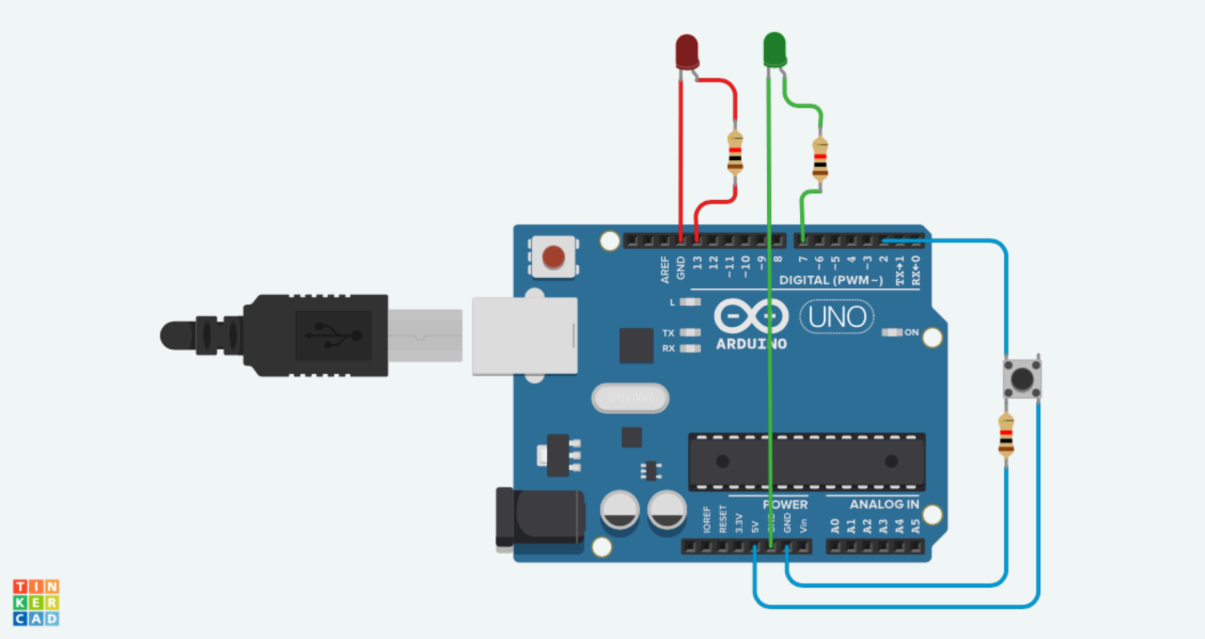
EXERCISE 1: SET-UP

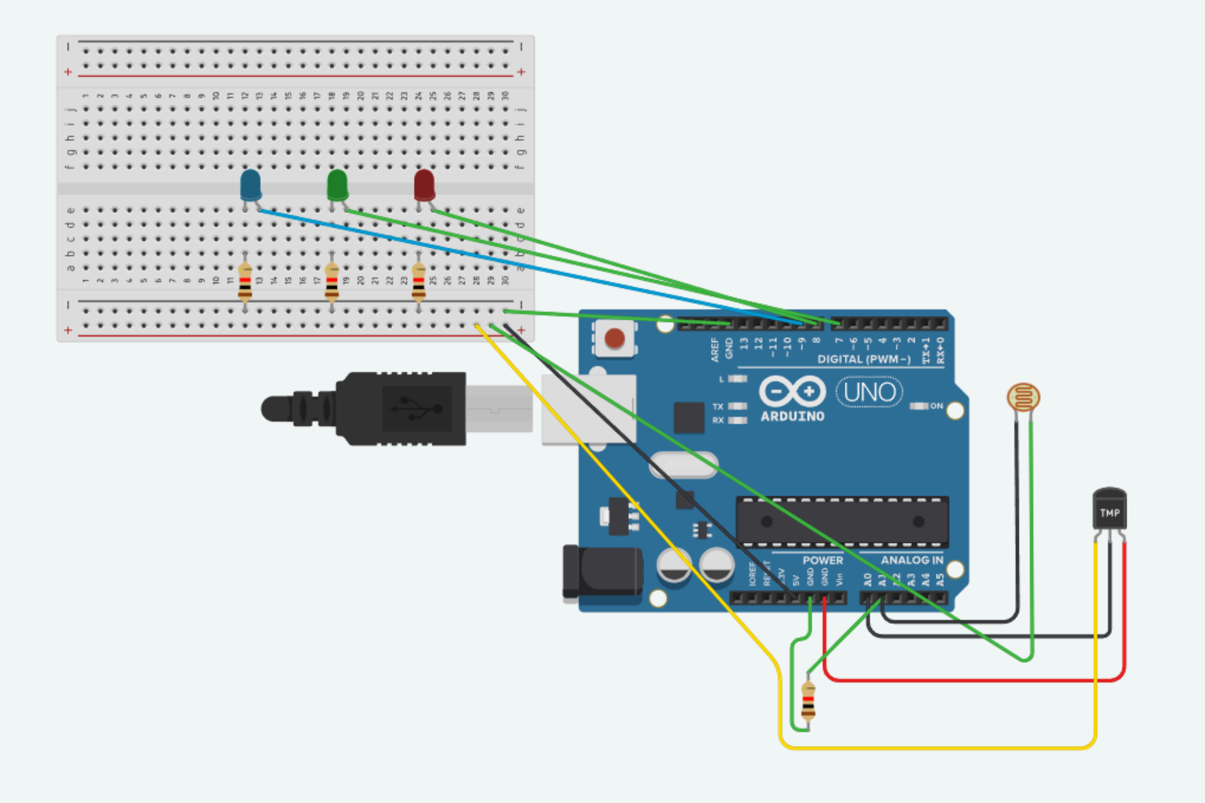


Link to board

Code:

```
// (C) Jitish Rajankumar Padhya, Raghav Tengse, Utkarsh Singh,
group: 23 (2024)
// Work package 3
// Exercise 1:
// Submission code: 2345166


//defining the pins connected to the respective components
int redLED = 13;// red led is connected to the 13th digital pin
int buttonState = 0; //button is at 0, meaning its at normal pos
int pushButton = 2;//red led is connected to the 2nd digital pin
int greenLED = 7;// green led is connected to the 7th digital pin
void setup(){
  //define and initialise the components used in the circuit
  // i.e red LED light, push button and a green LED light
  //the pinmode also defines the type,i.e red light provides output
  //and the button takes input(i.e getting pressed
  pinMode(redLED, OUTPUT);
  pinMode(pushButton, INPUT);
  pinMode(greenLED, OUTPUT);
}
void loop(){
  //reads the state of the button(pushed or not pushed)
  buttonState = digitalRead(pushButton);
  //if buttonState is high, it indicates that the button is pressed
  if (buttonState == HIGH) {
    //this in turn switches the green light on
    digitalWrite(greenLED, HIGH);
  }else {
    //else it switches the green light off
    digitalWrite(greenLED, LOW);
  }
//the red light will switch on
  digitalWrite(redLED, HIGH);
  //delay of 500 ms
  delay(500);
  //red light switch off
  digitalWrite(redLED, LOW);
  //delay of 500 ms
  delay(500);
  //these delays cause the red light to flash at intervals
}
```

