5.Sensors - (Part 1 - LDR)

5.1.LDR Sensors

5.1.1.a.Reading value from LDR

https://www.tinkercad.com/things/g6oWlOqTbrc-511areading-value-from-ldr

```
/**************************************************************************
**
Program Name: Reading value from LDR (Light Dependent Resistor)

Program Objective:Printing the value read from LDR to serial output


Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
// Reading value from LDR (Light Dependent Resistor)
void setup()
{
  // Put your setup code here, to run once:
  // Initializes serial communication at 9600 bits per second
  Serial.begin(9600);

}

void loop()
{
  // Reading the value from analog input A0
  // 'ldr' variable will hold the value read from LDR
  int ldr = analogRead(A0);

  // Print the LDR value to the serial monitor
  Serial.println(ldr);
}
```

5.1.1.b.Observing the output according to different LDR values

https://www.tinkercad.com/things/9cWJo5hZlkT-511bobserving-the-output-according-to-different-ldr-values

```
/**************************************************************************
**
Program Name: Observing the changes in serial output by selecting

Program Objective:

This code reads the values of Light Dependent Resistors (LDR)
with different resistance values connected to the Arduino's
analog inputs A0, A1, A2 and A3 and prints these values to
the serial monitor. With the delay(1000) command, a 1-second
waiting period is added between each reading, thus allowing
the values to be easily observed on the serial monitor. Since
each LDR is connected with a different resistor value, different
analog values will be read depending on the light intensity. This
code is used to observe the response of LDR with different
resistors.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
// 100, 1k, 4.7k, and 10k resistors and reading the values
// from analog inputs depending on the current

void setup()
{
  // Put your setup code here, to run once:
  // Initializes serial communication at 9600 bits per second
  Serial.begin(9600);
}

void loop()
{
  // Reading the value from analog inputs
  // Variables 'ldr0', 'ldr1', 'ldr2', and 'ldr3' will hold the values read from LDRs
  int ldr0 = analogRead(A0); // Reading from LDR with 100 ohm resistor
  int ldr1 = analogRead(A1); // Reading from LDR with 1k ohm resistor
  int ldr2 = analogRead(A2); // Reading from LDR with 4.7k ohm resistor
  int ldr3 = analogRead(A3); // Reading from LDR with 10k ohm resistor

  // Print the LDR values to the serial monitor
  Serial.println(ldr0); // Print value from LDR with 100 ohm resistor
```

```
  Serial.println(ldr1); // Print value from LDR with 1k ohm resistor
  Serial.println(ldr2); // Print value from LDR with 4.7k ohm resistor
  Serial.println(ldr3); // Print value from LDR with 10k ohm resistor

  // Wait for a second before the next loop iteration
  delay(1000);
}
```

5.1.1.c.Observing the output according to different LDR values (opt)

https://www.tinkercad.com/things/j2EBTNdLzPn-511cobserving-different-ldr-values-opt

```
/**********************************************************************
**
Program Name: Reading values from multiple LDRs with different
resistors and printing to serial monitor

 (opt)

Program Objective:

This code reads the values of Light Dependent Resistors (LDR)
with different resistance values connected to the Arduino's
analog inputs A0, A1, A2 and A3 and prints these values to
the serial monitor. With the delay(1000) command, a 1-second
waiting period is added between each reading, thus allowing
the values to be easily observed on the serial monitor. Since
each LDR is connected with a different resistor value, different
analog values will be read depending on the light intensity. This
code is used to observe the response of LDR with different
resistors.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**********************************************************************
***/


const int analogPins[] = {A0, A1, A2, A3}; // Analog pin numbers for LDRs
const int numSensors = 4; // Number of LDR sensors

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

void loop() {
  for (int i = 0; i < numSensors; i++) {
    int sensorValue = readLDR(analogPins[i]);
    printSensorValue(i, sensorValue);
  }
  delay(1000);
}

int readLDR(int pin) {
  return analogRead(pin); // Read value from LDR
```

```
}

void printSensorValue(int sensorNumber, int value) {
  Serial.print("Sensor ");
  Serial.print(sensorNumber);
  Serial.print(": ");
  Serial.println(value); // Print LDR value to serial monitor
}
```

5.1.2.a. 5.1.2.a.Changing LDR connected LED brightnessy

https://www.tinkercad.com/things/4SdNUrjLwPK-512achanging-ldr-connected-led-brightnessy

```
/**********************************************************************
**
Program Name:Changing LDR connected LED brightness

Program Objective: Adjusting LED brightness based on ambient light intensity


Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**********************************************************************
***/
// Adjusting LED brightness based on ambient light intensity

int ldrAnalog = A0; // Analog pin connected to LDR
int ledPin = 11;    // PWM capable pin connected to LED
int potValue = 0;   // Variable to store the LDR value
int mapValue = 0;   // Variable to store the mapped value

void setup()
{
  pinMode(ldrAnalog, INPUT); // Set LDR pin as input
  pinMode(ledPin, OUTPUT);   // Set LED pin as output
  Serial.begin(9600);        // Start serial communication at 9600 bps
}

void loop()
{
  potValue = analogRead(ldrAnalog); // Read the value from LDR
  // Map the LDR value from 0-1023 to a range of 0-255
  mapValue = map(potValue, 0, 1024, 0, 255);

  analogWrite(ledPin, mapValue); // Set the LED brightness
  Serial.println(mapValue);      // Print the mapped value to the serial monitor
  delay(5);                      // Short delay for stability
}
```

5.1.2.b.Changing LDR connected LED brightnessy (opt)

https://www.tinkercad.com/things/duJuSIic1uR-512bchanging-ldr-connected-led-brightnessy-opt

```
/**************************************************************************
**
Program Name:Changing LDR connected LED brightness

Program Objective: Adjusting LED brightness based on ambient light intensity
Hysteresis: We use a certain threshold value (10) to vary
the LED brightness. This prevents the LED from flashing quickly
when there are small changes in ambient light.

Debounce Function: The debounceLdr function helps reduce sudden
and unwanted changes between the values read from the LDR.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
// Adjusting LED brightness based on ambient light intensity with debounce and hysteresis

const int ldrAnalog = A0;
const int ledPin = 11;
int lastLdrValue = 0;
int ledBrightness = 0;

void setup() {
  pinMode(ldrAnalog, INPUT);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int ldrValue = analogRead(ldrAnalog);
  ldrValue = debounceLdr(ldrValue, lastLdrValue);

  if (abs(ldrValue - lastLdrValue) > 10) { // Hysteresis threshold
    lastLdrValue = ldrValue;
    ledBrightness = map(ldrValue, 0, 1024, 0, 255);
    analogWrite(ledPin, ledBrightness);
    Serial.print("LDR Value: ");
    Serial.print(ldrValue);
    Serial.print(", LED Brightness: ");
    Serial.println(ledBrightness);
  }
```

```
  delay(5);
}

int debounceLdr(int currentValue, int lastValue) {
  // Simple debounce algorithm
  if (abs(currentValue - lastValue) > 5) {
    return currentValue;
  } else {
    return lastValue;
  }
}
```

5.1.3.a.Street Lamp Controller

https://www.tinkercad.com/things/fqsN4qDcMbZ-513astreet-lamp-controller

```
/**************************************************************************
**
Program Name:  Street Lamp Controller

Program Objective:

This code is used to automatically control a street light.
The lamp turns on when the ambient light drops below a certain
threshold and turns off when the light reaches a sufficient
level. This simple control mechanism helps street lights save
energy.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/

int ldrAnalog = A0; // Analog pin connected to the LDR
int relayPin = 11;  // Digital pin connected to the relay
int ldrValue = 0;   // Variable to store the LDR value

void setup()
{
  pinMode(ldrAnalog, INPUT); // Set LDR pin as input
  pinMode(relayPin, OUTPUT); // Set relay pin as output
  Serial.begin(9600);        // Start serial communication at 9600 bps
}

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

  if (ldrValue > 200) {
    digitalWrite(relayPin, HIGH); // Turn on the street lamp (relay on)
  } else {
    digitalWrite(relayPin, LOW);  // Turn off the street lamp (relay off)
  }

  Serial.println(ldrValue); // Print the LDR value to the serial monitor
  delay(5);                 // Short delay for stability
}
```

