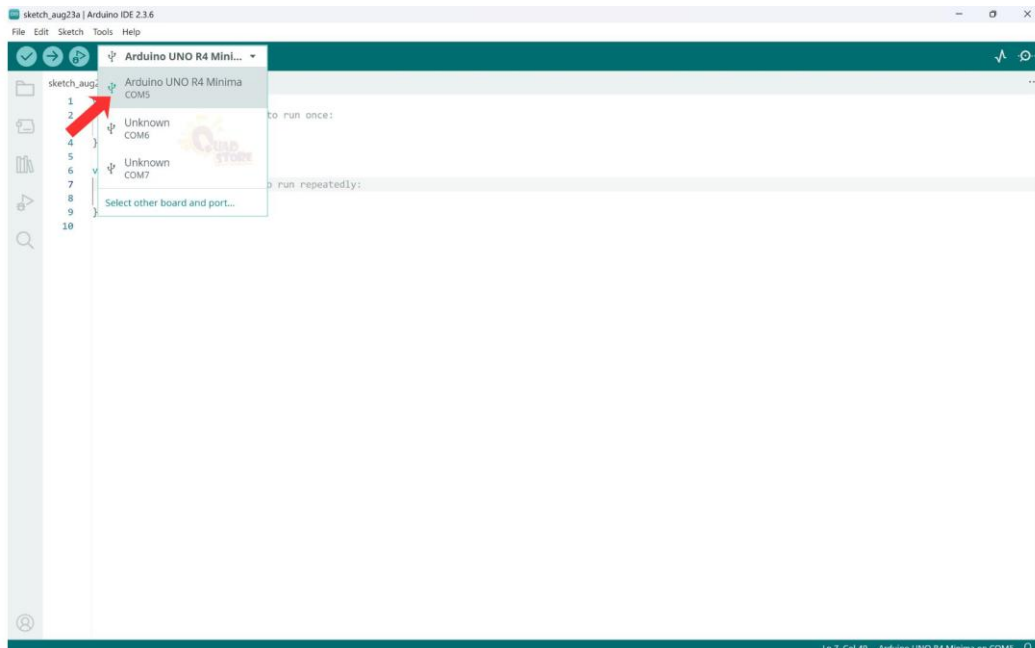
3. The power LED on the board will turn ON.
4. Go to **Tools** → **Board** → **Arduino Uno R4 Board** → Select “**Arduino Uno R4 Minima**” or “**Arduino Uno R4 Wifi**” based on your board. Here you need to select “**Arduino Uno R4 Minima**”.



5. Make sure the correct port is selected by going to **Tools → Port → COM** (Arduino Uno R4 boards) based on your COM port. **NOTE:** COM port number will change based on your computer/laptop.



6. Now you need to select the board. Click the drop-down menu under “Select Board” option. Select the corresponding board. Ensure the correct board is getting reflected in the IDE screen.





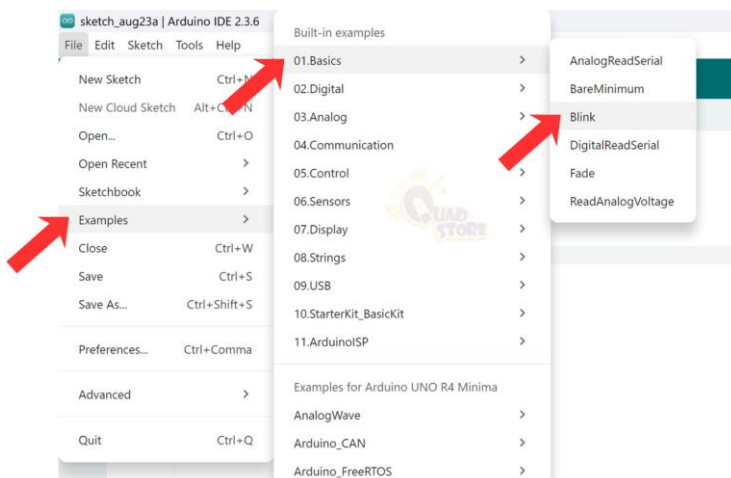
Uploading Your First Program (Blink Test)

This program makes the built-in LED blink and ensures the board and IDE are working correctly.

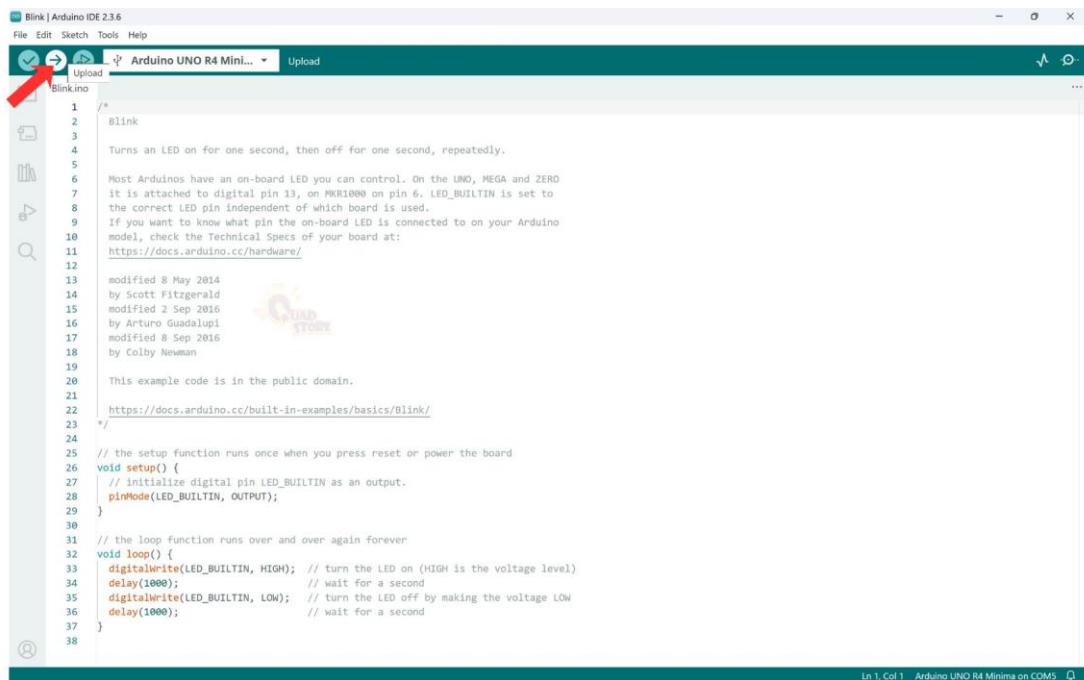
```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // Turn LED on
  delay(1000);                     // Wait 1 second
  digitalWrite(LED_BUILTIN, LOW);  // Turn LED off
  delay(1000);                     // Wait 1 second
}
```

Steps to Upload Code

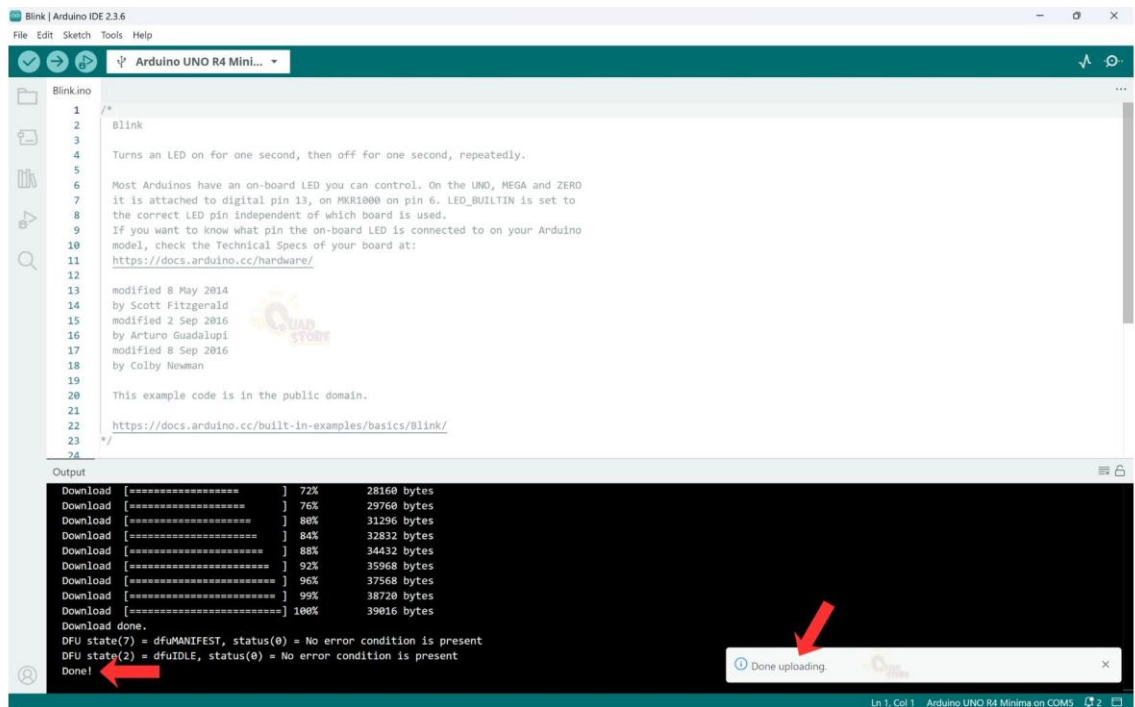
1. Either **copy & paste the above code** into the Arduino IDE **(OR)** open **File → Examples → 0.1 Basics → Blink** program.



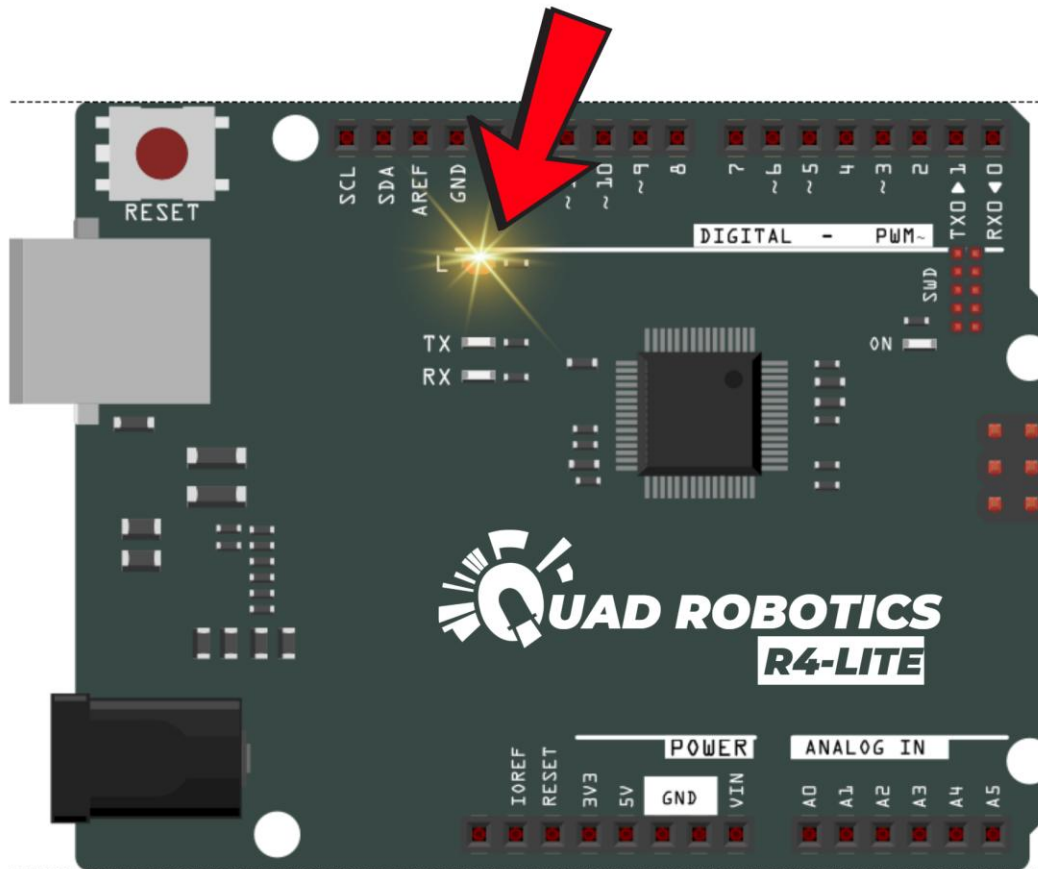
2. Click on the **Upload button** (right arrow icon).



3. Wait for the code to **compile and upload**. You should see a message “Done!” at the bottom of the screen. The board will start running the program automatically.



Output: You should see the built-in LED under PIN 13 starts to blink at a regular interval.