EXERCISE 2: TEMPERATURE AND LIGHT METER

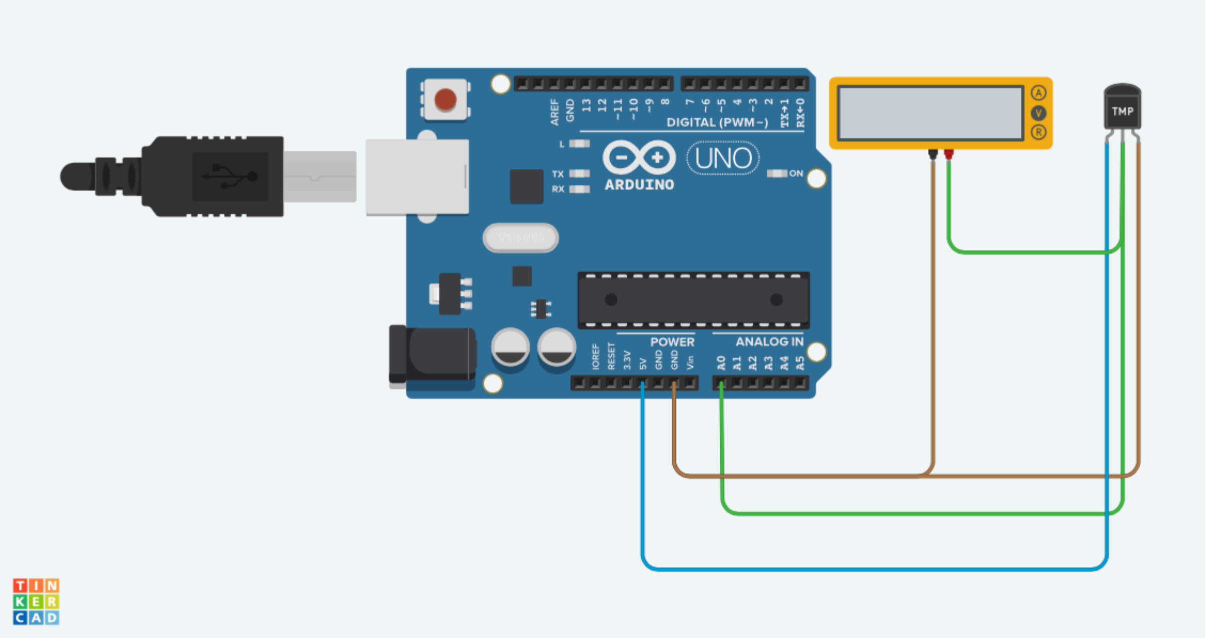


[Link to board]

```
// (C) Jitish Rajankumar Padhya, Raghav Tengse, Utkarsh Singh,
group: 23 (2024)
// Work package 3
// Exercise 3.1
// Submission code: 2345166
const int temperaturePin = A0; // Analog pin for temperature sensor
const int lightPin = A1;       // Analog pin for light sensor
const int greenLedPin = 7;     // Pin for green LED
const int redLedPin = 8;       // Pin for red LED
const int blueLedPin = 9;      // Pin for blue LED
//temperature thresholds as per the requirement WP3
const int temperatureThresholds[4] = {-12, 0, 20, 21};
//light intensity thresholds as per the requirement WP3
const int lightThresholds[4] = {0, 20, 60, 100};

void setup() {
  //Configures the pins connected to LEDs as output pins.
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  pinMode(blueLedPin, OUTPUT);

  Serial.begin(9600);
}

void loop() {
  //variables to store the temperature
  int temperature = readTemperature();
  //variable to store the light intensity
  int lightIntensity = readLightIntensity();

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.print("C, Light Intensity: ");
  Serial.print(lightIntensity);
  Serial.println("%");
  //variable to store the threshold index of temperature
  int tempIndex = getThresholdIndex(temperature,
temperatureThresholds, 4);
  //variable to store the threshold index of light intensity
  int lightIndex = getThresholdIndex(lightIntensity,
lightThresholds, 4);

  Serial.print("Temperature Index: ");
  Serial.print(tempIndex);
  Serial.print(", Light Index: ");
  Serial.println(lightIndex);
  //if light and threshold have the same intensity.
  // This indicates normal dependencies. So, the green light is set
to high.
  if (tempIndex == lightIndex) {
    digitalWrite(greenLedPin, HIGH);
    digitalWrite(redLedPin, LOW);
    digitalWrite(blueLedPin, LOW);
  //temperature index is lower than the light index
  // This indicates the temperature is lower. So, the blue light is
set to high.
  } else if (tempIndex < lightIndex) {
    digitalWrite(greenLedPin, LOW);
    digitalWrite(redLedPin, LOW);
    digitalWrite(blueLedPin, HIGH);
  } else {
   //temperature index is greater than the light index
   // This indicates the temperature is higher. So, the RED light is
set to high
    digitalWrite(greenLedPin, LOW);
    digitalWrite(redLedPin, HIGH);
    digitalWrite(blueLedPin, LOW);
  }

  delay(1000); // Periodicity in seconds
}

int readTemperature() {
  // Read analog voltage from temperature sensor
  int sensorValue = analogRead(temperaturePin);
  // Convert analog voltage to voltage (0-5V)
  float voltage = sensorValue * (5.0 / 1023.0);
  // Convert voltage to Celsius temperature
  float temperatureC = (voltage - 0.5) * 100;
  // Convert temperature to integer and return
  return (int)temperatureC;
}

int readLightIntensity() {
  // Read analog value from light sensor
  int sensorValue = analogRead(lightPin);
  // Map the analog value to a percentage representing light
intensity
```

```
  // The map() function scales the sensor value from the range [0,
1023] to [0, 100]
  // This ensures that the light intensity percentage is within a
manageable range
  return map(sensorValue, 0, 1023, 0, 100);
}

int getThresholdIndex(int value, const int thresholds[], int size) {
  // Iterate through the thresholds array
  for (int i = 0; i < size; i++) {
    // Check if the given value is less than the current threshold
    // If true, return the index of the current threshold
    if (value < thresholds[i]) {
      return i;
    }
  }
  // If the value is greater than or equal to all thresholds,
  // return the index of the last threshold
  return size - 1;
}
```