5.1.3.b.Street Lamp Controller (opt)

https://www.tinkercad.com/things/jzbqEB6eNwC-513bstreet-lamp-controller-opt

```
/**************************************************************************
**
Program Name:  Street Lamp Controller (opt)

Program Objective:

Improvements made to this code:

Hysteresis: A hysteresis threshold is used to prevent the lamp
from turning on and off quickly

Adjustable Threshold Values: The threshold value is stored
in a variable instead of a fixed number, so it can be easily
adapted to different environments.

Functions: shouldToggleLamp and toggleLamp functions, made
the code more readable and modular

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/

// Street Lamp Controller with adjustable threshold and hysteresis

const int ldrAnalog = A0;
const int relayPin = 11;
const int threshold = 200; // Adjustable threshold for LDR
const int hysteresis = 10; // Hysteresis value to prevent flickering
int lastLdrValue = 0;

void setup() {
  pinMode(ldrAnalog, INPUT);
  pinMode(relayPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int ldrValue = analogRead(ldrAnalog);

  if (shouldToggleLamp(ldrValue, lastLdrValue, threshold, hysteresis)) {
    toggleLamp(relayPin);
    lastLdrValue = ldrValue;
```

```
  }

  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  delay(5);
}

bool shouldToggleLamp(int currentValue, int lastValue, int threshold, int hysteresis) {
  if (abs(currentValue - lastValue) > hysteresis) {
    return (currentValue > threshold && lastValue <= threshold) ||
           (currentValue <= threshold && lastValue > threshold);
  }
  return false;
}

void toggleLamp(int pin) {
  bool isLampOn = digitalRead(pin);
  digitalWrite(pin, !isLampOn);
  Serial.println(isLampOn ? "Lamp turned off" : "Lamp turned on");
}
```

5.1.4.a.LDR Alarm System

https://www.tinkercad.com/things/l4TstqprD0I-514aldr-alarm-system

```
/**************************************************************************
**
Program Name:  LDR Alarm System

Program Objective:

This code creates an alarm system using an LDR. When a certain
light level is detected, it activates a relay and sounds
a buzzer. This system can be used to alert, for example,
if a room is entered without permission.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
//

int ldrAnalog = A0; // Analog pin connected to the LDR
int relayPin = 11;  // Digital pin connected to the relay
int ldrValue = 0;   // Variable to store the LDR value

void setup()
{
  pinMode(ldrAnalog, INPUT); // Set LDR pin as input
  pinMode(relayPin, OUTPUT); // Set relay pin as output
  Serial.begin(9600);        // Start serial communication at 9600 bps
}

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

  if (ldrValue > 200) {
    digitalWrite(relayPin, HIGH); // Activate relay
    tone(8, 800);                 // Start buzzer sound on pin 8 with frequency 800Hz
  } else {
    digitalWrite(relayPin, LOW);  // Deactivate relay
    noTone(8);                    // Stop buzzer sound on pin 8
  }

  Serial.println(ldrValue); // Print the LDR value to the serial monitor
  delay(5);                 // Short delay for stability
}
```

5.1.4.b.LDR Alarm System with continuous alert

https://www.tinkercad.com/things/e6a78uTZ4W4-514bldr-alarm-system-with-continuous-alert

```
/**********************************************************************
**
Program Name:  LDR Alarm System with continuous alert

Program Objective:

Make improvements to the code

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**********************************************************************
***/

const int ldrAnalog = A0;  // Analog pin connected to the LDR
const int relayPin = 11;   // Digital pin connected to the relay
const int buzzerPin = 8;   // Digital pin connected to the buzzer
const int threshold = 200; // Threshold for LDR value

void setup() {
  pinMode(ldrAnalog, INPUT);
  pinMode(relayPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int ldrValue = analogRead(ldrAnalog);

  if (ldrValue > threshold) {
    digitalWrite(relayPin, HIGH); // Activate relay
    tone(buzzerPin, 800);         // Start buzzer sound at 800Hz
  } else {
    digitalWrite(relayPin, LOW);  // Deactivate relay
    noTone(buzzerPin);            // Stop buzzer sound
  }

  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  delay(5);
}
```

5.1.4.c.LDR Alarm System with continuous alert and hysteresis

https://www.tinkercad.com/things/i3oUfFVwjZO-514cldr-alarm-system-with-continuous-alert-and-hysteresis

```
/**********************************************************************
**
Program Name:  LDR Alarm System with continuous alert and hysteresis

Program Objective:

Improvements made to this code:

Hysteresis: Prevented small LDR value changes from accidentally
triggering the alarm

Functions: shouldActivateAlarm, activateAlarm, and
deactivateAlarm functions improved code readability and
modularity

Buzzer Duration Control: A control has been added to prevent
the buzzer from operating continuously.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**********************************************************************
***/
// LDR Alarm System with continuous alert and hysteresis

const int ldrAnalog = A0;
const int relayPin = 11;
const int buzzerPin = 8;
const int threshold = 200; // Adjustable threshold for LDR
const int hysteresis = 10; // Hysteresis value
int lastLdrValue = 0;
bool isAlarmOn = false;

void setup() {
  pinMode(ldrAnalog, INPUT);
  pinMode(relayPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int ldrValue = analogRead(ldrAnalog);
```

```
  if (isAlarmConditionMet(ldrValue, threshold, hysteresis)) {
    if (!isAlarmOn) {
      activateAlarm(relayPin, buzzerPin);
      isAlarmOn = true;
    }
  } else {
    if (isAlarmOn) {
      deactivateAlarm(relayPin, buzzerPin);
      isAlarmOn = false;
    }
  }

  lastLdrValue = ldrValue;
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  delay(5);
}

bool isAlarmConditionMet(int currentValue, int threshold, int hysteresis) {
  return (currentValue > threshold && abs(currentValue - lastLdrValue) > hysteresis) ||
         (currentValue <= threshold && isAlarmOn);
}

void activateAlarm(int relay, int buzzer) {
  digitalWrite(relay, HIGH);
  tone(buzzer, 800); // Start buzzer sound at 800Hz
  Serial.println("Alarm Activated");
}

void deactivateAlarm(int relay, int buzzer) {
  digitalWrite(relay, LOW);
  noTone(buzzer); // Stop buzzer sound
  Serial.println("Alarm Deactivated");
}
```

5.1.5.a.Counter using LDR (0-9)