Project 1: Blinking External LED

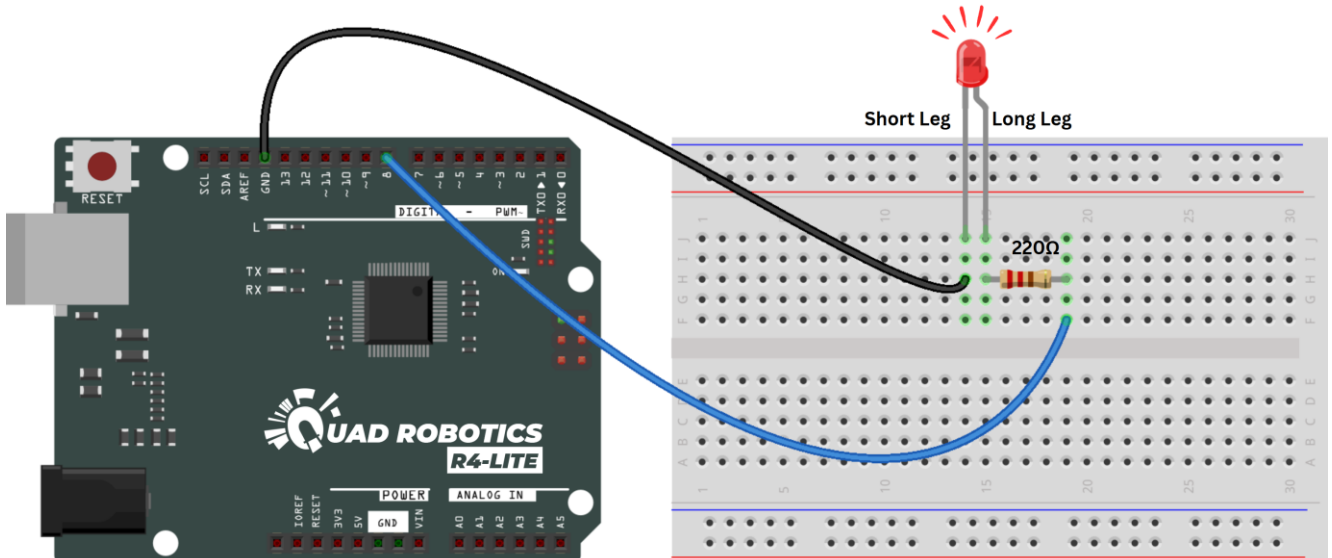
🌟 Objective: Blink an external LED connected on a breadboard.

📦 Components Required

- ✓ UNO R4 board
- ✓ Breadboard
- ✓ 1 × LED (any color)
- ✓ 1 × 220Ω Resistor
- ✓ Male to Male Jumper wires
- ✓ Type C to A USB cable

💡 Circuit Diagram

Component	Connection Method	Uno R4
LED (Anode, Long Leg +)	Through 220Ω resistor	Pin 8
LED (Cathode, Short Leg -)	Direct Connection	GND



💻 Code

```
int led = 8;

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

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


Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste the above code** into the Arduino IDE OR open the file **1.LEDBlink.ino** from the provided **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically

Output

The program turns the LED ON (HIGH) and OFF (LOW) every second.

DIY Extension

Try different colors of LEDs.

Add more LEDs on pins 9 and 10 for patterns like traffic lights.

Project 2: Traffic Lights with 3 LEDs

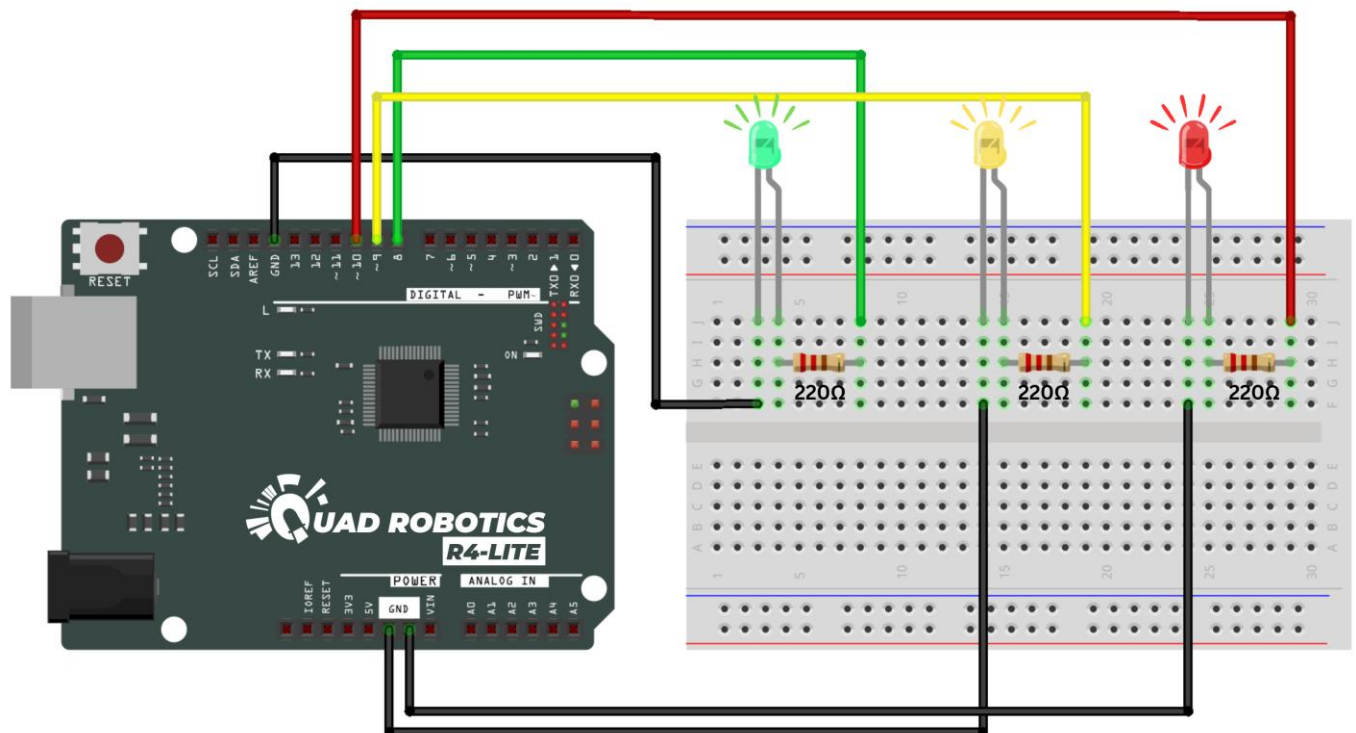
🌟 Objective: Simulate a simple traffic light system using 3 LEDs.

📦 Components Required

- ✓ UNO R4 board
- ✓ Breadboard
- ✓ 3 × LEDs (Red, Yellow, Green)
- ✓ 3 × 220Ω Resistors
- ✓ Male to Male Jumper wires
- ✓ Type C to A USB cable

💡 Circuit Diagram

Component	Connection Method	Uno R4
Green LED (Anode, Long Leg +)	Through 220Ω resistor	Pin 8
Green LED (Cathode, Short Leg -)	Direct Connection	GND
Yellow LED (Anode, Long Leg +)	Through 220Ω resistor	Pin 9
Yellow LED (Cathode, Short Leg -)	Direct Connection	GND
Red LED (Anode, Long Leg +)	Through 220Ω resistor	Pin 10
Red LED (Cathode, Short Leg -)	Direct Connection	GND



Code

```
int red = 10;
int yellow = 9;
int green = 8;

void setup() {
  pinMode(red, OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop() {
  digitalWrite(red, HIGH);
  delay(3000);
  digitalWrite(red, LOW);

  digitalWrite(yellow, HIGH);
  delay(1000);
  digitalWrite(yellow, LOW);

  digitalWrite(green, HIGH);
  delay(3000);
  digitalWrite(green, LOW);
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste above code** into the Arduino IDE OR open the file **2.TrafficLights.ino** from the **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically.

Output

Red stays ON for 3 seconds, Yellow for 1 second, and Green for 3 seconds.

DIY Extension

Add a pedestrian button to change the lights when pressed.

Use a buzzer for sound alerts.

Project 3: LED control using Push Button

🌟 Objective: To control an **LED** using a **push button** connected to an **UNO R4**.

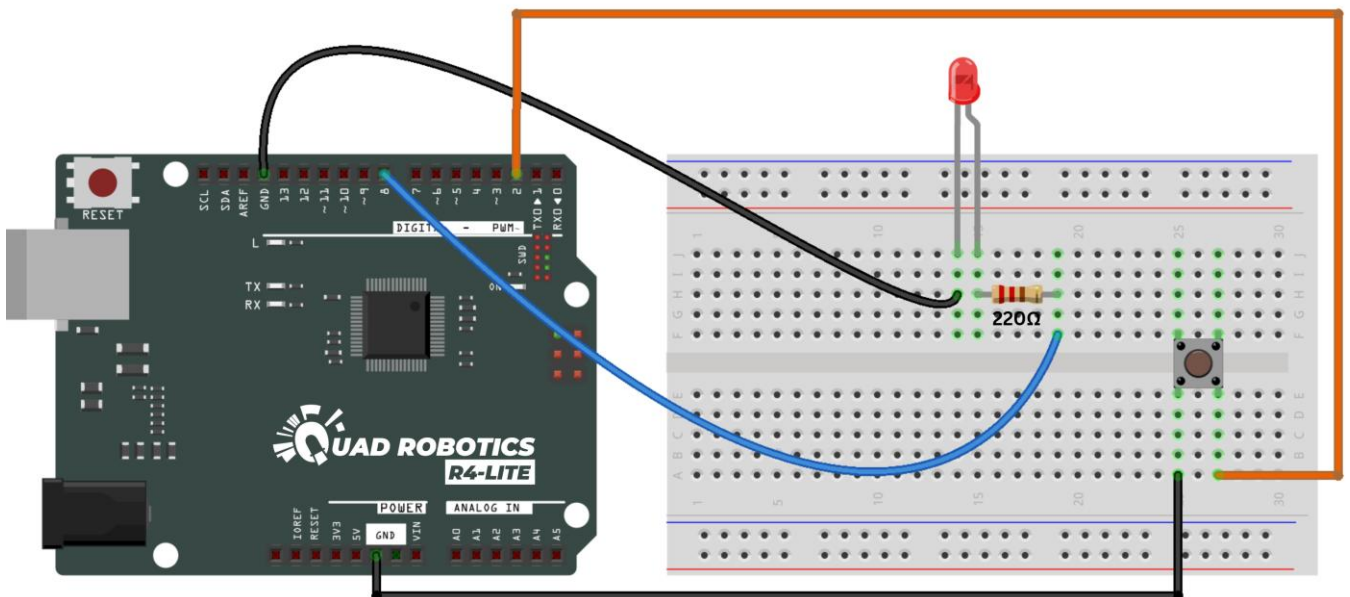
- When button is pressed → LED turns ON
- When button is released → LED turns OFF

🛒 Components Required

- ✓ UNO R4 board
- ✓ Breadboard
- ✓ 1 × LED (Any color of your choice)
- ✓ 1 × 220Ω Resistors
- ✓ 1 × Push Button Switch
- ✓ Male to Male Jumper wires

💡 Circuit Diagram

Component	Connection Method	Uno R4
LED (Anode, Long Leg +)	Through 220Ω resistor	Pin 6
LED (Cathode, Short Leg -)	Direct Connection	GND
Push Button	One side	Pin 2
Push Button	Other side	GND



Code

```
const int buttonPin = 2;
const int ledPin = 8;

void setup() {
  pinMode(buttonPin, INPUT_PULLUP); // internal pull-up
  pinMode(ledPin, OUTPUT);
}

void loop() {
  int buttonState = digitalRead(buttonPin);

  if (buttonState == LOW) {
    // button pressed
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste above code** into the Arduino IDE OR open the file **3.PushButtonLED.ino** from the provided **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
6. Click on the **Upload button** (right arrow icon).
7. Wait for the code to **compile and upload**. The board will start running the program automatically.

Output

- LED stays OFF initially
- Press the button → LED turns ON
- Release the button → LED turns OFF.

DIY Extension

Toggle LED mode

Press button once → LED ON

Press again → LED OFF

Project 4: Passive Buzzer Sound

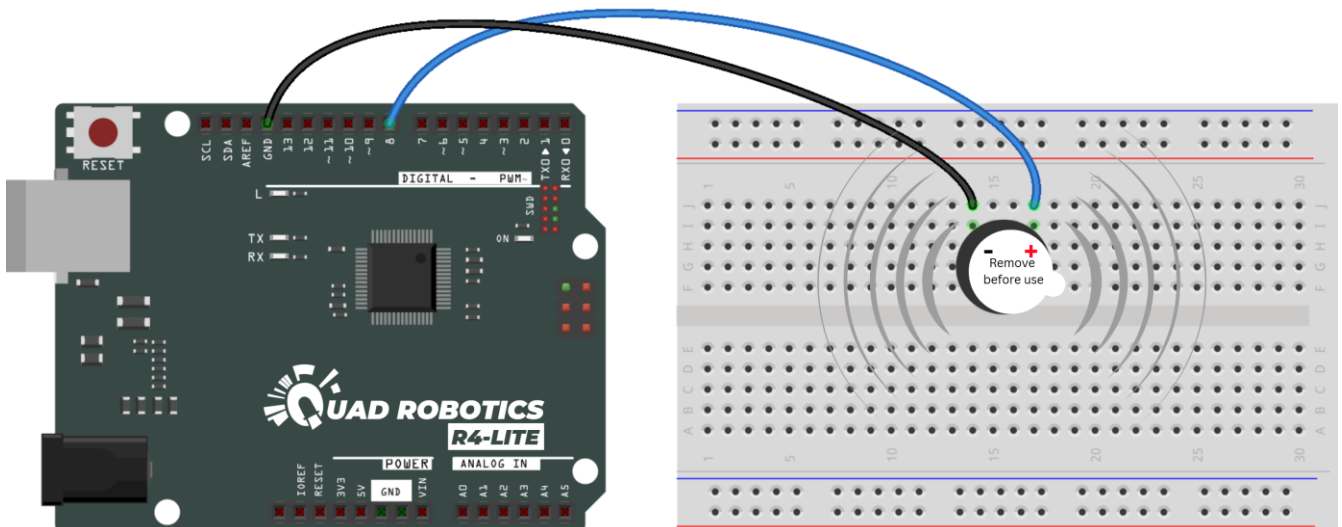
🌟 Objective: Generate sound using a passive buzzer with tone function.

🧰 Components Required

- ✓ UNO R4 board
- ✓ Passive Buzzer (remove the top sticker on the buzzer before use)
- ✓ Male to Male Jumper wires
- ✓ Breadboard

🔌 Circuit Diagram

Component	Connection Method	Uno R4
Buzzer (Positive +)	Direct Connection	Pin 8
Buzzer (Negative -)	Direct Connection	GND



💻 Code