EXERCISE 3.1: TEMPERATURE METER v2.0

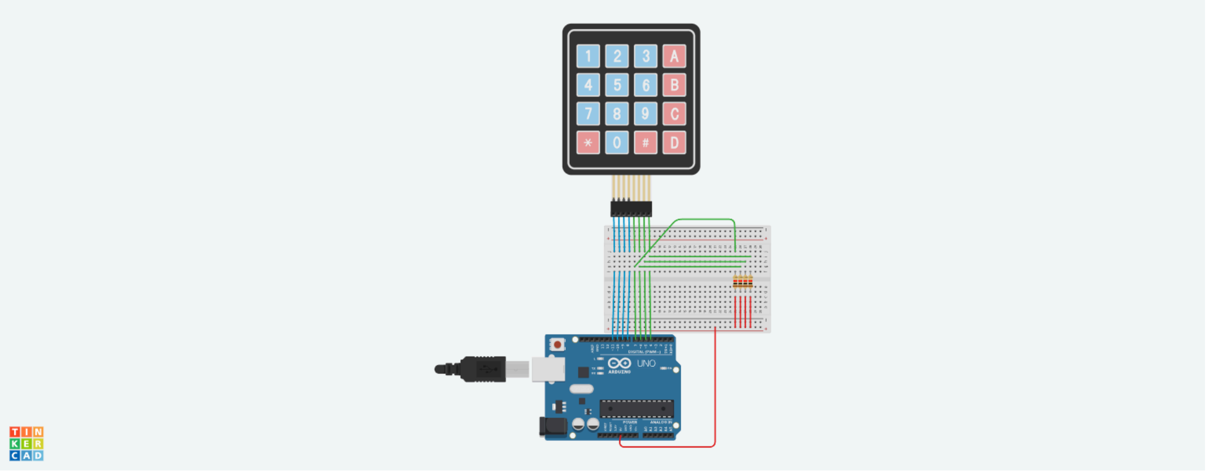


Link to board

Code:

```
// (C) Jitish Rajankumar Padhya, Raghav Tengse, Utkarsh Singh,
group: 23 (2024)
// Work package 3
// Exercise 3.1
// Submission code: 2345166
void setup(){
  //define the bit rate at which data is sent
 Serial.begin(9600);
}
void loop(){
  //store analog data in a variable
  float tempAnalogData = analogRead(A0);
  //formula to convert analog data to voltage
  //where 5.0 is the voltage provided and 1023 is the possible
  //values for resistance, ranging from 0 to 1023
  float voltage = (tempAnalogData * 5.0)/1023.0;
  //calculating the temperature using the voltage reading
  float temperature = (voltage - 0.5)/0.01;
  //printing the values returned from the analogReading
  Serial.print(temperature);
  Serial.print("°C");
  Serial.println("");
  // 500ms delay between every reading provided
  delay(500);
}
```