https://www.tinkercad.com/things/bjSuvjrdxnP-515acounter-using-ldr-0-9

```
/**************************************************************************
**
Program Name: Counter using LDR (Light Dependent Resistor)

Program Objective:

This code can be used for a simple object counting and visual
display system. For example, it can be used to count pieces
on a conveyor belt or to keep track of the number of people
passing through a room.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
**************************************************************************
**/
//

// Counter using LDR (Light Dependent Resistor)

int ldrAnalog = A0; // Analog pin connected to the LDR
int ldrValue = 0;   // Variable to store the LDR value
int counter = 0;    // Counter variable

// Setting up the pins as outputs
void setup() {
  pinMode(ldrAnalog, INPUT);

  // Setting digital pins 6 to 13 as outputs
  for (int pin = 13; pin > 5; pin--) {
    pinMode(pin, OUTPUT);
  }
  Serial.begin(9600);
  delay(5000); // Initial delay of 5 seconds
}

void loop() {
  ldrValue = analogRead(ldrAnalog);

  // If an object is detected (assuming lower LDR value indicates this)
  if (ldrValue < 400) {
    counter++;
    countUp();
    delay(5000); // Wait for 5 seconds before reading again
  }
```

```
  Serial.println(ldrValue);
  delay(5);
}

// Function to display the counter value
void countUp() {
  switch (counter) {
    case 0: displayZero(); break;
    case 1: displayOne(); break;
    case 2: displayTwo(); break;
    case 3: displayThree(); break;
    case 4: displayFour(); break;
    case 5: displayFive(); break;
    case 6: displaySix(); break;
    case 7: displaySeven(); break;
    case 8: displayEight(); break;
    case 9: displayNine(); break;
  }
}

// Functions to display numbers 0-9 by turning on/off LEDs connected to pins 10-13
void displayZero() {
  digitalWrite(13, LOW);
  digitalWrite(12, LOW);
  digitalWrite(11, LOW);
  digitalWrite(10, LOW);
}

void displayOne() {
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(11, LOW);
  digitalWrite(10, LOW);
}

void displayTwo() {
  digitalWrite(13, LOW);
  digitalWrite(12, HIGH);
  digitalWrite(11, LOW);
  digitalWrite(10, LOW);
}

void displayThree() {
  digitalWrite(13, HIGH);
  digitalWrite(12, HIGH);
  digitalWrite(11, LOW);
  digitalWrite(10, LOW);
}

void displayFour() {
```

```
  digitalWrite(13, LOW);
  digitalWrite(12, LOW);
  digitalWrite(11, HIGH);
  digitalWrite(10, LOW);
}

void displayFive() {
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(11, HIGH);
  digitalWrite(10, LOW);
}

void displaySix() {
  digitalWrite(13, LOW);
  digitalWrite(12, HIGH);
  digitalWrite(11, HIGH);
  digitalWrite(10, LOW);
}

void displaySeven() {
  digitalWrite(13, HIGH);
  digitalWrite(12, HIGH);
  digitalWrite(11, HIGH);
  digitalWrite(10, LOW);
}

void displayEight() {
  digitalWrite(13, LOW);
  digitalWrite(12, LOW);
  digitalWrite(11, LOW);
  digitalWrite(10, HIGH);
}

void displayNine() {
  digitalWrite(13, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(11, LOW);
  digitalWrite(10, HIGH);
}
```

5.1.5.b.Counter using LDR (0-9)_opt_1

https://www.tinkercad.com/things/hNsyiO2g2F8-515bcounter-using-ldr-0-9opt1

```
/**************************************************************************
**
Program Name: Counter using LDR (Light Dependent Resistor) (opt 1)

Program Objective:

This code can be used for a simple object counting and visual
display system. For example, it can be used to count pieces
on a conveyor belt or to keep track of the number of people
passing through a room.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
**************************************************************************
**/
// LED Display for numbers 0-9 using binary representation

const int ledPins[] = {10, 11, 12, 13}; // LED pins for display
const int numLeds = sizeof(ledPins) / sizeof(ledPins[0]);
const int ldrAnalog = A0; // Analog pin connected to the LDR
int ldrValue = 0; // Variable to store the LDR value
int counter = 0;  // Counter variable

// Binary representation of numbers 0-9
const int numberPatterns[10] = {
  0b0000, // 0
  0b0001, // 1
  0b0010, // 2
  0b0011, // 3
  0b0100, // 4
  0b0101, // 5
  0b0110, // 6
  0b0111, // 7
  0b1000, // 8
  0b1001  // 9
};

void setup() {
  pinMode(ldrAnalog, INPUT);
  for (int i = 0; i < numLeds; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
  // Since LDR min starts in the simulation,
  // we manually increase it to a higher value with a delay.
```

```
  delay(5000);
  Serial.begin(9600);
}

void displayNumber(int number) {
  if (number < 0 || number > 9) return; // Check if the number is within 0-9 range

  int pattern = numberPatterns[number];
  for (int i = 0; i < numLeds; i++) {
    digitalWrite(ledPins[i], (pattern >> i) & 1 ? HIGH : LOW);
  }
}

void loop() {
  ldrValue = analogRead(ldrAnalog);
  Serial.print("LDR Degeri: ");
  Serial.println(ldrValue);

  if (ldrValue < 400) {
    counter = (counter + 1) % 10; // Increment counter and cycle from 0 to 9
    displayNumber(counter);
    delay(1000); // Delay to prevent rapid increment
  }
  Serial.print("sayaç Degeri: ");
  Serial.println(counter);
  delay(300);// Time delay to see the numbers on the serial monitor
}
```

5.1.5.c.Counter using LDR (0-9)_opt_2

```
/**************************************************************************
**
Program Name: Counter using LDR (Light Dependent Resistor) (opt 21)

Program Objective:

This code can be used for a simple object counting and visual
display system. For example, it can be used to count pieces
on a conveyor belt or to keep track of the number of people
passing through a room.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
**************************************************************************
**/
// LED Display for numbers 0-9 using binary representation

const int ledPins[] = {10, 11, 12, 13}; // Define LED pins for display
const int numLeds = sizeof(ledPins) / sizeof(ledPins[0]);
const int ldrPin = A0; // Define LDR pin
int ldrValue = 0; // Variable to store the LDR value
int counter = 0;  // Counter variable

// Binary representation of numbers 0-9
const int numberPatterns[10] = {
  0b0000, // 0
  0b0001, // 1
  0b0010, // 2
  0b0011, // 3
  0b0100, // 4
  0b0101, // 5
  0b0110, // 6
  0b0111, // 7
  0b1000, // 8
  0b1001  // 9
};

void setup() {
  pinMode(ldrPin, INPUT); // Set LDR pin as input
  for (int i = 0; i < numLeds; i++) {
    pinMode(ledPins[i], OUTPUT); // Set LED pins as output
  }
  Serial.begin(9600); // Start serial communication
  delay(5000); // Initial delay for simulation setup
```

```
}

void displayNumber(int number) {
  if (number < 0 || number > 9) return; // Validate number range

  int pattern = numberPatterns[number]; // Get binary pattern for the number
  for (int i = 0; i < numLeds; i++) {
    digitalWrite(ledPins[i], (pattern >> i) & 1); // Display pattern on LEDs
  }
}

void loop() {
  ldrValue = analogRead(ldrPin); // Read value from LDR
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);

  if (ldrValue < 400) {
    counter = (counter + 1) % 10; // Increment and cycle counter
    displayNumber(counter); // Display current counter value
    delay(1000); // Debounce delay
  }

  Serial.print("Counter Value: ");
  Serial.println(counter);
  delay(300); // Delay for serial monitor readability
}
```

5.1.6.a.Turn on LEDs (lamps) as it gets darker using an LDR

https://www.tinkercad.com/things/4A11gFVpI0Q-516aturn-on-leds-lamps-as-it-gets-darker-using-an-ldr

```
/*********************************************************************
Program Name: Turn on LEDs (lamps) as it gets darker using an LDR

Program Purpose:

This code can be used to automatically illuminate LEDs (or lamps),
especially as ambient light decreases.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
*********************************************************************
***/
// Turn on LEDs (lamps) as it gets darker using an LDR

int ldrValue; // Variable to store the current value of the LDR

void setup() {
  Serial.begin(9600); // Start serial communication
  for(int led = 6; led < 14; led++) { // Iterate from pin 6 to 13
    pinMode(led, OUTPUT); // Set pins 6 to 13 as outputs for the LEDs
  }
}

void loop() {
  ldrValue = analogRead(A0); // Read the analog value from pin A0 and store it

  Serial.println(ldrValue); // Print the read value to the serial output

  for(int led = 6; led < 14; led++) { // Iterate from pin 6 to 13
    digitalWrite(led, LOW); // Turn off all LEDs
  }

  // Evaluate the read value

  if(ldrValue >= 120) { // If the value is equal to or greater than 120
    for(int led = 6; led < 14; led++) { // Iterate from pin 6 to 13
      digitalWrite(led, HIGH); // Turn on all LEDs
    }
  } else if(ldrValue >= 90 && ldrValue < 120) { // If the value is between 90 and 120
    for(int led = 6; led < 13; led++) { // Iterate from pin 6 to 12
      digitalWrite(led, HIGH); // Turn on LEDs up to pin 12
    }
  }
```

```
  // ... (Similar conditions for other value ranges)
  else if(ldrValue < 30) { // If the value is less than 30
    for(int led = 6; led < 14; led++) {
      digitalWrite(led, LOW); // Turn off all LEDs
    }
  }

  delay(500); // Delay of 500 ms
}
```