```
int buzzer = 8;

void setup() {
}

void loop() {
  tone(buzzer, 1000); // 1kHz tone
  delay(1000);
  noTone(buzzer);
  delay(1000);
}
```


Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **copy and paste above code** into the Arduino IDE OR open the file **4.PassiveBuzzer.ino** from the provided **CODE folder**.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as Arduino Uno R4 minima from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

The passive buzzer produces sound when driven with different frequencies.

Here we generate a 1kHz tone for 1 second ON and 1 second OFF.

DIY Extension

Change frequency for different musical notes.

Try making a melody.

Project 5: Active Buzzer Beep

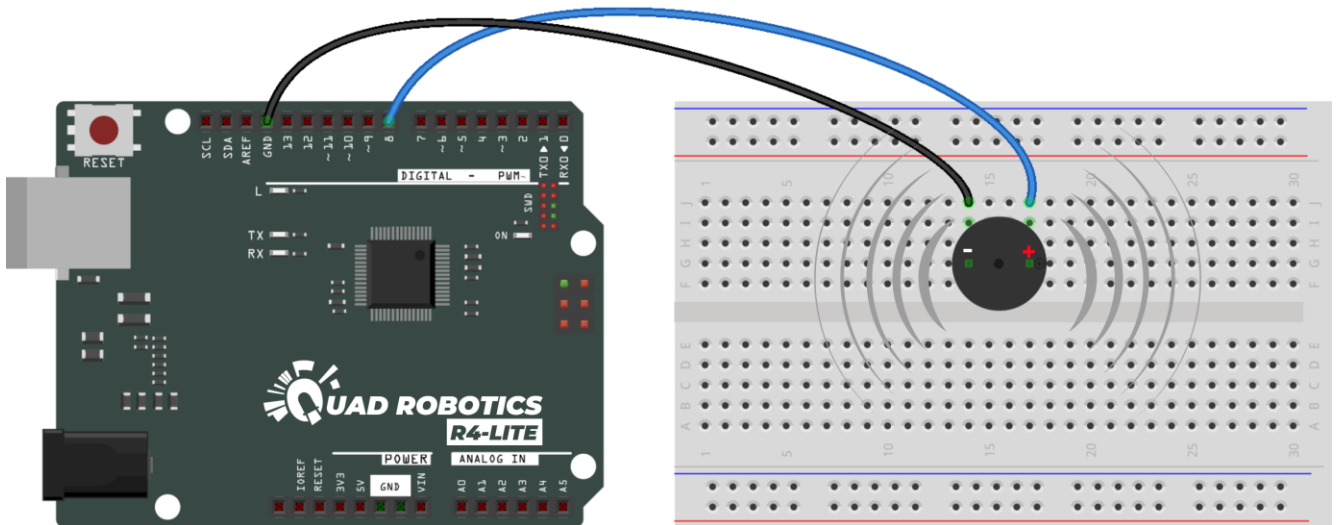
🌟 Objective: Turn an active buzzer ON and OFF.

📦 Components Required

- ✓ UNO R4 board
- ✓ Active Buzzer
- ✓ Male to Mae Jumper wires
- ✓ Breadboard

🔌 Circuit Diagram

Component	Connection Method	Uno R4
Buzzer (Positive +)	Direct Connection	Pin 8
Buzzer (Negative -)	Direct Connection	GND



💻 Code

```
int buzzer = 8;

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

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


Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **5.ActiveBuzzer.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

An active buzzer produces a fixed beep when powered.

We simply switch it ON and OFF every second.

DIY Extension

Use buzzer for alarms in sensor projects.

Project 6: RGB LED Color Mixing

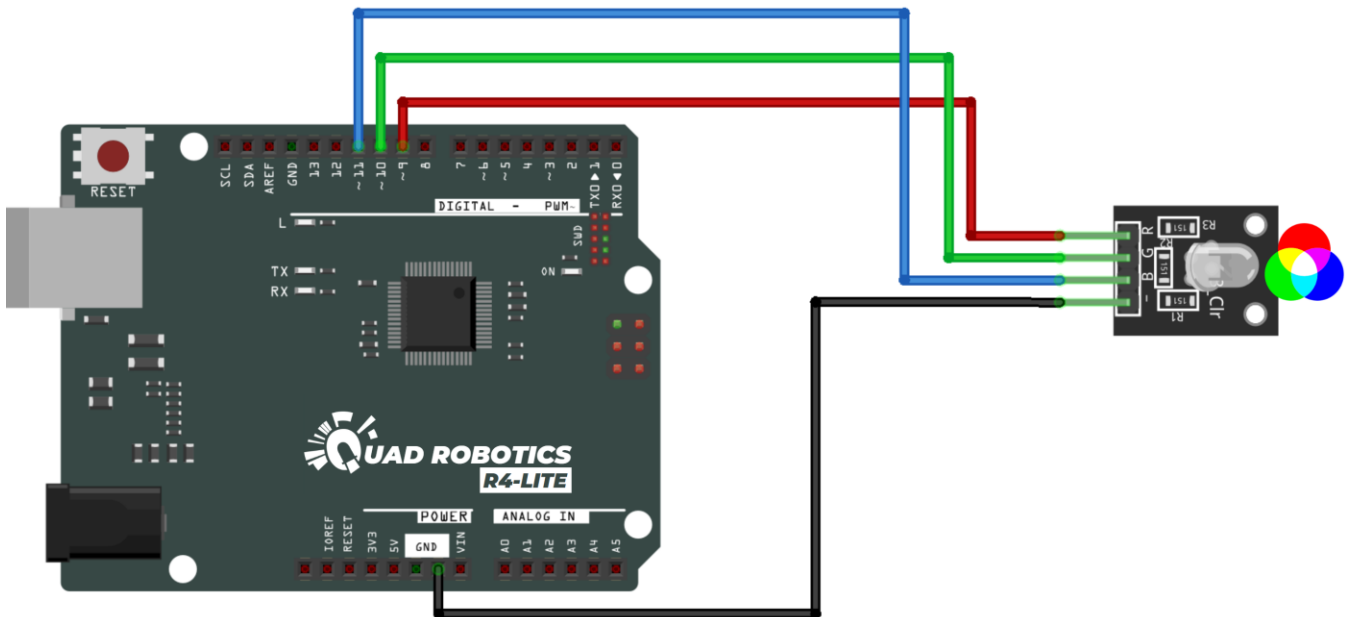
🌟 Objective: Control an RGB LED to show different colors.

📦 Components Required

- ✓ UNO R4 board
- ✓ RGB LED Module
- ✓ Male to Female Jumper wires. A breadboard is not required when using these wires, as you can directly connect the RGB module to the female end.

🔌 Circuit Diagram

Component	Connection Method	Uno R4
RGB Module – R (Red)	Direct Connection	Pin 9
RGB Module – G (Green)	Direct Connection	Pin 10
RGB Module – B (Blue)	Direct Connection	Pin 11
RGB Module – GND	Direct Connection	GND



Code

```
int red = 9;
int green = 10;
int blue = 11;

void setup() {
  pinMode(red, OUTPUT);
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
}

void loop() {
  digitalWrite(red, HIGH);
  digitalWrite(green, LOW);
  digitalWrite(blue, LOW);
  delay(1000);

  digitalWrite(red, LOW);
  digitalWrite(green, HIGH);
  delay(1000);

  digitalWrite(green, LOW);
  digitalWrite(blue, HIGH);
  delay(1000);
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **6.RGBLed.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

An RGB LED combines red, green, and blue light to produce colors. We switch between red, green, and blue every second.

DIY Extension

Mix multiple colors by turning ON two pins together.
Use analogWrite for smooth fading.

Project 7: Potentiometer-controlled LED Brightness

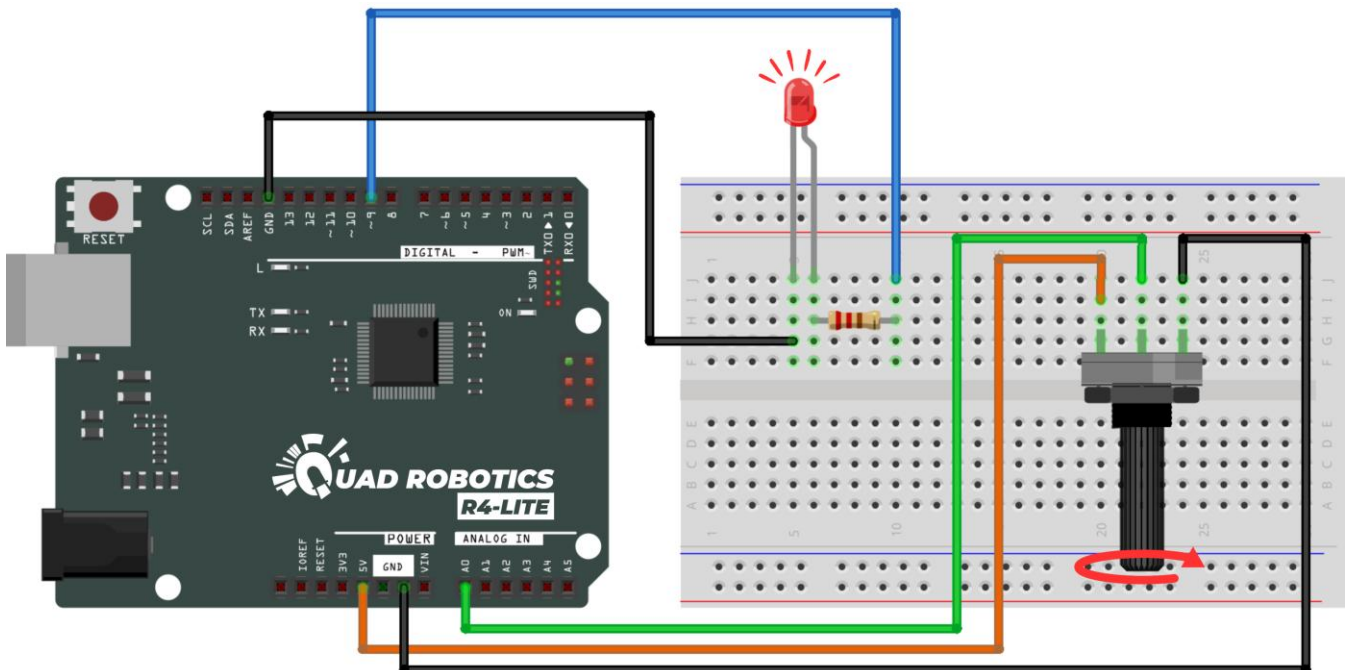
✦ Objective: Use a potentiometer to adjust LED brightness (PWM).

🧰 Components Required

- ✓ UNO R4 board
- ✓ 10K Potentiometer
- ✓ LED
- ✓ 220Ω Resistor
- ✓ Breadboard
- ✓ Male to Male Jumper wires

💡 Circuit Diagram

Component	Connection Method	Uno R4
Potentiometer (Middle Pin / Wiper)	Direct Connection	A0
Potentiometer (Left Pin)	Direct Connection	5V
Potentiometer (Right Pin)	Direct Connection	GND
LED (Anode, Long Leg +)	Through 220Ω resistor	Pin 9
LED (Cathode, Short Leg -)	Direct Connection	GND



Code

```
int potPin = A0;
int led = 9;
int val = 0;

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

void loop() {
  val = analogRead(potPin);
  val = map(val, 0, 1023, 0, 255);
  analogWrite(led, val);
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **7.PotLED.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

Potentiometer provides analog input from 0–1023.

We map it to 0–255 to control LED brightness with PWM.

Output: Turn the potentiometer knob from left to right to see the LED brightness change from low to high.

DIY Extension

Try controlling buzzer pitch with potentiometer.

Project 8: Photoresistor/ Light Dependent Resistor (LDR) or Automatic Night Lamp

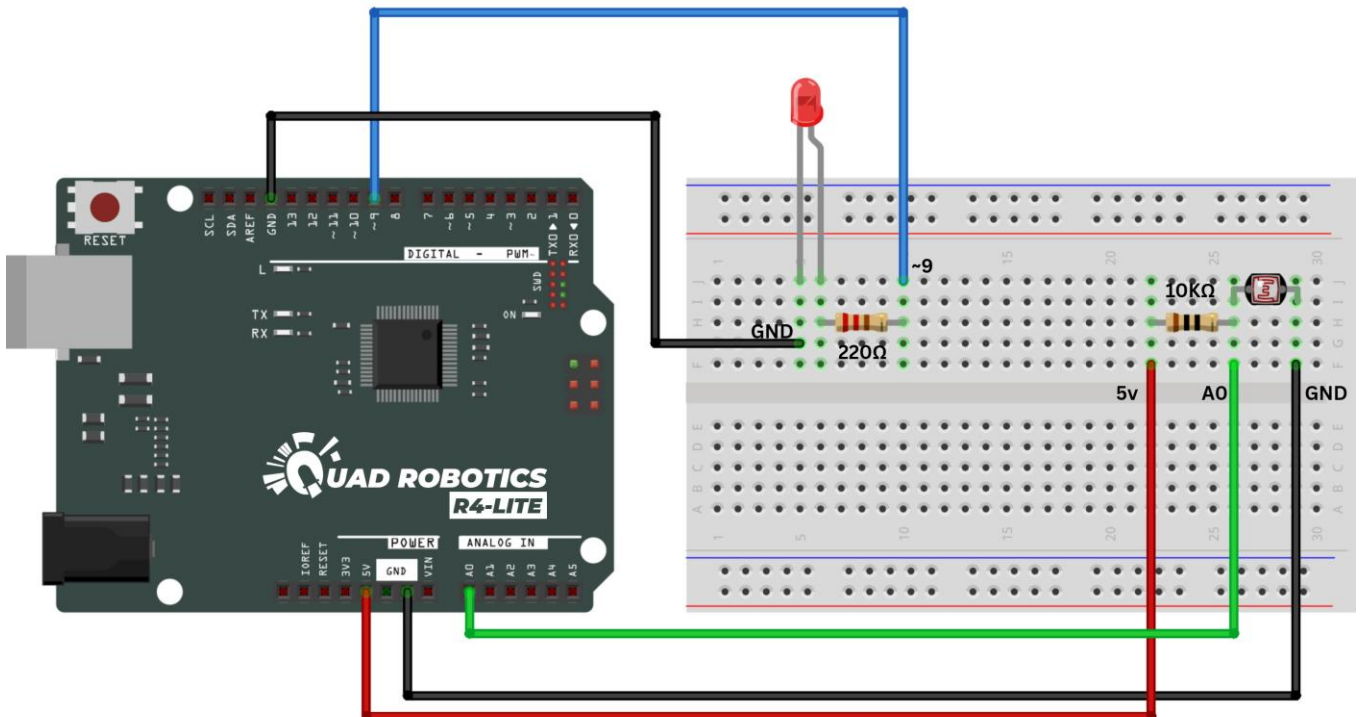
✦ Objective: Turn ON an LED in darkness using LDR sensor.

