Project Title: Soil Moisture Level Detection using Arduino.

Introduction:

Effective water management is essential for resource conservation and good plant growth in contemporary gardening and agriculture. A crucial component of sustainable farming practices is soil moisture monitoring because excessive or insufficient irrigation can have a negative impact on plant health and productivity. This project aims to create a straightforward yet efficient soil moisture monitoring system that uses an analog soil moisture sensor, an Arduino microcontroller, and a number of LEDs to show the soil's moisture content graphically. Users may make educated judgments about watering their plants using the system's real-time feedback through LED indicators and continuous soil moisture level measurement.

Materials:

The soil moisture monitoring system is designed around the following core components:

1. Arduino Microcontroller: As the system's brain, it processes sensor data and manages the LEDs.
2. Soil Moisture Sensor: Determines the soil's volumetric water content by monitoring variations in electrical resistance. The sensor outputs an analog signal that varies with moisture levels.
3. LED Indicators: Give consumers an immediate understanding of the soil moisture level by visually representing it.

Theory:

Usually, the soil moisture sensor utilized for this project is made up of two soil-inserted probes. The probes measure the electrical resistance between them, which varies with the water content in the soil. Soil resistivity is higher in dry conditions and lower in wet conditions. The analog input pin on the Arduino receives an analog voltage proportionate to the moisture level output by the sensor.

The Arduino reads the sensor value, converts it into a percentage representing the soil moisture level, and then controls the LEDs based on predefined moisture thresholds. Here is a thorough explanation of the procedure:

1. **Analog Read and Conversion:**

   The sensor outputs an analog signal with a range of 0 to 1023 (10-bit ADC resolution of the Arduino). This sensor value is mapped by the project code to a moisture percentage using the following formula:

   $$\text{moisturePercentage} = \frac{(1028 - moistureValue)}{10}$$

   With a value of 1028 for completely dry soil and values closer to 0 for extremely moist soil, this formula inversely scales the sensor value.

As an example, Let’s say,

$$moistureValue = 500$$

First, the expression, $(1028 - moistureValue)$ will be calculated.

1028−500=528

Then, the result is divided by 10,

528/10.0=52.8

So, Moisture Percentage is set to 52.8, indicating that the soil moisture level is approximately 52.8%.

2. **LED Indication Logic:**

   - The system uses four LEDs to represent different moisture levels:

     ○LED1 (D2): Indicates 15% and 25% moisture percentage.
     ○LED2 (D3): Indicates 25% and 50% moisture percentage.
     ○LED3 (D6): Indicates 50% and 75% moisture percentage.
     ○LED4 (D7): Indicates moisture percentage above 75%.

   - The LEDs provide a fast visual clue regarding the state of the soil by gradually lighting up in response to the moisture percentage.

3. **System Operation:**

   Arduino uses the loop function to constantly read the moisture sensor data, determine the moisture percentage, and set the relevant LEDs to HIGH or LOW. Users can decide if the soil has enough moisture or needs to be watered by looking at this visual cue.

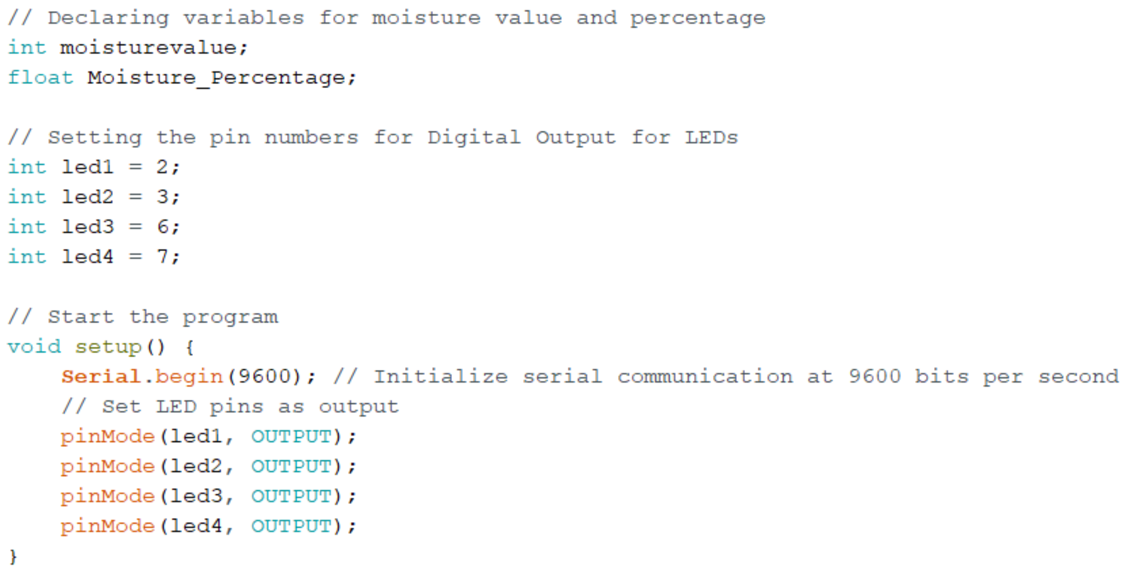
Benefits And Applications:

The soil moisture monitoring system is beneficial for:

1. **Farmers:** Improves agriculture production and water saving through increasing irrigation efficiency.
2. **Greenhouses:** Helps to maintain ideal growth conditions by automating moisture monitoring.
3. **Home Gardeners:** Makes watering plants easier and reduces plant stress brought on by incorrect watering.

Sustainable water management practices are essential for modern agriculture and horticulture, and this technology supports them by offering real-time soil moisture data and visually clear feedback.

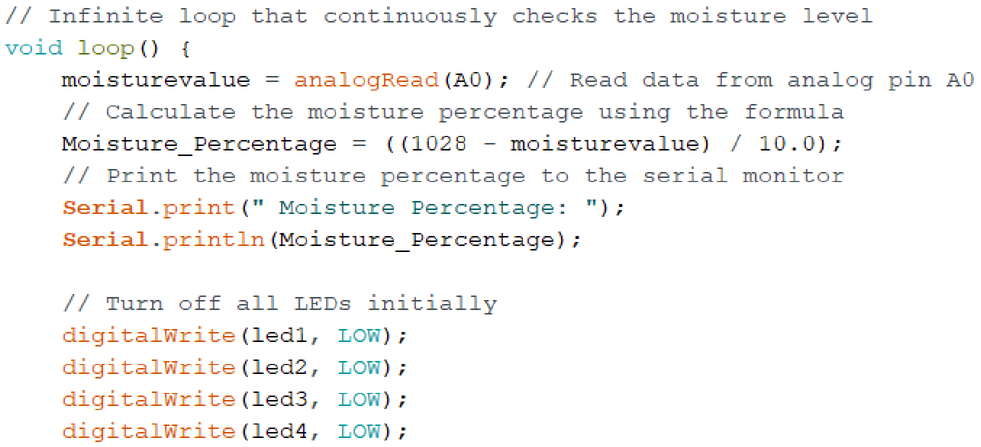
Arduino Code:



```
// Declaring variables for moisture value and percentage
int moisturevalue;
float Moisture_Percentage;

// Setting the pin numbers for Digital Output for LEDs
int led1 = 2;
int led2 = 3;
int led3 = 6;
int led4 = 7;

// Start the program
void setup() {
    Serial.begin(9600); // Initialize serial communication at 9600 bits per second
    // Set LED pins as output
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
    pinMode(led4, OUTPUT);
}
```



```
// Infinite loop that continuously checks the moisture level
void loop() {
    moisturevalue = analogRead(A0); // Read data from analog pin A0
    // Calculate the moisture percentage using the formula
    Moisture_Percentage = ((1028 - moisturevalue) / 10.0);
    // Print the moisture percentage to the serial monitor
    Serial.print(" Moisture Percentage: ");
    Serial.println(Moisture_Percentage);

    // Turn off all LEDs initially
    digitalWrite(led1, LOW);
    digitalWrite(led2, LOW);
    digitalWrite(led3, LOW);
    digitalWrite(led4, LOW);
    // Decide which LEDs to turn on based on the moisture percentage
    if (Moisture_Percentage >= 15 && Moisture_Percentage < 25) {
        // Turn on the first LED
        digitalWrite(led1, HIGH);
    } else if (Moisture_Percentage >= 25 && Moisture_Percentage < 50) {
        // Turn on the first two LEDs
        digitalWrite(led1, HIGH);
        digitalWrite(led2, HIGH);
    } else if (Moisture_Percentage >= 50 && Moisture_Percentage < 75) {
        // Turn on the first three LEDs
        digitalWrite(led1, HIGH);
        digitalWrite(led2, HIGH);
        digitalWrite(led3, HIGH);
    } else if (Moisture_Percentage >= 75) {
        // Turn on all four LEDs
        digitalWrite(led1, HIGH);
        digitalWrite(led2, HIGH);
        digitalWrite(led3, HIGH);
        digitalWrite(led4, HIGH);
    } else {
        // Ensure all LEDs are off (redundant but clear)
        digitalWrite(led1, LOW);
        digitalWrite(led2, LOW);
        digitalWrite(led3, LOW);
        digitalWrite(led4, LOW);
    }

    delay(1000); // Delay for 1 second before the next reading
}
```