5.1.6.b.Optimized LED Display for ambient light adjustment using LDR

https://www.tinkercad.com/things/5CX9XDtk6dW-516boptimized-led-display-for-ambient-light-

```
/*********************************************************************
Program Name: Optimized LED Display for ambient light adjustment using LDR

Program Purpose:

This code can be used to automatically illuminate LEDs (or lamps),
especially as ambient light decreases.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
*********************************************************************
***/

const int ledPins[] = {6,7,8,9, 10, 11, 12, 13};
const int numLeds = sizeof(ledPins) / sizeof(ledPins[0]);
const int ldrPin = A0;
int ldrValue = 0;
const int maxLdrValue = 120; // Maximum threshold for LDR value

void setup() {
  pinMode(ldrPin, INPUT);
  for (int i = 0; i < numLeds; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
  Serial.begin(9600);
}

void updateLeds(int numOfLedsOn) {
  for (int i = 0; i < numLeds; i++) {
    digitalWrite(ledPins[i], i < numOfLedsOn ? HIGH : LOW);
  }
}

void loop() {
  ldrValue = analogRead(ldrPin);
  Serial.println("LDR Value: " + String(ldrValue));

  int ledsToLight = map(ldrValue, 0, maxLdrValue, numLeds, 0);
  ledsToLight = constrain(ledsToLight, 0, numLeds);
  updateLeds(ledsToLight);

  delay(500);
}
```

5.1.7.a.RGB LED control with LDR

https://www.tinkercad.com/things/8eehFMDD0Em-517argb-led-control-with-ldr

```
/*************************************************************************
**
Program Name: RGB LED control with LDR

Objective of the Program:

This code can be used to create an ambient light-sensitive
RGB LED lighting system with an LDR sensor.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova Deneyap Tech Workshop / 2022
*************************************************************************
***/
//

int ldrPin = A0; // Analog pin for the LDR
int ldrValue = 0; // Variable to store the LDR value

// Setup the pins for analog output as outputs
void setup() {
  Serial.begin(9600); // Start serial communication
  pinMode(11, OUTPUT); // Set pin 11 for LED output
  pinMode(10, OUTPUT); // Set pin 10 for LED output
  pinMode(9, OUTPUT);  // Set pin 9 for LED output
}

void loop() {
  ldrValue = analogRead(ldrPin); // Read the analog value from LDR
  Serial.println(ldrValue); // Print the read value to the serial output

  // Evaluate the read value and control the RGB LED accordingly

  if (ldrValue >= 250) {
    // We light up the LED in one combination
    analogWrite(9, HIGH);  // Red component
    analogWrite(10, LOW);  // Green component
    analogWrite(11, LOW);  // Blue component
    delay(1); // Provide a 1 ms delay
  }
  else if (ldrValue >= 220 && ldrValue < 250) {
    // Light up in a different combination
    analogWrite(9, LOW);
    analogWrite(10, HIGH);
    analogWrite(11, LOW);
```

```
    delay(1);
  }
  // ... (Similar conditions for other value ranges)
  else if (ldrValue < 70) {
    // Turn off the LED if the value is below 70
    analogWrite(9, LOW);
    analogWrite(10, LOW);
    analogWrite(11, LOW);
    delay(1);
  }

  delay(10); // Delay of 10 ms for stability
}
```

5.1.7.b.showing colors from cold to hot

https://www.tinkercad.com/things/lVDUMs8d0IP-517bshowing-colors-from-cold-to-hot

```
/**************************************************************************
**
Program Name: RGB LED control with LDR showing colors
              from cold to hot (white to red)

Objective of the Program:

This code can be used to create an ambient light-sensitive
RGB LED lighting system with an LDR sensor.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova Deneyap Tech Workshop / 2022
**************************************************************************
***/
//

int ldrPin = A0; // Analog pin for the LDR
int ldrValue = 0; // Variable to store the LDR value

// RGB LED pins
int redPin = 11;
int greenPin = 10;
int bluePin = 9;

// Colors from white to red in RGB format
int colors[7][3] = {
  {255, 255, 255}, // White
  {128, 0, 128},   // Purple
  {0, 0, 255},     // Blue
  {0, 255, 0},     // Green
  {255, 255, 0},   // Yellow
  {255, 165, 0},   // Orange
  {255, 0, 0}      // Red
};

void setup() {
  pinMode(ldrPin, INPUT);
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  Serial.begin(9600);
}

void setColor(int colorIndex) {
```

```
  if (colorIndex >= 0 && colorIndex < 7) {
    analogWrite(redPin, colors[colorIndex][0]);
    analogWrite(greenPin, colors[colorIndex][1]);
    analogWrite(bluePin, colors[colorIndex][2]);
  }
}

void loop() {
  ldrValue = analogRead(ldrPin);
  Serial.println("LDR Value: " + String(ldrValue));

  // Map LDR values to color index and set the color
  int colorIndex = map(ldrValue, 0, 310, 6, 0);
  colorIndex = constrain(colorIndex, 0, 6);
  setColor(colorIndex);

  delay(500);
}
```

5.1.8.a.Night Light Control

```
/**************************************************************************
**
Program Name: Night Light Control

Program Objective:

This code can be used to automatically turn on and off a lamp,
especially in dark environments.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
//

int relayPin = 11; // Relay pin
int buttonPin = 2; // Button pin
int LDRPin = A0; // LDR (Light Dependent Resistor) sensor pin

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

void setup() {
  pinMode(relayPin, OUTPUT); // Set relay pin as output
  pinMode(buttonPin, INPUT); // Set button pin as input

  Serial.begin(9600); // Start serial communication
}

void loop() {
  lightValue = analogRead(LDRPin); // Read the value from LDR
  buttonState = digitalRead(buttonPin); // Read the state of the button

  // Check if button is pressed and light value is low (dark environment)
  if (buttonState == 1 && lightValue <= 200) {
    digitalWrite(relayPin, HIGH); // Turn on the relay (light)
  } else if (buttonState == 0 || lightValue > 200) {
    digitalWrite(relayPin, LOW); // Turn off the relay (light)
  }

  // Print the sensor values to the serial monitor
  Serial.println(lightValue);
  //Serial.println(buttonPin); // This line is unnecessary and commented out
  Serial.println(buttonState);
```

```
  Serial.println("_________");

  delay(5000); // Delay of 5000 ms (5 seconds)
}
```

5.1.8.b.Optimized Night Light Control

https://www.tinkercad.com/things/6liphsdkKzc-518boptimized-night-light-control

```
/**********************************************************************
**
Program Name: Optimized Night Light Control

Program Objective:

This code can be used to automatically turn on and off a lamp,
especially in dark environments.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**********************************************************************
***/


int relayPin = 11; // Relay pin
int buttonPin = 2; // Button pin
int LDRPin = A0; // LDR sensor pin

int lightValue = 0; // Variable to store the LDR value
int buttonState = 0; // Variable to store the button state
bool isLightOn = false; // Variable to track the light status

void setup() {
  pinMode(relayPin, OUTPUT);
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
}

void controlLight(bool turnOn) {
  digitalWrite(relayPin, turnOn ? HIGH : LOW);
  isLightOn = turnOn;
}

void loop() {
  lightValue = analogRead(LDRPin);
  buttonState = digitalRead(buttonPin);

  bool shouldTurnOn = buttonState == 1 && lightValue <= 200;
  if (shouldTurnOn != isLightOn) {
    controlLight(shouldTurnOn);
  }
```

```
  // Optional: Only print when there is a change
  if (shouldTurnOn != isLightOn || millis() % 5000 < 100) {
    Serial.print("LDR Value: ");
    Serial.println(lightValue);
    Serial.print("Button State: ");
    Serial.println(buttonState);
    Serial.println("_________");
  }
}
```