🧰 Components Required

- ✓ UNO R4 board
- ✓ Photoresistor / LDR
- ✓ LED
- ✓ 220Ω Resistor
- ✓ Breadboard
- ✓ Jumper wires

💡 Circuit Diagram

Component	Connection Method	Uno R4
LDR (One Leg)	Direct Connection	GND
LDR (Other Leg)	Through 10 kΩ resistor	5V
LDR + Resistor Junction (Middle Point)	Direct Connection	A0
LED (Anode, Long Leg +)	Through 220 Ω resistor	Pin 9
LED (Cathode, Short Leg -)	Direct Connection	GND



Code

```
int ldrPin = A0;
int ledPin = 9;

void setup() {
  pinMode(ledPin, OUTPUT);
}
void loop() {
  int lightValue = analogRead(ldrPin);
  if (lightValue < 500) {    // Bright
    digitalWrite(ledPin, LOW);
  } else {                  // Dark
    digitalWrite(ledPin, HIGH);
  }
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **8.LDRlamp.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

The LDR detects light level.

LED turns ON when it's dark (low value) or close the LDR with your finger to turn ON the LED.

DIY Extension

Adjust threshold value for sensitivity.

Use for automatic room lights.

Project 9: Infrared Obstacle Detector

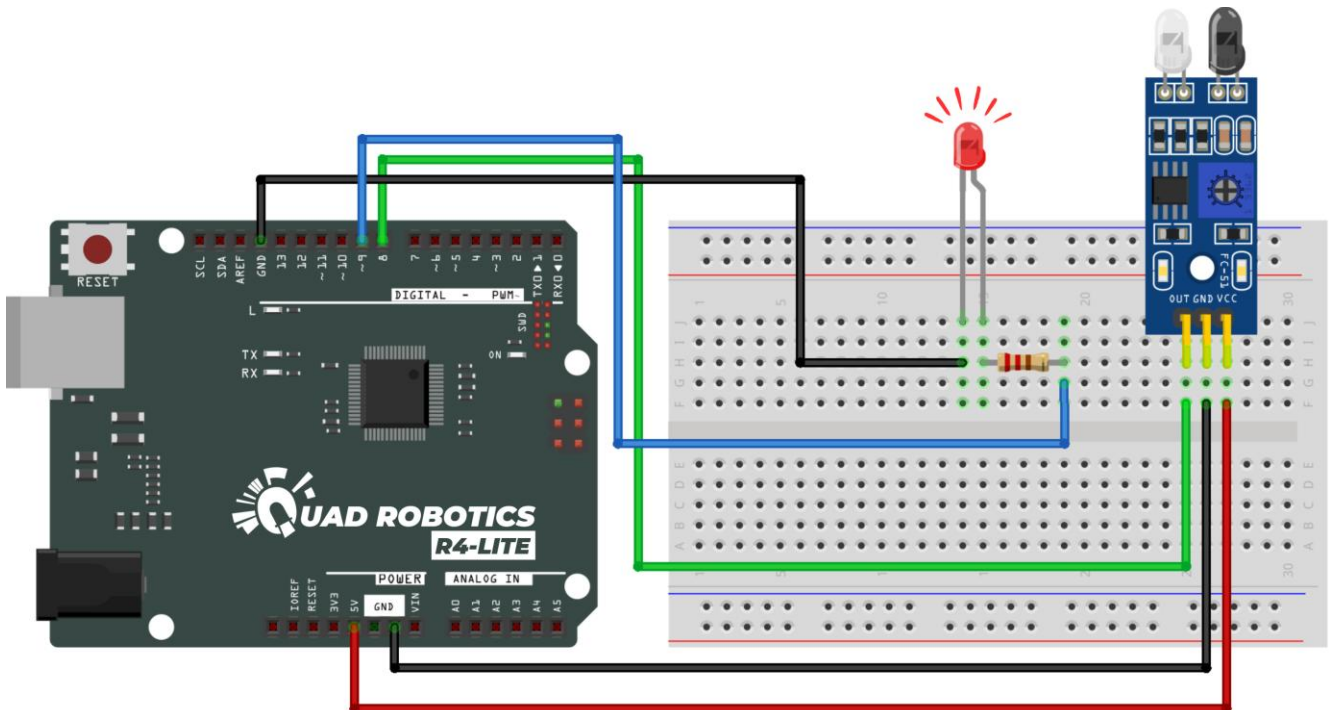
🌟 Objective: Detect objects using IR sensor.

📦 Components Required

- ✓ UNO R4 board
- ✓ Infrared IR Sensor
- ✓ Buzzer or LED
- ✓ Breadboard
- ✓ Jumper wires

🔌 Circuit Diagram

Component	Connection Method	Uno R4
IR SensorSensor OUT/D0	Direct Connection	Pin 8
IR SensorSensor VCC	Direct Connection	5V
IR SensorSensor GND	Direct Connection	GND
LED (Positive / Long Leg +)	Through 220Ω resistor	Pin 9
LED (Negative / Short Leg -)	Direct Connection	GND



Code

```
int ir = 8;
int led = 9;

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

void loop() {
  if(digitalRead(ir) == LOW){
    digitalWrite(led, HIGH);
  } else {
    digitalWrite(led, LOW);
  }
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **9. IRDetector.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

IR sensor detects reflected infrared light from objects.

When obstacle is detected, it turns on the LED.

DIY Extension

Use in anti-theft alarms.

Project 10: Servo Motor Sweep

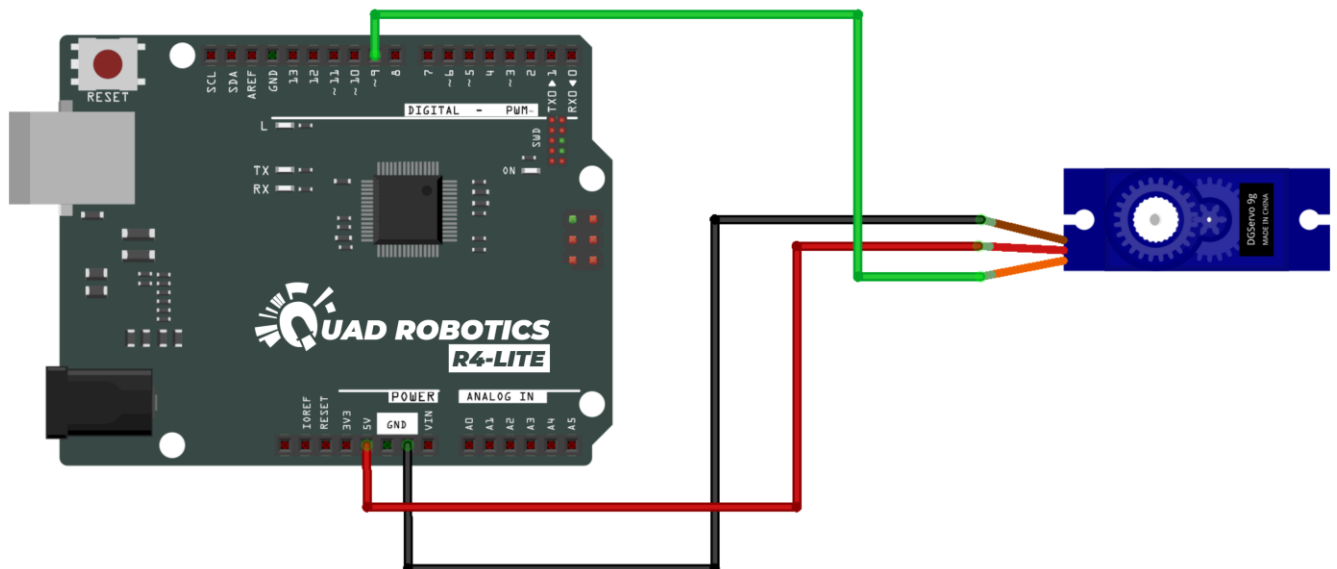
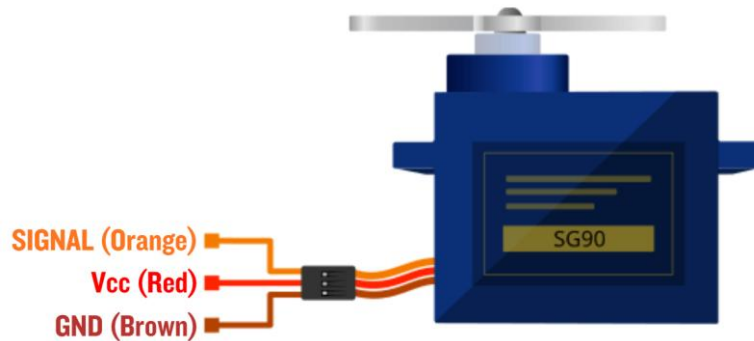
🌟 Objective: Move SG90 servo back and forth.

📦 Components Required

- ✓ UNO R4 board
- ✓ SG90 Servo
- ✓ Jumper wires. Use male to male jumper wires and insert into servo motor socket pins.

🔌 Circuit Diagram

Component	Connection Method	Uno R4
Servo Signal (Orange wire)	Direct Connection	Pin 9
Servo VCC (Red wire)	Direct Connection	5V
Servo GND (Brown wire)	Direct Connection	GND



Code

```
#include <Servo.h>

Servo myservo;

void setup() {
  myservo.attach(9);
}

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

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **10.ServoSweep.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Explanation & Output

Servo is controlled by PWM signals.

We sweep from 0° to 180° and back.

DIY Extension

Use for robot arms.

Project 11: DHT11 Weather Monitor

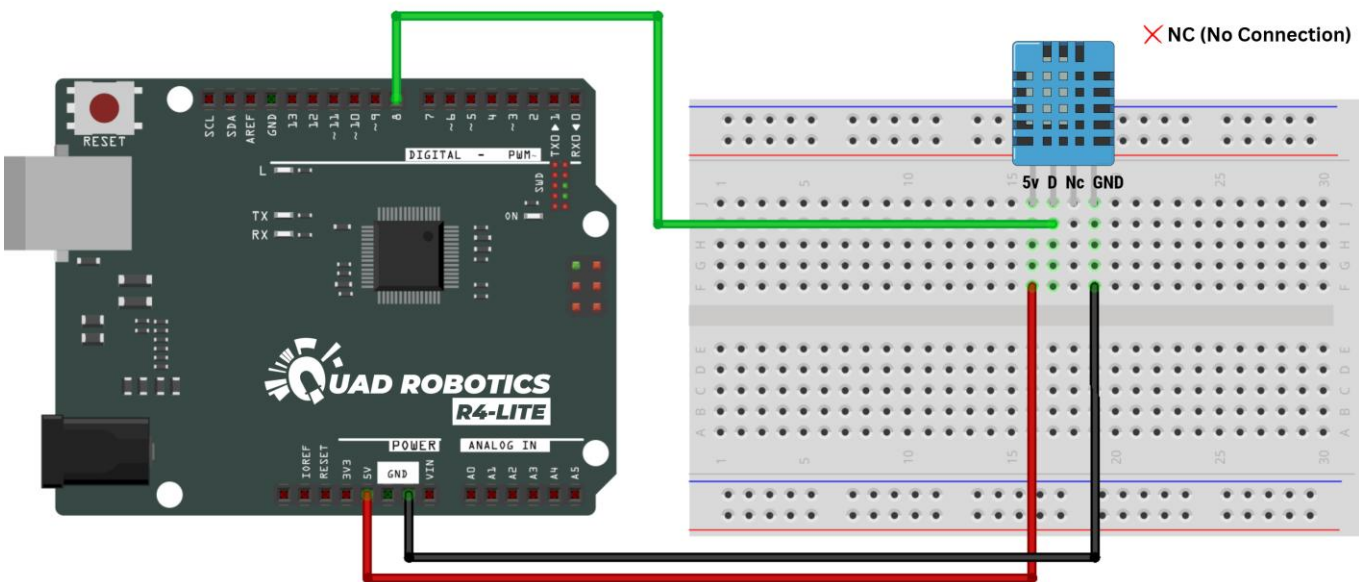
🌟 Objective: Read temperature and humidity using DHT11 sensor.

📦 Components Required

- ✓ UNO R4 board
- ✓ DHT11 Module
- ✓ Breadboard
- ✓ Jumper wires

🔌 Circuit Diagram

Component	Connection Method	Uno R4
DHT11 – VCC	Direct Connection	5V
DHT11 – Data	Direct Connection	Pin 8
DHT11 – NC	No Connection	—
DHT11 – GND	Direct Connection	GND



💻 Code

```
#include <DHT.h>
#define DHTPIN 8
#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

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

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();
```



```

Serial.print("Humidity: ");
Serial.print(h);
Serial.print(" % ");
Serial.print("Temperature: ");
Serial.print(t);
Serial.println(" *C");
delay(2000);
}

```

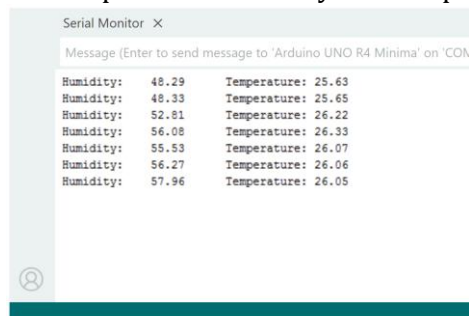
Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **11.DHT11Monitor.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.



Explanation & Output

DHT11 provides humidity and temperature readings. Values are printed on Serial Monitor.



 **DIY Extension:** Make your own weather station.

Project 12: I2C LCD Display Basics

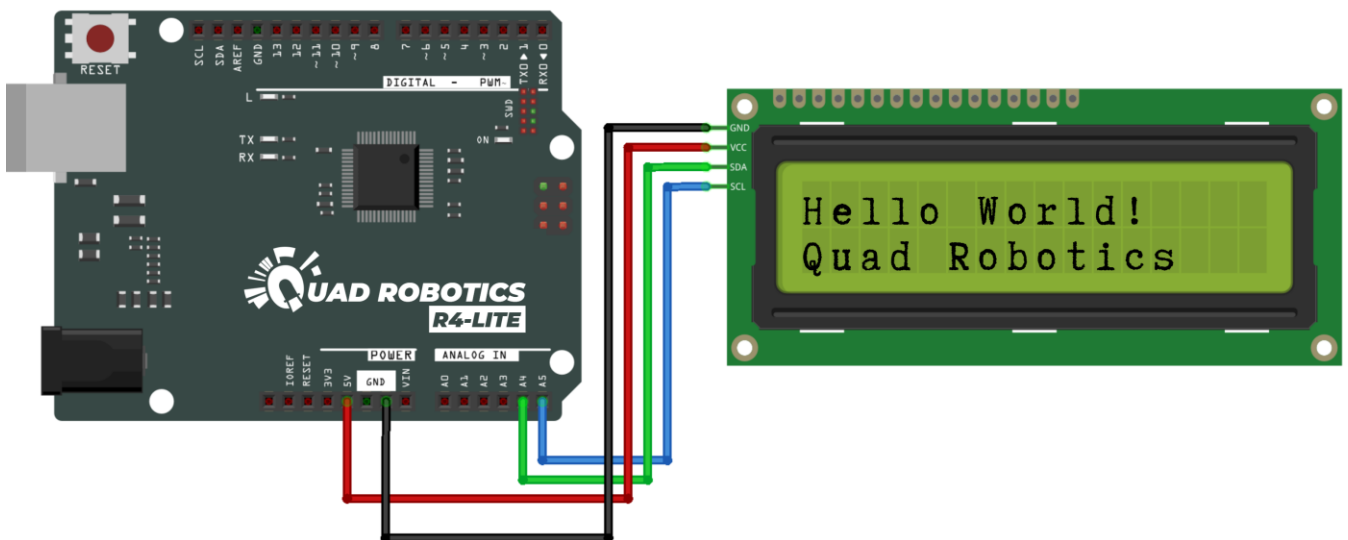
🌟 Objective: Display text on I2C LCD display.

📦 Components Required

- ✓ UNO R4
- ✓ I2C LCD Display
- ✓ Male to Female Jumper wires

💡 Circuit Diagram

Component	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	5V
I2C LCD – GND	Direct Connection	GND
I2C LCD – SDA	Direct Connection	A4
I2C LCD – SCL	Direct Connection	A5



✅ Required Library

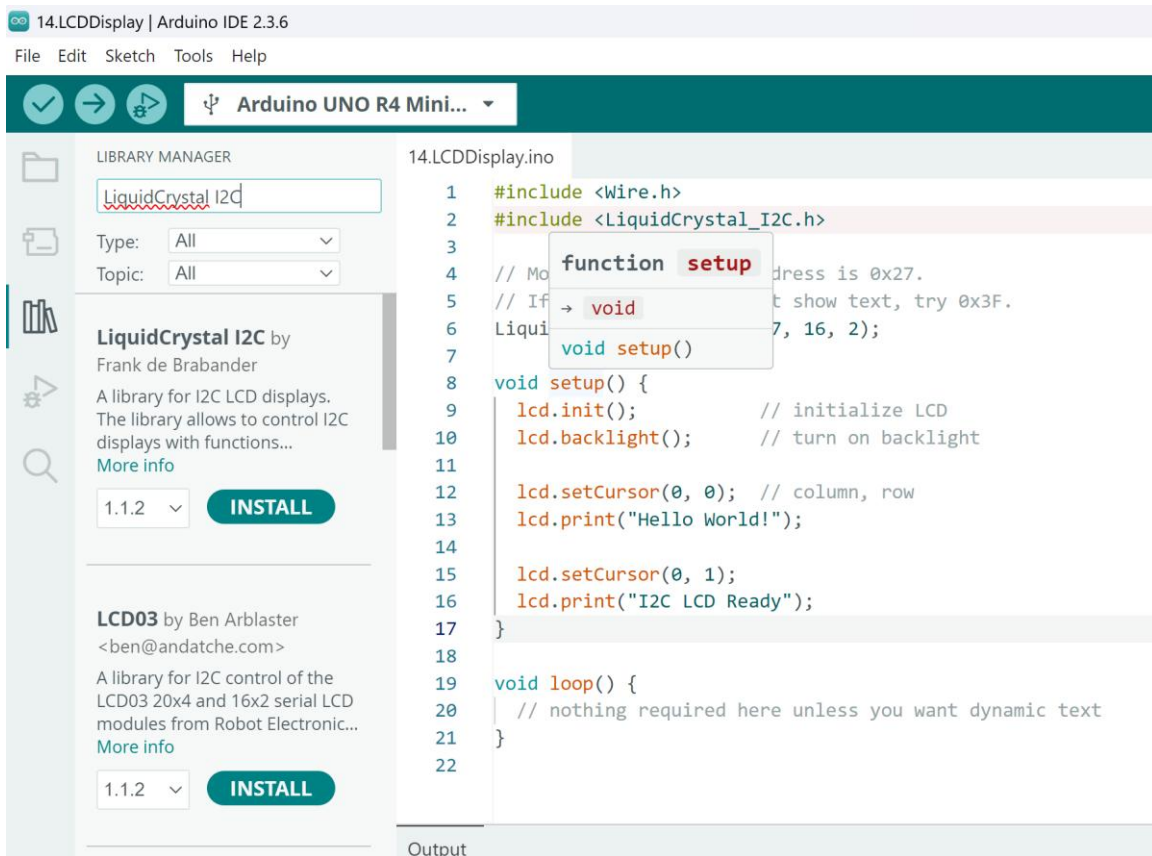
Install this library in Arduino IDE:

LiquidCrystal_I2C

(Author: Frank de Brabander)

Steps:

Sketch → Include Library → Manage Libraries → search "**LiquidCrystal I2C**" → Install



Code

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

// Most common I2C LCD address is 0x27.
// If your display doesn't show text, try 0x3F.
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
    lcd.init();           // initialize LCD
    lcd.backlight();       // turn on backlight

    lcd.setCursor(0, 0);   // column, row
    lcd.print("Hello World!");

    lcd.setCursor(0, 1);
    lcd.print("Quad Robotics");
}

void loop() {
    // nothing required here unless you want dynamic text
}
```


Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **12.LCDDisplay.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

NOTE:

If the display does not show any text or only squares then change the address of the I2C in the code to 0x3F

DIY Extension

Explanation & Output

The LED uses I2C communication (SDA, SCL).

Output: We display simple text “Hello World” on the first line of screen and “Quad Robotics” text on the second line.

DIY Extension

Show sensor values on LED.

Make a mini weather station display.

Project 13: 4x4 Keypad Input Display

🌟 **Objective:** Read key presses from a 4x4 keypad and display them on Serial Monitor.

🧰 Components Required

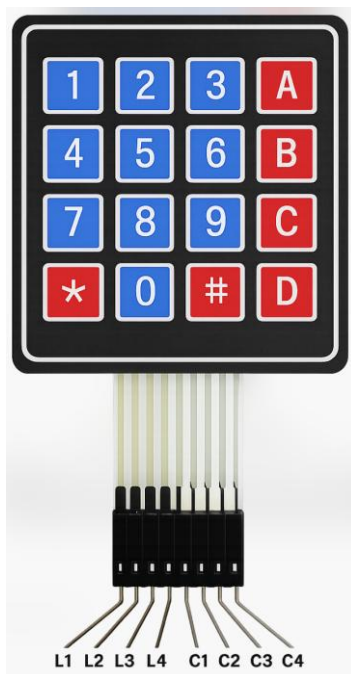
- ✓ UNO R4 board
- ✓ 4x4 Keypad Module
- ✓ Jumper wires (Male to Female)
- ✓ USB cable for Arduino

💡 Circuit Diagram

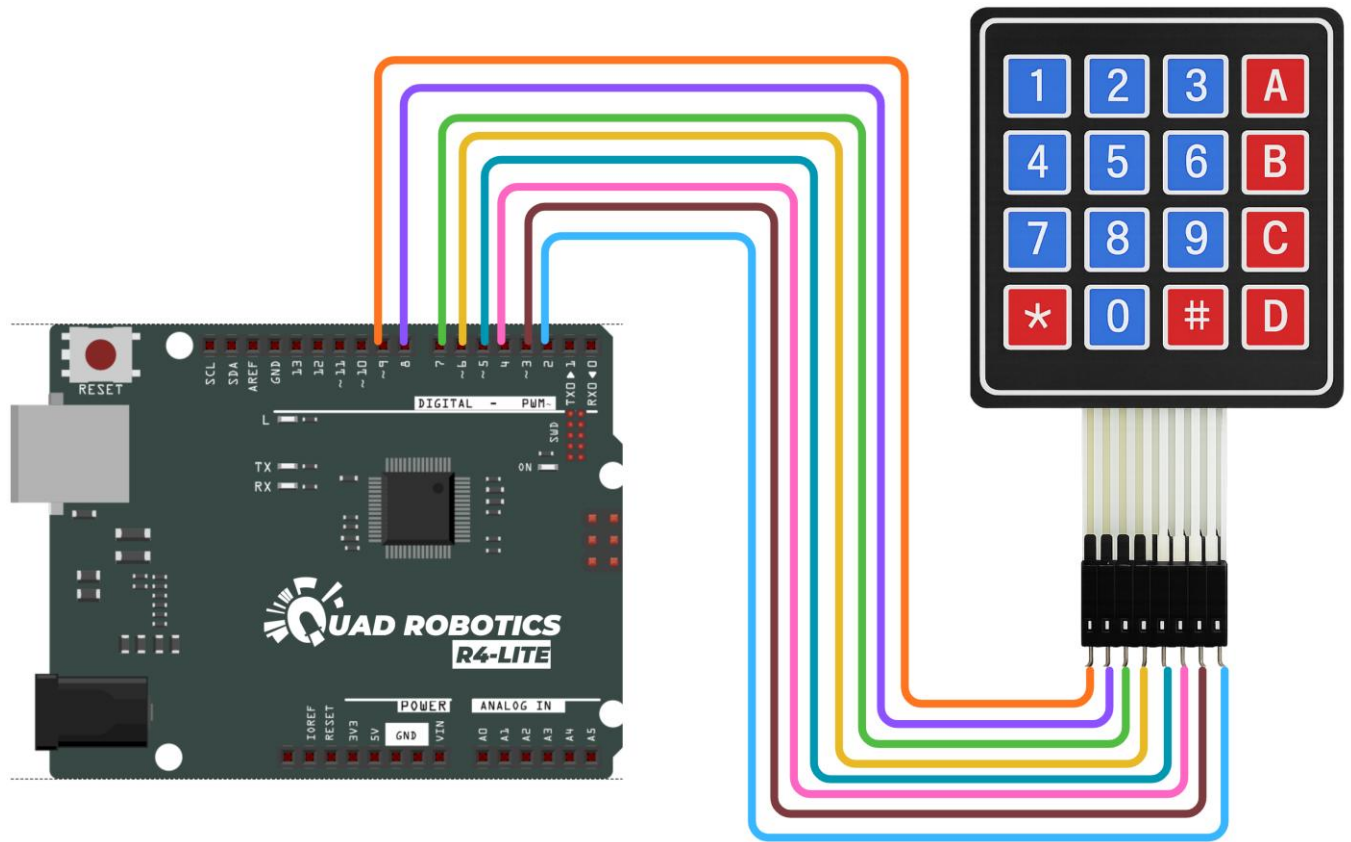
✅ **Standard Rule:** Pins are counted from **LEFT → RIGHT** (when keypad is facing you and cable is at the bottom)

⚠️ **Note:**

- Keypad has **8 pins (4 rows + 4 columns)**
- No VCC/GND required (pure digital scanning)



Component - Keypad (Left to Right)	Connection Method	Uno R4
Keypad Pin - L1	Direct Connection	Pin D9
Keypad Pin - L2	Direct Connection	Pin D8
Keypad Pin - L3	Direct Connection	Pin D7
Keypad Pin - L4	Direct Connection	Pin D6
Keypad Pin - C1	Direct Connection	Pin D5
Keypad Pin - C2	Direct Connection	Pin D4
Keypad Pin - C3	Direct Connection	Pin D3
Keypad Pin - C4	Direct Connection	Pin D2



✅ Required Library

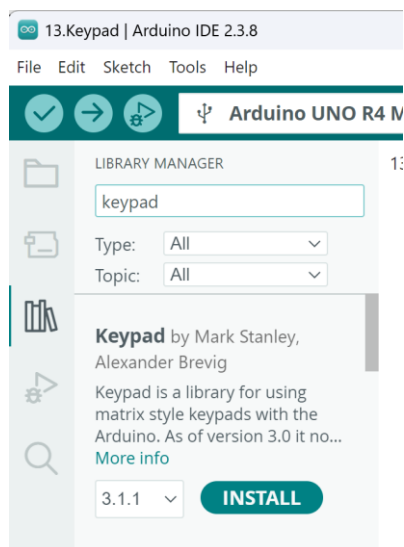
Install this library in Arduino IDE:

Keypad

(Author: Mark Stanley, Alexander Brevig)

Steps:

Sketch → Include Library → Manage Libraries → search "**Keypad**" → Install



Code

```
#include <Keypad.h>

const byte ROWS = 4;
const byte COLS = 4;

// Define keypad layout
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

// Connect to Arduino pins
byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};

// Create keypad object
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

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

void loop() {
  char key = keypad.getKey();

  if (key) {
    Serial.print("Key Pressed: ");
    Serial.println(key);
  }
}
```

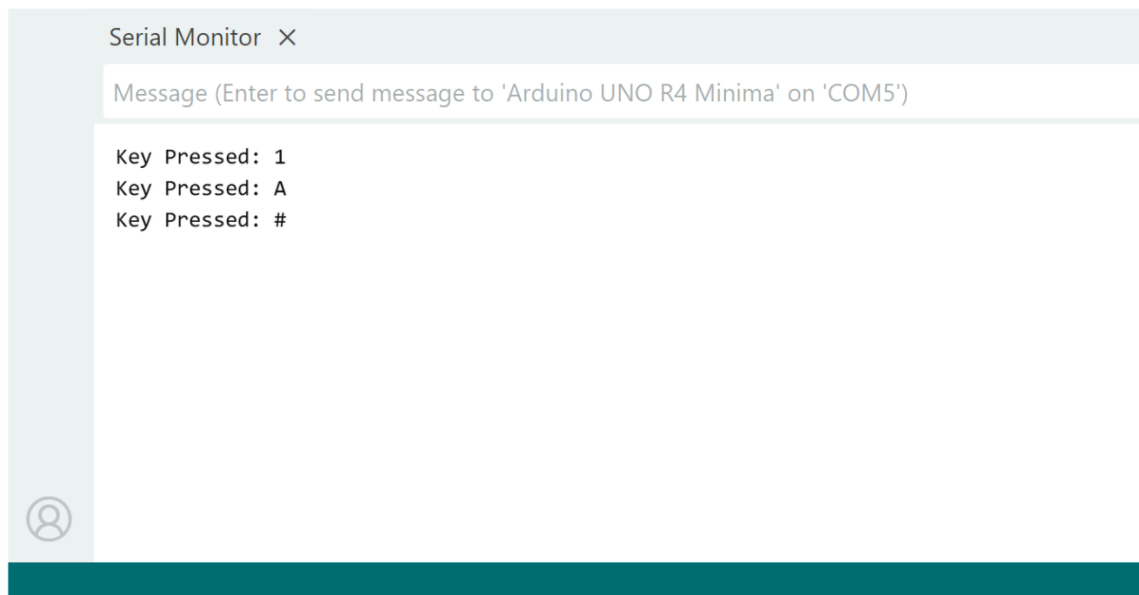
Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **13.Keypad.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.
9. Open Serial Monitor (Ctrl + Shift + M).
Press keys on keypad → output will be displayed





Output

Open **Serial Monitor (Ctrl + Shift + M)**

Press keys on keypad → output will be displayed.



DIY Extension:

- Build a password lock system 
- Control robot movement using keypad 
- Create a calculator project  
- Interface with LCD to display input

Project 14: Joystick Control

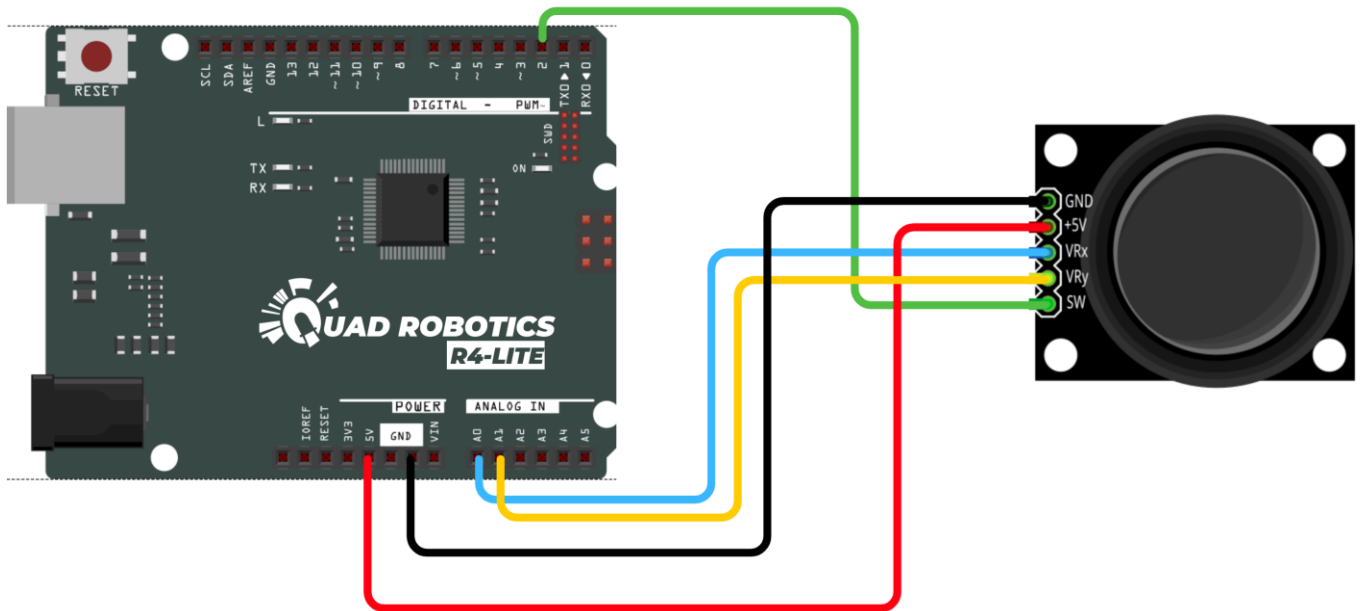
🌟 Objective: Read X, Y axis and button input from joystick module and display values on Serial Monitor.

🧰 Components Required

- ✓ UNO R4 board
- ✓ Joystick Module
- ✓ Jumper wires (Male to Female)

💡 Circuit Diagram

Component - Joystick	Connection Method	Uno R4
Joystick GND	Direct Connection	GND
Joystick 5v	Direct Connection	5v
Joystick VRx	Direct Connection	Pin A0
Joystick VRy	Direct Connection	Pin A1
Joystick SW	Direct Connection	Pin D2



Code

```
int xPin = A0;
int yPin = A1;
int swPin = 2;

void setup() {
  Serial.begin(9600);
  pinMode(swPin, INPUT_PULLUP);
}

void loop() {
  int xValue = analogRead(xPin);
  int yValue = analogRead(yPin);
  int button = digitalRead(swPin);

  Serial.print("X: ");
  Serial.print(xValue);
  Serial.print(" | Y: ");
  Serial.print(yValue);
  Serial.print(" | Button: ");

  if(button == LOW){
    Serial.println("Pressed");
  } else {
    Serial.println("Not Pressed");
  }

  delay(200);
}
```

Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **13.Keypad.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.
9. Open Serial Monitor (Ctrl + Shift + M).
Press keys on keypad → output will be displayed

Explanation & Output

```
Serial Monitor X
Message (Enter to send message to 'Arduino UNO R4 Minima' on 'COM5')
X: 512 | Y: 520 | Button: Not Pressed
X: 1023 | Y: 500 | Button: Not Pressed
X: 300 | Y: 800 | Button: Pressed
```

- Joystick has 2 potentiometers (X & Y axis)
- Center position ≈ 512
- Moving:
Left/Right \rightarrow changes X value
Up/Down \rightarrow changes Y value
Pressing joystick \rightarrow button (SW).

DIY Extension

Control robot movement (F/B/L/R)
Make a game controller interface
Control servo motors (pan/tilt system)
Build a menu navigation system with LCD.

Mini Project-1: Automated Gate/Smart dustbin

✦ Objective: To make an automated gate for a miniature model using Infrared IR sensor so that when an object comes within a threshold the system:

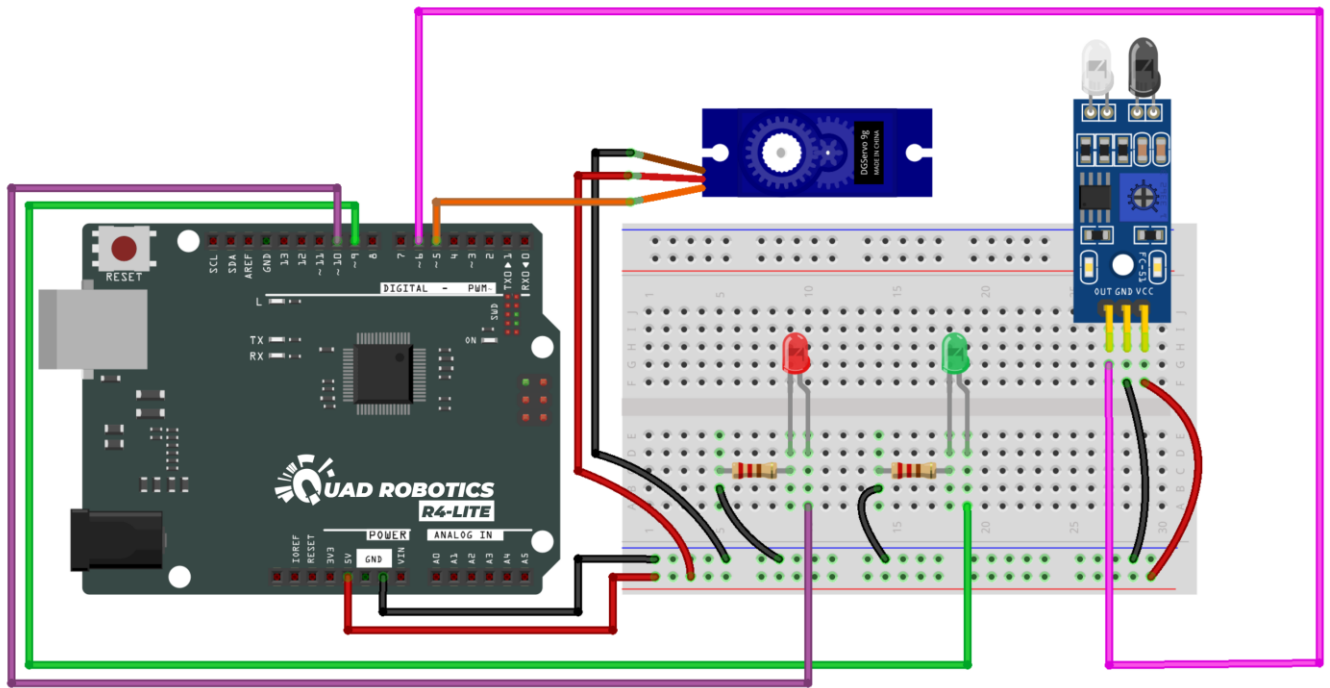
- turns Red LED ON,
- moves a servo to *open* the gate/ Or *open* the dustbin lid (e.g., rotate to opening angle),
and when the object is away:
- turns Green LED ON,
- moves the servo to *close* the gate / close the dustbin lid.

Components Required

- ✓ UNO R4
- ✓ Infrared IR sensor
- ✓ 1 × Red LED + 220Ω resistor
- ✓ 1 × Green LED + 220Ω resistor
- ✓ 1 × SG90 Servo Motor
- ✓ Jumper wires
- ✓ Breadboard
- ✓ USB cable for Arduino

Circuit Diagram

Component - Infrared Sensor	Connection Method	Uno R4
Infrared Sensor – VCC	Direct Connection	5V
Infrared Sensor – GND	Direct Connection	GND
Infrared Sensor – OUT	Direct Connection	Pin D6
Component - LED	Connection Method	Uno R4
Green LED (Anode, Long Leg +)	Through 220Ω resistor	Pin D9
Green LED (Cathode, Short Leg –)	Direct Connection	GND
Red LED (Anode, Long Leg +)	Through 220Ω resistor	Pin D10
Red LED (Cathode, Short Leg –)	Direct Connection	GND
Component - Servo Motor	Connection Method	Uno R4
Servo Signal (Orange wire)	Direct Connection	Pin D5
Servo VCC (Red wire)	External 5V supply (<i>recommended</i>)	5V
Servo GND (Brown wire)	Common Ground	GND



Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste above code** into the Arduino IDE (OR) open file **1.MiniProject1.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Code

```
// UNO R4 - IR + LEDs + Servo Gate + Dustbin

#include <Servo.h>

int servoPin = 5;
int irPin = 6;
int redLED = 10;
int greenLED = 9;

Servo myServo;

unsigned long openTime = 0;
```



```

bool gateOpen = false;

void setup() {
  pinMode(irPin, INPUT);
  pinMode(redLED, OUTPUT);
  pinMode(greenLED, OUTPUT);

  myServo.attach(servoPin);
  myServo.write(0); // Gate closed
}

void loop() {
  int sensorValue = digitalRead(irPin);

  // Object detected
  if (sensorValue == LOW && !gateOpen) {
    digitalWrite(redLED, HIGH);
    digitalWrite(greenLED, LOW);

    myServo.write(90); // Open gate
    openTime = millis(); // store time
    gateOpen = true;
  }

  // If gate is open, wait before closing
  if (gateOpen && (millis() - openTime > 3000)) {
    digitalWrite(redLED, LOW);
    digitalWrite(greenLED, HIGH);

    myServo.write(0); // Close gate
    gateOpen = false;
  }

  delay(120); // measurement pacing
}

```

Output

After uploading the code:

- On power-up: gate closed, Green LED ON.
- When an object moves within the threshold (e.g., car toy or hand): Red LED ON, servo rotates to open gate (smooth motion).
- When object moves away: Green LED ON, servo rotates back to closed position.