5.1.9.a.Adjusting blinking speed based on light intensity

https://www.tinkercad.com/things/52EESaU9bUK-519aadjusting-blinking-speed-based-on-light-intensity

```
/*********************************************************************
Program Name:  Adjusting blinking speed based on light intensity

Program Purpose:

This code can be used to control the blinking rate of
a series of LEDs depending on light intensity.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
*********************************************************************
***/

//

int leds[] = {6, 7, 8, 9, 10, 11, 12, 13}; // Array of LED pins
int speed; // Variable for blink speed
int ldrPin = A0; // LDR sensor pin
int ldrValue; // Variable to store the LDR value
int ratio; // Variable to store the mapped value

void setup() {
  // Set all the LED pins as outputs
  for (int i = 6; i < 14; i++) {
    pinMode(i, OUTPUT);
  }
  Serial.begin(9600); // Start serial communication
}

void loop() {
  ldrValue = analogRead(ldrPin); // Read the value from LDR
  ratio = map(ldrValue, 0, 1023, 0, 1000); // Map the LDR value to a usable range
  Serial.println(ratio); // Print the mapped value to the serial monitor

  // Loop to turn on each LED, wait, then turn it off
  for (int i = 6; i < 14; i++) {
    digitalWrite(i, HIGH); // Turn on LED
    delay(ratio); // Wait for 'ratio' milliseconds
    digitalWrite(i, LOW); // Turn off LED
  }

  // Loop to turn on each LED in reverse order, wait, then turn it off
  for (int i = 13; i > 5; i--) {
```

```
    digitalWrite(i, HIGH); // Turn on LED
    delay(ratio); // Wait for 'ratio' milliseconds
    digitalWrite(i, LOW); // Turn off LED
  }
}
```

5.1.9.b.Optimized Blinking Speed Adjustment Based on Light Intensity

https://www.tinkercad.com/things/3EuIYpCf7kV-519boptimized-blinking-speed-adjustment-based-on-light-

```
/*********************************************************************
Program Name:  Optimized Blinking Speed Adjustment Based on Light Intensity

Program Purpose:

This code can be used to control the blinking rate of
a series of LEDs depending on light intensity.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
*********************************************************************
***/


const int ledPins[] = {6, 7, 8, 9, 10, 11, 12, 13};
const int numLeds = sizeof(ledPins) / sizeof(ledPins[0]);
const int ldrPin = A0;
int ldrValue;
int blinkDelay;

void setup() {
  for (int i = 0; i < numLeds; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
  Serial.begin(9600);
}

void blinkLEDs(int delayTime) {
  for (int i = 0; i < numLeds; i++) {
    digitalWrite(ledPins[i], HIGH);
    delay(delayTime);
    digitalWrite(ledPins[i], LOW);
  }

  for (int i = numLeds - 1; i >= 0; i--) {
    digitalWrite(ledPins[i], HIGH);
    delay(delayTime);
    digitalWrite(ledPins[i], LOW);
  }
}

void loop() {
  ldrValue = analogRead(ldrPin);
```

```
  blinkDelay = map(ldrValue, 0, 1023, 0, 1000);

  blinkLEDs(blinkDelay);

  if (millis() % 5000 < 100) { // Print every 5 seconds
    Serial.println("LDR Value: " + String(ldrValue));
    Serial.println("Blink Delay: " + String(blinkDelay));
  }
}
```

5.1.10.a.Lamp that turns on when the door is closed and dark

https://www.tinkercad.com/things/jYITUS9f8ym-5110alamp-that-turns-on-when-the-door-is-closed-and-dark

```
/**************************************************************************
**
Program Name:  Lamp that turns on when the door is closed and dark

Program Objective:

When the door is closed and dark, the button is activated and turns the lamp on
or off.
If "the door is open" or "it is light" the button is disabled.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/

int lightValue = 0;
int doorPin = 3; // Pin for the door contact
int ledPin = 11; // Pin for the LED
int ldrPin = A0; // Analog pin for the LDR
int buttonPin = 2; // Pin for the button
bool ledStatus = false; // Status of the LED (on or off)
bool lastButtonState = HIGH; // Last state of the button
bool buttonPressed = false; // Track if the button has been pressed

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(doorPin, INPUT);
  pinMode(ldrPin, INPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop() {
  lightValue = analogRead(ldrPin);
  bool buttonState = digitalRead(buttonPin);
  Serial.println(lightValue);

  // Check for button press and release
  if (buttonState == LOW && lastButtonState == HIGH) {
    buttonPressed = true;
    lastButtonState = buttonState;
  } else if (buttonState == HIGH && lastButtonState == LOW) {
```

```
    buttonPressed = false;
    lastButtonState = buttonState;
  }

  // Change LED status only on button press
  if (buttonPressed && lightValue <= 180 && digitalRead(doorPin) == HIGH) {
    ledStatus = !ledStatus;
    digitalWrite(ledPin, ledStatus ? HIGH : LOW);
    buttonPressed = false; // Reset button press status
  }

  // Automatically turn off the LED if the conditions are not met
  if (lightValue > 180 || digitalRead(doorPin) == LOW) {
    digitalWrite(ledPin, LOW);
  }
}
```

5.1.11.a.Automatic Light Intensity Adjustment

https://www.tinkercad.com/things/6p93rkEfI9t-5111aautomatic-light-intensity-adjustment

```
/*********************************************************************************
**
Program Name:  Automatic Light Intensity Adjustment

Program Objective:

This code can be used to automatically adjust the brightness of
an LED depending on the ambient light intensity. This is especially
suitable for energy saving and situations where lighting needs to be
automatically adjusted according to the intensity of the ambient light.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

*********************************************************************************
***/
// Define constants for LDR and ADC settings
#define LDR_PIN 0 // Pin number where LDR is connected

int ledPin = 3; // Pin number where LED is connected
int readValue; // Variable to store the analog value read from LDR
int scaledValue; // Variable to store the scaled LED brightness value

void setup() {
  pinMode(ledPin, OUTPUT); // Set the LED pin as output
  Serial.begin(9600); // Start serial communication for debugging
}

void loop() {
  readValue = analogRead(LDR_PIN); // Read the analog value from LDR
  Serial.println(readValue); // Print the read value to the serial monitor

  // Check if the read value is below a certain threshold (here, 180)
  if (readValue < 180) {
    // Map the read value to a brightness value (1 to 255)
    scaledValue = map(readValue, 1, 180, 255, 1);
    Serial.println(scaledValue); // Print the scaled value for debugging
    analogWrite(ledPin, scaledValue); // Adjust the LED brightness
    delay(100); // Short delay for stability
  } else {
    // If the LDR value is above the threshold, turn off the LED
    analogWrite(ledPin, 0);
  }
}
```

5.1.11.b.Automatic Light Intensity Adjustment (opt)

https://www.tinkercad.com/things/jmKirEaMZuI-5111bautomatic-light-intensity-adjustment-opt

```
/**************************************************************************
**
Program Name:  Automatic Light Intensity Adjustment (opt)

Program Objective:

This code can be used to automatically adjust the brightness of
an LED depending on the ambient light intensity. This is especially
suitable for energy saving and situations where lighting needs to be
automatically adjusted according to the intensity of the ambient light.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
#define LDR_PIN 0
#define LED_PIN 3
#define LDR_THRESHOLD 180
#define MAX_BRIGHTNESS 255
#define MIN_BRIGHTNESS 1

void setup() {
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int ldrReading = analogRead(LDR_PIN);
  Serial.println(ldrReading);

  if (ldrReading < LDR_THRESHOLD) {
    int ledBrightness = map(ldrReading, MIN_BRIGHTNESS, LDR_THRESHOLD,
MAX_BRIGHTNESS, MIN_BRIGHTNESS);
    Serial.println(ledBrightness);
    analogWrite(LED_PIN, ledBrightness);
  } else {
    analogWrite(LED_PIN, 0);
  }
}
```

5.1.12.a.Door Open Alert

https://www.tinkercad.com/things/7tedyCiPW1A-5112adoor-open-alert

```
/**************************************************************************
**
Program Name:  Door Open Alert

Program Objective:

This code can be used to alert, for example, if a refrigerator
door is left open for longer than necessary.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/
//
int speakerPin = 3; // Define the pin number connected to the speaker
int ldrPin = A0; // Define the pin number connected to the LDR (Light Dependent Resistor)
int ledPin = 2; // Define the pin number connected to the LED
int readValue = 0; // Variable to store the value read from the LDR
int averageValue = 0; // Variable to store the average value for calibration
unsigned long time = 0; // Variable to keep track of time

void setup() {
  pinMode(speakerPin, OUTPUT); // Set the speaker pin as output
  pinMode(ldrPin, INPUT); // Set the LDR pin as input
  pinMode(ledPin, OUTPUT); // Set the LED pin as output

  // Calibrate by reading 10 different values from the LDR
  for (int i = 0; i < 10; i++) { // Loop 10 times for different readings
    averageValue += analogRead(ldrPin); // Read the value and add it to the total in
averageValue
  }
  averageValue /= 10; // Calculate the average by dividing the total by 10
}

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

  // Check if the read value is significantly higher than the average
  // This could indicate that the door is open and the light has increased the LDR value
  if (readValue > averageValue + 100) {
    delay(10); // Wait for 10 ms
    time += 10; // Increment the time variable by 10
```

```
    // If the time variable exceeds 10000 (10 seconds), assume the door has been open for too
long
    if (time >= 10000) {
      tone(speakerPin, 1000); // Play a sound on the speaker
      digitalWrite(ledPin, HIGH); // Turn on the LED
      delay(250); // Wait for 250 ms
      digitalWrite(ledPin, LOW); // Turn off the LED
      noTone(speakerPin); // Stop the sound on the speaker
      delay(250); // Wait for 250 ms
    }
  } else { // If the door is closed or the LDR reading is back to normal
    time = 0; // Reset the time variable
    noTone(speakerPin); // Stop the sound on the speaker
    digitalWrite(ledPin, LOW); // Turn off the LED
  }
}
```

5.1.12.b.Door Open Alert (opt)

https://www.tinkercad.com/things/glF74mpblGf-5112bdoor-open-alert-opt

```
/*****************************************************************************
**
Program Name:  Door Open Alert

Program Objective:

This code can be used to alert, for example, if a refrigerator
door is left open for longer than necessary.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

*****************************************************************************
***/
#define LDR_PIN 0 // Pin number for the LDR (Light Dependent Resistor)
#define LED_PIN 2 // Pin number for the LED
#define SPEAKER_PIN 3 // Pin number for the speaker
#define CALIBRATION_READS 10 // Number of readings to take for calibration
#define THRESHOLD_OFFSET 100 // Offset value for LDR threshold
#define ALARM_DELAY 300 // Delay before the alarm is triggered in milliseconds

int ldrAverage = 0; // Variable to store the average LDR reading
unsigned long alarmStartTime = 0; // Variable to track when the alarm starts

void setup() {
  pinMode(LED_PIN, OUTPUT); // Set LED pin as an output
  pinMode(SPEAKER_PIN, OUTPUT); // Set speaker pin as an output
  pinMode(LDR_PIN, INPUT); // Set LDR pin as an input
  Serial.begin(9600); // Start serial communication
  ldrAverage = calculateLdrAverage(); // Calculate the average LDR value
}

void loop() {
  int ldrReading = analogRead(LDR_PIN); // Read the current value from the LDR
  Serial.println(ldrReading); // Print the LDR value to the serial monitor

  // Check if the LDR reading is above the threshold
  if (ldrReading > ldrAverage + THRESHOLD_OFFSET) {
    triggerAlarm(); // Trigger the alarm if conditions are met
  } else {
    resetAlarm(); // Reset the alarm if conditions are not met
  }
}
```

```
// Function to calculate the average LDR reading
int calculateLdrAverage() {
  int total = 0;
  for (int i = 0; i < CALIBRATION_READS; i++) {
    total += analogRead(LDR_PIN); // Sum the LDR readings
  }
  return total / CALIBRATION_READS; // Return the average reading
}

// Function to trigger the alarm
void triggerAlarm() {
  // Start the timer if it's not already started
  if (alarmStartTime == 0) {
    alarmStartTime = millis(); // Record the start time
  }

  // Check if the alarm delay has passed
  if (millis() - alarmStartTime > ALARM_DELAY) {
    tone(SPEAKER_PIN, 1000, 250); // Play a tone on the speaker
    digitalWrite(LED_PIN, HIGH); // Turn on the LED
    delay(250); // Wait for a short period
    digitalWrite(LED_PIN, LOW); // Turn off the LED
  }
}

// Function to reset the alarm
void resetAlarm() {
  alarmStartTime = 0; // Reset the start time
  noTone(SPEAKER_PIN); // Stop any ongoing tone
  digitalWrite(LED_PIN, LOW); // Turn off the LED
}
```

5.1.13.a.Color Change Based on Ambient Light Level

https://www.tinkercad.com/things/h18WR8IA5Tw-5113bcolor-change-based-on-ambient-light-level-opt

```
/*************************************************************************
**
Program Name:  Color Change Based on Ambient Light Level

 (OPT)

Program Objective:

This code can be used to light LEDs of different colors
depending on the ambient light level using an LDR, as well as
output audio tones from the speaker depending on the light level.
The color of the LEDs changes depending on the light level, and
the tone coming out of the speaker also changes depending on
the light level. This can be used as an ambient light sensor or
in an artistic project, for example.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

*************************************************************************
***/

// Color Change Based on Ambient Light Level

#define LDR_PIN A0 // Analog pin connected to the LDR
#define RED_LED_PIN 6 // Pin number for the red LED
#define GREEN_LED_PIN 5 // Pin number for the green LED
#define BLUE_LED_PIN 3 // Pin number for the blue LED
#define SPEAKER_PIN 8 // Pin number for the speaker

// Threshold values for light levels
#define LIGHT_LEVEL_1 300
#define LIGHT_LEVEL_2 400
#define LIGHT_LEVEL_3 600
#define MAX_LIGHT_LEVEL 1024

void setup() {
  // Initialize LED and speaker pins as outputs
  pinMode(RED_LED_PIN, OUTPUT);
  pinMode(GREEN_LED_PIN, OUTPUT);
  pinMode(BLUE_LED_PIN, OUTPUT);
  pinMode(SPEAKER_PIN, OUTPUT);
  Serial.begin(9600); // Start serial communication
```

```
}

void loop() {
  int lightValue = analogRead(LDR_PIN); // Read light level from LDR
  Serial.println(lightValue); // Print light value to serial monitor
  controlLEDsAndSound(lightValue); // Control LEDs and sound based on light level
  delay(100); // Short delay for stability
}

// Function to control LEDs and sound based on light level
void controlLEDsAndSound(int lightValue) {
  if (lightValue < LIGHT_LEVEL_1) {
    activateAllLEDs(); // Activate all LEDs in very low light
    tone(SPEAKER_PIN, lightValue); // Generate tone based on light value
  } else if (lightValue < LIGHT_LEVEL_2) {
    activateSingleLED(RED_LED_PIN); // Activate only red LED
    tone(SPEAKER_PIN, lightValue);
  } else if (lightValue < LIGHT_LEVEL_3) {
    activateSingleLED(BLUE_LED_PIN); // Activate only blue LED
    tone(SPEAKER_PIN, lightValue);
  } else if (lightValue < MAX_LIGHT_LEVEL) {
    activateSingleLED(GREEN_LED_PIN); // Activate only green LED
    tone(SPEAKER_PIN, lightValue);
  } else {
    noTone(SPEAKER_PIN); // No tone if light level is very high
    deactivateAllLEDs(); // Deactivate all LEDs
  }
}

// Function to activate all LEDs
void activateAllLEDs() {
  digitalWrite(RED_LED_PIN, LOW);
  digitalWrite(GREEN_LED_PIN, LOW);
  digitalWrite(BLUE_LED_PIN, LOW);
}

// Function to deactivate all LEDs
void deactivateAllLEDs() {
  digitalWrite(RED_LED_PIN, HIGH);
  digitalWrite(GREEN_LED_PIN, HIGH);
  digitalWrite(BLUE_LED_PIN, HIGH);
}

// Function to activate a single LED and deactivate others
void activateSingleLED(int ledPin) {
  deactivateAllLEDs(); // First, turn off all LEDs
  digitalWrite(ledPin, LOW); // Then, turn on the specified LED
}
```