NOTE: The same wiring setup and code can be used for Smart Dustbin project as well. The concept is the same.

DIY Extension

- **Manual override:** add a push button to manually open/close the gate.
- **Safety:** add a limit switch to detect fully open/closed positions.
- **Add buzzer:** beep when gate is opening or when object detected.

Mini Project -2: Digital Locker

✦ Objective: Enter a **4-digit PIN** using the keypad.

If correct → **Servo unlocks door + LCD shows “Access Granted”**

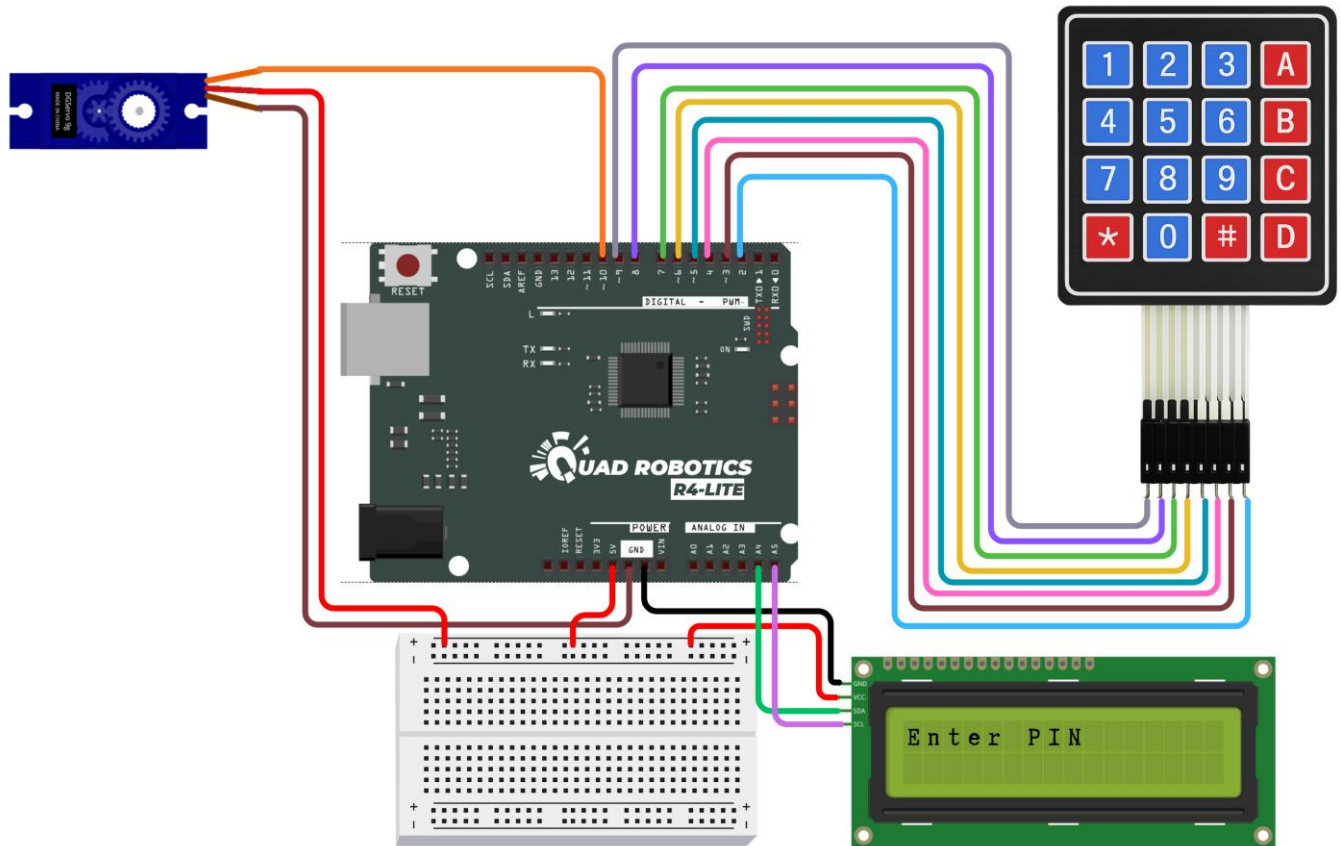
If wrong → **LCD shows “Wrong Password”**

Components Required

- ✓ UNO R4
- ✓ 4 × 4 Keypad Module
- ✓ I2C 1602 LCD
- ✓ Servo motor (SG90)
- ✓ Jumper wires
- ✓ Breadboard

Circuit Diagram

Component - Keypad (Left to Right)	Connection Method	Uno R4
Keypad Pin - L1	Direct Connection	Pin D9
Keypad Pin - L2	Direct Connection	Pin D8
Keypad Pin - L3	Direct Connection	Pin D7
Keypad Pin - L4	Direct Connection	Pin D6
Keypad Pin - C1	Direct Connection	Pin D5
Keypad Pin - C2	Direct Connection	Pin D4
Keypad Pin - C3	Direct Connection	Pin D3
Keypad Pin - C4	Direct Connection	Pin D2
Component - Servo Motor	Connection Method	Uno R4
Servo Signal (Orange wire)	Direct Connection	Pin D10
Servo VCC (Red wire)	External 5V supply (<i>recommended</i>)	5V
Servo GND (Brown wire)	Common Ground	GND
Component - LCD Display	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	5V
I2C LCD – GND	Direct Connection	GND
I2C LCD – SDA	Direct Connection	A4
I2C LCD – SCL	Direct Connection	A5



Steps to Upload Code

1. Connect your **UNO R4** board to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste** above code into the Arduino IDE (OR) open file **2.MiniProject2.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Code

```
#include <Keypad.h>
#include <Servo.h>
#include <LiquidCrystal_I2C.h>

// LCD setup
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Servo setup
Servo lockServo;
int servoPin = 10;

// Keypad setup
const byte ROWS = 4;
const byte COLS = 4;

char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

// Pins based on your wiring
byte rowPins[ROWS] = {8, 7, 6, 5};
byte colPins[COLS] = {4, 3, 2, 1};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

// Password
String correctPassword = "1234";
String enteredPassword = "";

void setup() {
  lcd.init();
  lcd.backlight();

  lockServo.attach(servoPin);
  lockServo.write(0); // Locked position

  lcd.setCursor(0,0);
  lcd.print("Enter PIN:");
}

void loop() {
  char key = keypad.getKey();

  if (key) {
    lcd.setCursor(0,1);
    lcd.print("          "); // clear line
    lcd.setCursor(0,1);

    if (key == '#') {
      checkPassword();
    }
    else if (key == '*') {
      enteredPassword = "";
      lcd.print("Cleared");
    }
  }
}
```



```

        delay(1000);
        lcd.clear();
        lcd.print("Enter PIN:");
    }
    else {
        enteredPassword += key;
        lcd.print(enteredPassword);
    }
}
}

void checkPassword() {
    lcd.clear();

    if (enteredPassword == correctPassword) {
        lcd.print("Access Granted");

        lockServo.write(90); // Unlock
        delay(3000);

        lockServo.write(0); // Lock again
    } else {
        lcd.print("Wrong Password");
        delay(2000);
    }

    enteredPassword = "";
    lcd.clear();
    lcd.print("Enter PIN:");
}

```

Output

Working Explanation

◆ Step-by-step flow:

1. LCD shows **“Enter PIN”**
2. User types digits which will be displayed on LCD
3. Press # to the submit password
4. If correct:
 - Servo rotates to **90° (unlock)**
 - Wait 3 seconds
 - Returns to **0° (lock)**

If wrong: Shows error message

Mini Project -3: Smart Weather Station

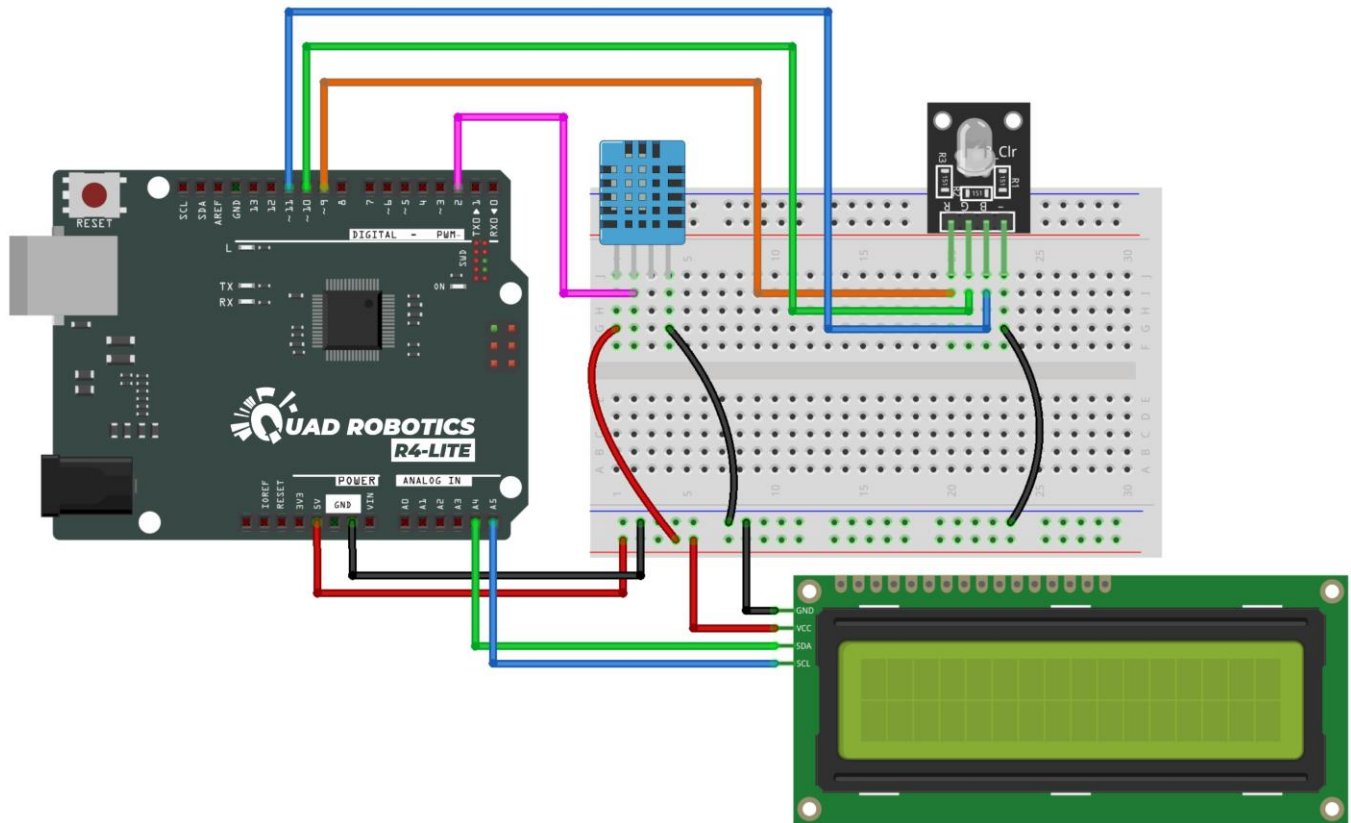
✦ Objective: Build a mini weather station that shows temperature & humidity on the LCD and uses RGB LED to show climate condition.

Components Required

- ✓ UNO R4
- ✓ DHT11 sensor
- ✓ I2C 1602 LCD
- ✓ RGB LED
- ✓ Jumper wires

Circuit Diagram

Component - DHT11 Temperature Sensor	Connection Method	Uno R4
DHT11 – Signal	Direct Connection	Pin D2
DHT11 – VCC	Direct Connection	5V
DHT11 – GND	Direct Connection	GND
Component - LCD Display	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	5V
I2C LCD – GND	Direct Connection	GND
I2C LCD – SDA	Direct Connection	A4
I2C LCD – SCL	Direct Connection	A5
Component - RGB Led	Connection Method	Uno R4
RGB LED – Red (R)	Direct Connection	Pin D9
RGB LED – Green (G)	Direct Connection	Pin D10
RGB LED – Blue (B)	Direct Connection	Pin D11
RGB LED – Common Cathode (-)	Direct Connection	GND



Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste below code** into the Arduino IDE (OR) open file **3.MiniProject3.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Code

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

LiquidCrystal_I2C lcd(0x27, 16, 2);