5.1.14.a.Lighted Signboard

https://www.tinkercad.com/things/1tSeMG1oFbl-5114alighted-signboard-

```
/*************************************************************************
Program Name:  Lighted Signboard

Program Purpose:

This code controls the LED array based on the light level of
an LDR. Above a certain light level, the LEDs turn on and off
sequentially and in a specific pattern. This can be used as
an eye-catching illuminated sign or a visual warning system.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
*************************************************************************
***/
// Lighted Signboard

int leds[] = {4, 5, 6, 7, 8, 9, 10, 11, 12, 13}; // Define the pin numbers connected to the LEDs
int ldrPin = A5; // Define the pin number connected to the LDR (Light Dependent Resistor)
int speed = 200; // Variable for controlling the speed of LED animation, can be adjusted as
desired

void setup() {
  Serial.begin(9600);
  // Set the LED pins as outputs using a for loop
  for (int i = 0; i < 10; i++) {
    pinMode(leds[i], OUTPUT);
  }
  pinMode(ldrPin, INPUT_PULLUP); // Set the LDR pin as input and enable the pull-up
resistor
}

void loop() {
  // Check the state of the LDR
  int ldrValue = analogRead(ldrPin);
  Serial.println(ldrValue);
  if (ldrValue > 600) { // If light intensity value is greater than 600
    // Sequentially light up the LEDs
    for (int i = 2; i < 10; i++) {
      digitalWrite(leds[i], HIGH); // Turn on the LED at leds[i]
      delay(speed); // Wait for the duration specified by the speed variable
    }

    digitalWrite(leds[0], HIGH); // Turn on the LED at leds[0]
    delay(speed);
```

```
    digitalWrite(leds[1], HIGH); // Turn on the LED at leds[1]
    delay(speed);

    // Turn off all LEDs
    for (int l = 0; l < 10; l++) {
      digitalWrite(leds[l], LOW); // Turn off the LED at leds[l]
    }
    delay(speed);

    // Repeat the pattern 3 times
    for (int i = 0; i < 3; i++) {
      // Turn on all LEDs
      for (int l = 0; l < 10; l++) {
        digitalWrite(leds[l], HIGH);
      }
      delay(speed);

      // Turn off all LEDs
      for (int l = 0; l < 10; l++) {
        digitalWrite(leds[l], LOW);
      }
      delay(speed);
    }
  }
}
```

5.1.14.b.Lighted Signboard (opt)

https://www.tinkercad.com/things/50UT1zUmS4j-5114blighted-signboard-opt

```
/**************************************************************************
Program Name:  Lighted Signboard (opt)

Program Purpose:

This code controls the LED array based on the light level of
an LDR. Above a certain light level, the LEDs turn on and off
sequentially and in a specific pattern. This can be used as
an eye-catching illuminated sign or a visual warning system.

Written by: Kamil Bala
kamilbala42@gmail.com
tw: @tek_elo
Yalova / 2023
**************************************************************************
***/
// Lighted Signboard Animation Based on Ambient Light

#define LDR_PIN A5 // Define the analog pin connected to the LDR
#define LED_COUNT 10 // Number of LEDs in the array
#define LIGHT_THRESHOLD 600 // Light level threshold for activating LEDs
#define ANIMATION_SPEED 200 // Speed of the LED animation

// Array of LED pin numbers
int leds[] = {4, 5, 6, 7, 8, 9, 10, 11, 12, 13};

void setup() {
  Serial.begin(9600); // Start serial communication for debugging
  // Set all LED pins as outputs
  for (int i = 0; i < LED_COUNT; i++) {
    pinMode(leds[i], OUTPUT);
  }
  pinMode(LDR_PIN, INPUT_PULLUP); // Set LDR pin as input with pull-up resistor
}

void loop() {
  int ldrValue = analogRead(LDR_PIN); // Read the light value from the LDR
  Serial.println(ldrValue); // Print the LDR value to the serial monitor

  // Check if the light level exceeds the threshold
  if (ldrValue > LIGHT_THRESHOLD) {
    animateLEDs(); // Animate LEDs if light level is high enough
  }
}

// Function to animate LEDs
```

```
void animateLEDs() {
  // Sequentially light up the LEDs
  for (int i = 2; i < LED_COUNT; i++) {
    turnOnLED(i); // Turn on each LED one by one
    delay(ANIMATION_SPEED); // Delay to control the speed of animation
  }

  // Turn off all LEDs and repeat the pattern 3 times
  for (int j = 0; j < 3; j++) {
    turnOffAllLEDs(); // Turn off all LEDs
    delay(ANIMATION_SPEED); // Delay
    turnOnAllLEDs(); // Turn on all LEDs
    delay(ANIMATION_SPEED); // Delay
  }
  turnOffAllLEDs(); // Turn off all LEDs at the end
}

// Function to turn on a specific LED
void turnOnLED(int ledIndex) {
  digitalWrite(leds[ledIndex], HIGH); // Turn on the LED at the specified index
}

// Function to turn off all LEDs
void turnOffAllLEDs() {
  for (int i = 0; i < LED_COUNT; i++) {
    digitalWrite(leds[i], LOW); // Turn off each LED in the array
  }
}

// Function to turn on all LEDs
void turnOnAllLEDs() {
  for (int i = 0; i < LED_COUNT; i++) {
    digitalWrite(leds[i], HIGH); // Turn on each LED in the array
  }
}
```

5.1.15.a.LDR Barrier Control

https://www.tinkercad.com/things/2aW3AscCl5F-5115aldr-barrier-control

```
/**************************************************************************
**
Program Name: LDR Barrier Control

Program Objective:

This code provides an automatic barrier control system using
an LDR and servo motor. When the light intensity read
by the LDR is above a certain threshold, the barrier
(controlled by the servo motor) is automatically raised.
If the LDR detects a new vehicle within a certain period of
time after the barrier is removed, the barrier closing process
is postponed. This functionality can be especially useful
for parking lots or controlled access points.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/

// LDR Barrier Control

#include <Servo.h> // Include the Servo library

int ldrPin = A0; // Define the pin number connected to the LDR
int servoPin = 9; // Define the pin number connected to the servo motor
int barrierStatus = 0; // Variable to keep track of the barrier status
int servoStep = 90; // Variable for servo motor step size
int angleValue = 90; // Variable to store the current angle of the servo
int triggerStatus = 0; // Variable to check if the motion has been triggered
Servo servo; // Create a servo object

void setup() {
  pinMode(ldrPin, INPUT); // Set the LDR pin as input
  servo.attach(servoPin); // Attach the servo motor to its pin

  servo.write(90); // Set the servo to 90 degrees position
}

void loop() {
  // Check if the LDR reading is above a certain threshold
  if (analogRead(ldrPin) > 900) {
    barrierStatus = 1; // Set barrier status as open
```

```
  }

  // If barrier status is open
  if (barrierStatus == 1) {
    // Lift the barrier by rotating the servo motor to 90 degrees
    for (int i = 0; i < angleValue; i++) {
      servoStep += 1; // Rotate the servo motor by one degree at a time
      servo.write(servoStep); // Set the servo angle according to servoStep
      delay(25); // Wait for 25ms
    }

    // Wait for 5 seconds and check if a new car has arrived
    for (int j = 0; j < 100; j++) {
      delay(50); // Wait for 50ms
      // Check for a new car
      if (analogRead(ldrPin) > 900) {
        triggerStatus = 1; // Set trigger status to indicate barrier closure
        j = 100; // Exit the loop
        delay(200); // Wait for 200ms
        break; // Exit the for loop
      }
    }

    // Lower the barrier by rotating the servo motor back to its original position
    for (int i = 0; i < 90; i++) {
      servoStep -= 1; // Rotate the servo motor back by one degree at a time
      servo.write(servoStep); // Set the servo angle according to servoStep
      delay(25); // Wait for 25ms
      // Check if a new car has arrived during the lowering process
      if (analogRead(ldrPin) > 900) {
        angleValue = 180 - servoStep; // Continue from the current position
        i = 90; // Exit the loop
        delay(200); // Wait for 200ms
        break; // Exit the for loop
      } else {
        triggerStatus = 0; // Reset the control variable
        angleValue = 90; // Reset the angle value
      }
      if (servoStep == 90)
        barrierStatus = 0; // Set barrier status as closed
    }
  }
}
```