#define DHTPIN 2
#define DHTTYPE DHT11
```



```

DHT dht(DHTPIN, DHTTYPE);

int R = 9, G = 10, B = 11;

void setColor(int r, int g, int b) {
  analogWrite(R, r);
  analogWrite(G, g);
  analogWrite(B, b);
}

void setup() {
  lcd.init();
  lcd.backlight();
  dht.begin();
  pinMode(R, OUTPUT);
  pinMode(G, OUTPUT);
  pinMode(B, OUTPUT);
}

void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Temp: "); lcd.print(t); lcd.print("C");
  lcd.setCursor(0, 1);
  lcd.print("Hum : "); lcd.print(h); lcd.print("%");

  if (t < 20) setColor(0, 0, 255);    // cold - blue
  else if (t > 30) setColor(255, 0, 0); // hot - red
  else setColor(0, 255, 0);          // ideal - green

  delay(1500);
}

```

Output

After uploading the code:

- LCD updates every 1.5 seconds.
- RGB LED shows climate condition.

DIY Extension

◆ Add more emotions

- Add buzzer alarm for extreme heat
- Add humidity-controlled fan

Mini Project -4: Stickman Jump Game

✦ Objective: Create a simple stickman jumping game to avoid obstacles:

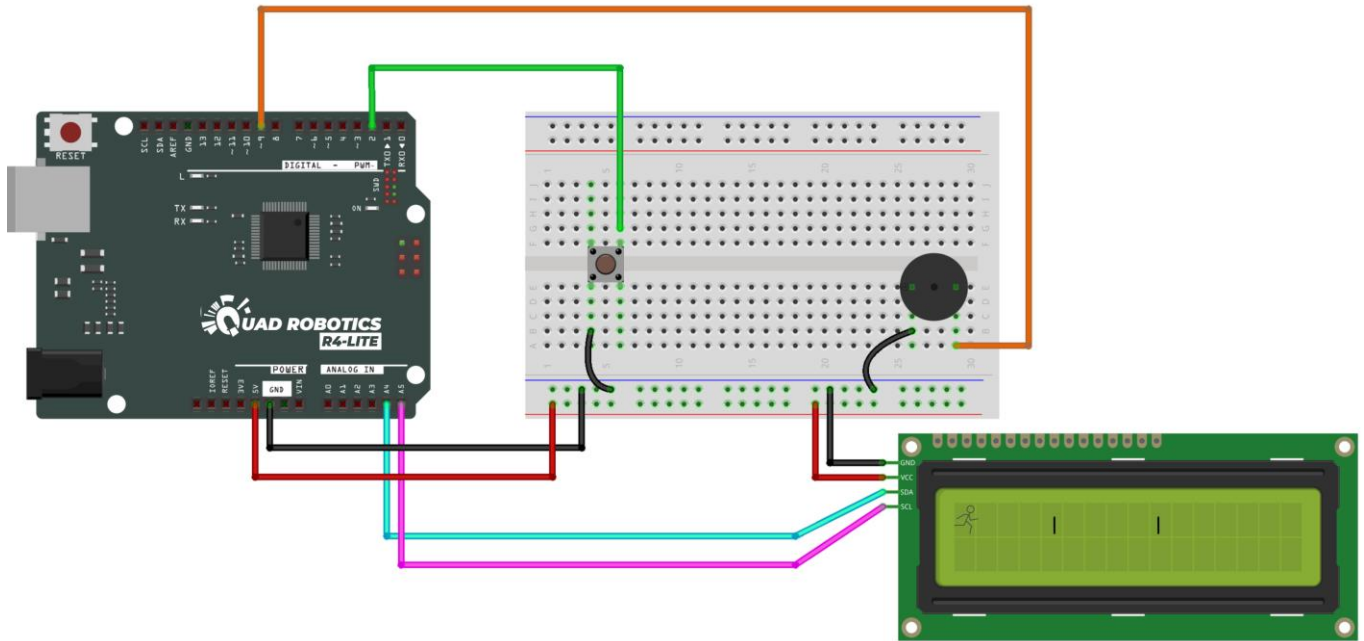
- man runs continuously
- Obstacles come from right
- Player uses **push button to jump**
- Avoid collision → score increases

Components Required

- ✓ UNO R4
- ✓ Push Button
- ✓ I2C 1602 LED display
- ✓ Active Buzzer
- ✓ Jumper wires
- ✓ Breadboard

Circuit Diagram

Component - LCD Display	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	5V
I2C LCD – GND	Direct Connection	GND
I2C LCD – SDA	Direct Connection	A4
I2C LCD – SCL	Direct Connection	A5
Component - Push Buttons	Connection Method	Uno R4
Push Button (One Leg)	Direct Connection	Pin D2
Push Button (Other Leg)	Direct Connection	GND
Component - Buzzer	Connection Method	Uno R4
Buzzer (Positive +)	Direct Connection	Pin D9
Buzzer (Negative –)	Direct Connection	GND



Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste below code** into the Arduino IDE (OR) open file **4.MiniProject4.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Code

```
// Stickman Jumping Game

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

LiquidCrystal_I2C lcd(0x27, 16, 2);

// Pins
int buttonPin = 2;
int buzzerPin = 9;

// Game variables
int playerY = 1;
int obstacleX = 15;
int score = 0;

bool jumping = false;
unsigned long jumpStart = 0;

// Speed control
int gameSpeed = 200;
unsigned long lastSpeedIncrease = 0;

// Custom characters
byte stickman[8] = {
    B00100,
    B01110,
    B00100,
    B01110,
    B10101,
    B00100,
    B01010,
    B10001
};

byte obstacle[8] = {
    B00100,
    B00100,
    B01110,
    B01110,
    B01110,
    B00100,
    B00100,
    B00000
};

void setup() {
    pinMode(buttonPin, INPUT_PULLUP);
    pinMode(buzzerPin, OUTPUT);

    lcd.init();
    lcd.backlight();

    lcd.createChar(0, stickman);
    lcd.createChar(1, obstacle);

    lcd.setCursor(0,0);
```



```

    lcd.print("Stickman Game");
    delay(2000);
    lcd.clear();
}

void loop() {
    // Jump
    if (digitalRead(buttonPin) == LOW && !jumping) {
        jumping = true;
        jumpStart = millis();
        playerY = 0;

        tone(buzzerPin, 1000, 100); // jump sound
    }

    // Jump duration
    if (jumping && millis() - jumpStart > 400) {
        playerY = 1;
        jumping = false;
    }

    // Move obstacle
    obstacleX--;
    if (obstacleX < 0) {
        obstacleX = 15;
        score++;

        tone(buzzerPin, 1500, 80); // score sound
    }

    // Collision detection
    if (obstacleX == 1 && playerY == 1) {
        gameOver();
    }

    // Increase speed
    if (millis() - lastSpeedIncrease > 5000) {
        if (gameSpeed > 80) {
            gameSpeed -= 10;
        }
        lastSpeedIncrease = millis();
    }

    drawGame();

    delay(gameSpeed);
}

void drawGame() {
    lcd.clear();

    lcd.setCursor(1, playerY);
    lcd.write(byte(0));

    lcd.setCursor(obstacleX, 1);
    lcd.write(byte(1));

    lcd.setCursor(10, 0);
    lcd.print("S:");
}

```



```
    lcd.print(score);
}

void gameOver() {
    lcd.clear();
    lcd.setCursor(2, 0);
    lcd.print("GAME OVER");

    lcd.setCursor(2, 1);
    lcd.print("Score:");
    lcd.print(score);

    // Game over sound
    for (int i = 0; i < 3; i++) {
        tone(buzzerPin, 500, 200);
        delay(250);
    }

    delay(2000);

    // Reset
    obstacleX = 15;
    score = 0;
    gameSpeed = 200;
}
```

Output

After uploading the code:

The LCD displays a stickman running while obstacles move from right to left. Pressing the button makes the stickman jump to avoid obstacles and score points. The game speed increases over time, and a buzzer provides sound feedback for actions and game over.

Mini Project -5: Car Dodging Game

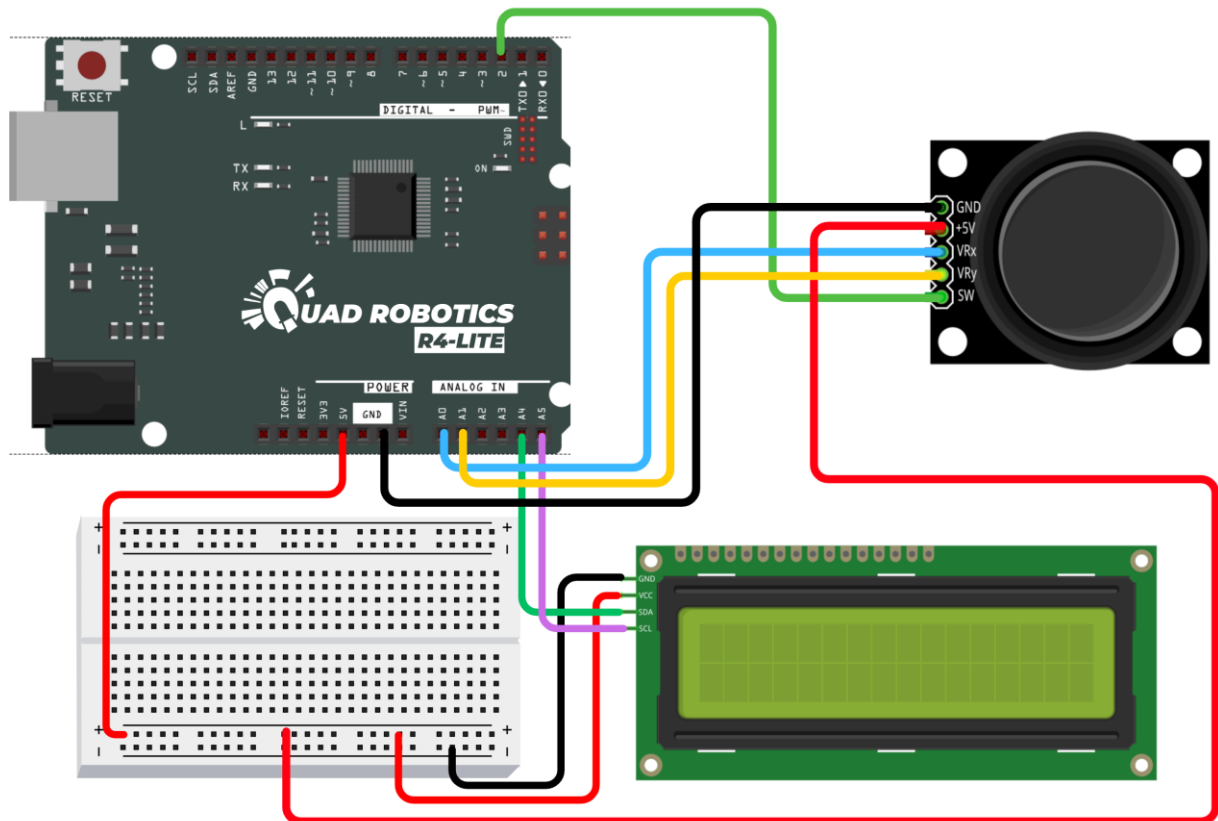
🌟 Objective: Control a car using a joystick to switch lanes and avoid incoming obstacles. Score increases over time, and speed increases gradually to make the game harder.

🛠 Components Required

- ✓ UNO R4 board
- ✓ Joystick Module
- ✓ I2C LCD display

💡 Circuit Diagram

Component - Joystick	Connection Method	Uno R4
Joystick GND	Direct Connection	GND
Joystick 5v	Direct Connection	5v
Joystick VRx	Direct Connection	Pin A0
Joystick VRy	Direct Connection	Pin A1
Joystick SW	Direct Connection	Pin D2
Component - LCD Display	Connection Method	Uno R4
I2C LCD – VCC	Direct Connection	5V
I2C LCD – GND	Direct Connection	GND
I2C LCD – SDA	Direct Connection	A4
I2C LCD – SCL	Direct Connection	A5



Steps to Upload Code

1. Connect your **UNO R4 board** to the computer using a **USB cable**.
2. Open **Arduino IDE** on your computer.
3. Go to **Tools > Board** and select **Arduino UNO R4**.
4. Go to **Tools > Port** and choose the correct **COM port**.
5. Either **Copy & Paste below code** into the Arduino IDE (OR) open file **5.MiniProject5.ino** from the provided **CODE** folder.
6. **NOTE:** If you open the file directly from CODE folder, you will need to select the Board again from the drop-down list. Select as **Arduino Uno R4 minima** from drop down list.
7. Click on the **Upload button** (right arrow icon).
8. Wait for the code to **compile and upload**. The board will start running the program automatically.

Code

```
// Car GAME
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

// Joystick pins
int joyX = A0;
int joyY = A1;
int joySW = 2;

// Game variables
int playerLane = 1;
int obstacleX = 15;
int obstacleLane = 0;

int score = 0;
int gameSpeed = 200;
unsigned long lastSpeedIncrease = 0;

// Clean car design
byte car[8] = {
    B00000,
    B01110,
    B10001,
    B11111,
    B10101,
    B11111,
    B10001,
    B00000
};

// Obstacle
byte obstacle[8] = {
    B00000,
    B00000,
    B00100,
    B01110,
```



```

    B11111,
    B01110,
    B00100,
    B00000
};

void setup() {
    pinMode(joySW, INPUT_PULLUP);

    lcd.init();
    lcd.backlight();

    lcd.createChar(0, car);
    lcd.createChar(1, obstacle);

    randomSeed(analogRead(A2));

    lcd.setCursor(0,0);
    lcd.print("Car Game");
    delay(1500);
    lcd.clear();
}

void loop() {
    readJoystick();
    moveObstacle();
    checkCollision();
    increaseDifficulty();
    drawGame();

    delay(gameSpeed);
}

void readJoystick() {
    int x = analogRead(joyX);

    if (x < 300) {
        playerLane = 0;
    } else if (x > 700) {
        playerLane = 1;
    }
}

void moveObstacle() {
    obstacleX--;

    if (obstacleX < 0) {
        obstacleX = 15;
        obstacleLane = random(0, 2);
        score++;
    }
}

void checkCollision() {
    if (obstacleX == 1 && obstacleLane == playerLane) {
        gameOver();
    }
}

```



```

void increaseDifficulty() {
  if (millis() - lastSpeedIncrease > 4000) {
    if (gameSpeed > 80) {
      gameSpeed -= 10;
    }
    lastSpeedIncrease = millis();
  }
}

void drawGame() {
  lcd.clear();

  lcd.setCursor(1, playerLane);
  lcd.write(byte(0));

  lcd.setCursor(obstacleX, obstacleLane);
  lcd.write(byte(1));

  lcd.setCursor(10, 0);
  lcd.print("S:");
  lcd.print(score);
}

void gameOver() {
  lcd.clear();
  lcd.setCursor(3, 0);
  lcd.print("GAME OVER");

  lcd.setCursor(3, 1);
  lcd.print("Score:");
  lcd.print(score);

  delay(2500);

  score = 0;
  gameSpeed = 200;
  obstacleX = 15;
}

```

Output

After uploading the code:

The LCD displays a car moving in two lanes while obstacles approach from the opposite side. The player uses the joystick to switch lanes and avoid collisions. The score increases as obstacles are successfully dodged, and the game speed gradually increases to make it more challenging.