5.1.15.b.LDR Barrier Control (opt)

https://www.tinkercad.com/things/lj8R0ZOGZ6I-5115bldr-barrier-control-opt

```
/**************************************************************************
**
Program Name: LDR Barrier Control (opt)

Program Objective:

This code provides an automatic barrier control system using
an LDR and servo motor. When the light intensity read
by the LDR is above a certain threshold, the barrier
(controlled by the servo motor) is automatically raised.
If the LDR detects a new vehicle within a certain period of
time after the barrier is removed, the barrier closing process
is postponed. This functionality can be especially useful
for parking lots or controlled access points.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/

// LDR Barrier Control

#include <Servo.h>

// Constants
#define LDR_PIN A0 // Analog pin for the LDR
#define SERVO_PIN 9 // Digital pin for the servo motor
#define LIGHT_THRESHOLD 900 // Light level threshold for activating the barrier
#define INITIAL_SERVO_ANGLE 90 // Initial angle of the servo
#define SERVO_MOVEMENT_DELAY 25 // Delay for servo movement
#define SERVO_FULL_ROTATION_DELAY 200 // Delay for a full rotation of the servo
#define CHECK_INTERVAL 50 // Interval for checking the LDR value
#define WAIT_TIME 5000 // Total time to wait for a car to pass

// Global variables
Servo servo;
int barrierStatus = 0;
int currentServoAngle = INITIAL_SERVO_ANGLE;

void setup() {
  pinMode(LDR_PIN, INPUT); // Set LDR pin as input
  servo.attach(SERVO_PIN); // Attach servo to its pin
  servo.write(INITIAL_SERVO_ANGLE); // Initialize servo to its starting position
```

```
}

void loop() {
  // Check the light level using LDR
  if (analogRead(LDR_PIN) > LIGHT_THRESHOLD) {
    barrierStatus = 1; // Set barrier status as active
  }

  // If barrier status is active, operate the barrier
  if (barrierStatus == 1) {
    raiseBarrier(); // Raise the barrier
    waitForCar(); // Wait for the car to pass
    lowerBarrier(); // Lower the barrier
    barrierStatus = 0; // Reset barrier status
  }
}

// Function to raise the barrier
void raiseBarrier() {
  moveServo(180); // Move servo to 180 degrees
}

// Function to lower the barrier
void lowerBarrier() {
  moveServo(INITIAL_SERVO_ANGLE); // Move servo back to its initial angle
}

// Function to move the servo to a specified angle
void moveServo(int targetAngle) {
  // Gradually move the servo to the target angle
  while (currentServoAngle != targetAngle) {
    currentServoAngle += (targetAngle > currentServoAngle) ? 1 : -1;
    servo.write(currentServoAngle); // Update the servo position
    delay(SERVO_MOVEMENT_DELAY); // Delay to control the speed of movement
  }
}

// Function to wait for a car to pass
void waitForCar() {
  int waitTime = 0;
  // Wait for a specified time or until a car is detected again
  while (waitTime < WAIT_TIME) {
    delay(CHECK_INTERVAL); // Short delay between checks
    waitTime += CHECK_INTERVAL;
    // If a car is detected, reset the wait time
    if (analogRead(LDR_PIN) > LIGHT_THRESHOLD) {
      waitTime = 0;
    }
  }
}
```

5.1.16.a.Observing LDR Measurement on an I2C LCD

https://www.tinkercad.com/things/lIsB8aGrXet-5116aobserving-ldr-measurement-on-an-i2c-lcd

```
/**********************************************************************
**
Program Name:  Observing LDR Measurement on an I2C LCD

Program Objective:

This code displays the reading of an LDR (Light Dependent
Resistor) on a connected LCD screen using the I2C protocol.
The analog value from the LDR is read continuously and
displayed on the second line of the LCD. This app is useful
for visually monitoring light levels and is ideal for
laboratory testing, hobby projects, or educational applications.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**********************************************************************
***/

#include <Adafruit_LiquidCrystal.h> // Include the Adafruit LiquidCrystal library for LCD
Adafruit_LiquidCrystal lcd(32); // Initialize the LCD on I2C address 32
#define LDR_PIN A0 // Define the analog pin for LDR
unsigned long previousMillis = 0; // Variable to store the last time the LDR was read

void setup() {
  previousMillis = millis(); // Initialize the previousMillis variable with the current time
  lcd.begin(16, 2); // Initialize the LCD with 16 columns and 2 rows
  pinMode(LDR_PIN, INPUT); // Set the LDR pin as an input
  lcd.print("LDR read is : "); // Print a message on the LCD
}

void loop() {
  while (true) { // Create an infinite loop
    // Check if more than 1 millisecond has passed
    if (millis() - previousMillis > 1) {
      lcd.setCursor(0, 1); // Set the cursor to the second row
      lcd.print(analogRead(LDR_PIN)); // Read the value from the LDR and print it on the
LCD
      previousMillis = millis(); // Update the time we last read the LDR
    }
  }
}
```

5.1.16.b.Observing LDR Measurement on an I2C LCD (opt)

https://www.tinkercad.com/things/3vAX77wl4ma-5116bobserving-ldr-measurement-on-an-i2c-lcd-opt

```
/**************************************************************************
**
Program Name:  Observing LDR Measurement on an I2C LCD
(opt)
Program Objective:

This code displays the reading of an LDR (Light Dependent
Resistor) on a connected LCD screen using the I2C protocol.
The analog value from the LDR is read continuously and
displayed on the second line of the LCD. This app is useful
for visually monitoring light levels and is ideal for
laboratory testing, hobby projects, or educational applications.

Yazan: Kamil Bala
      kamilbala42@gmail.com
      tw: @tek_elo
      Yalova / 2023

**************************************************************************
***/

#include <Adafruit_LiquidCrystal.h>

Adafruit_LiquidCrystal lcd(32); // Initialize the LCD on I2C address 32
#define LDR_PIN A0 // Define the analog pin for LDR
#define READ_INTERVAL 1000 // Set the interval for reading LDR (1 second)

unsigned long previousMillis = 0; // Variable to store the last time the LDR was read
int lastLdrValue = -1; // Variable to store the last LDR value

void setup() {
  lcd.begin(16, 2); // Initialize the LCD with 16 columns and 2 rows
  pinMode(LDR_PIN, INPUT); // Set the LDR pin as an input
  lcd.print("LDR read is : "); // Print a message on the LCD
}

void loop() {
  unsigned long currentMillis = millis();

  // Read and display the LDR value at defined intervals
  if (currentMillis - previousMillis >= READ_INTERVAL) {
    int ldrValue = analogRead(LDR_PIN); // Read the value from the LDR
    if (ldrValue != lastLdrValue) { // Update the LCD only if the value has changed
      lcd.setCursor(0, 1); // Set the cursor to the second row
      lcd.print("               "); // Clear the previous reading
```

```
      lcd.setCursor(0, 1); // Reset the cursor position
      lcd.print(ldrValue); // Display the new LDR value
      lastLdrValue = ldrValue; // Update the last LDR value
    }
    previousMillis = currentMillis; // Update the time we last read the LDR
  }
}
```