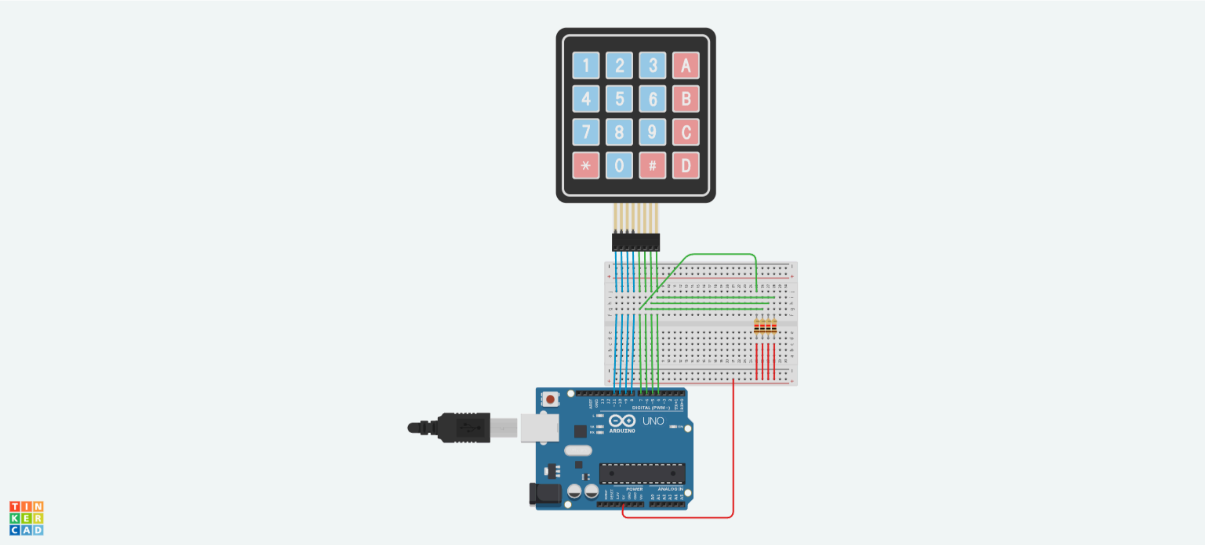
EXERCISE 3: KEYBOARD SCANNING



[Link to board](#)

```
// (C) Jitish Rajankumar Padhya, Raghav Tengse, Utkarsh Singh,
group: 23 (2024)
// Work package 3
// Exercise 3:Keyboard Scanning
// Submission code: 2345166
const int numRows = 4;
const int numCols = 4;

// Define the key map: a 4x4 matrix that maps keys on the keypad to
characters.map
char keys[numRows][numCols] = {
  {'1', '2', '3', 'A'},//First row of the keypad matrix
  {'4', '5', '6', 'B'},//Second row of the keypad matrix
  {'7', '8', '9', 'C'},//Third row of the keypad matrix
  {'*', '0', '#', 'D'}//Fourth row of the keypad matrix
};

// Define the pins for rows and columns
// An array holding the pin numbers for the rows of the keypad.
int rowPins[numRows] = {11, 10, 9, 8};
// An array holding the pin numbers for the columns of the keypad.
int colPins[numCols] = {7, 6, 5, 4};

void setup() {
  Serial.begin(9600);
  // Set column pins as inputs with pull-up resistors enabled
  for (int col = 0; col<  numCols; col++) {
    // Set the current column pin as input with pull-up resistor
enabled.
    pinMode(colPins[col], INPUT_PULLUP);
  }
  // Set row pins as outputs

  // Iterate over each row pin
  for (int row = 0; row < numRows; row++) {
    // Set the current row pin as output.
    pinMode(rowPins[row], OUTPUT);
    // Set the current row pin to HIGH initially.
    digitalWrite(rowPins[row], HIGH);
  }
}

void loop() {
  // Scan each row
  for (int row = 0; row < numRows; row++) {
    // Set current row low to scan
    digitalWrite(rowPins[row], LOW);

    // Check each column for key press
    for (int col = 0; col < numCols; col++) {
      if (digitalRead(colPins[col]) == LOW) { // Key pressed
        // Print the pressed key
        Serial.println(keys[row][col]);
        // Wait for debounce
        delay(50);
        // Wait until key released
        while (digitalRead(colPins[col]) == LOW);
        delay(50);
      }
    }

    // Reset current row to high
    digitalWrite(rowPins[row], HIGH);
  }
}
```

EXERCISE 4: KEYBOARD SCANNING v2.0



[Link to board]()

Code:

```
// (C) Jitish Rajankumar Padhya, Raghav Tengse, Utkarsh Singh,
group: 23 (2024)
// Work package 3
// Exercise 4:
// Submission code: 2345166

#include <Keypad.h> // keyboard function using keypad.h library

const byte ROWS = 4; // size of rows
const byte COLUMNS = 4; // size of rows

// initiate a 4*4 matrix to represent the keys on keypad
char Keys[ROWS][COLUMNS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

byte rowPins[ROWS] = {11, 10, 9, 8}; // pins connected to the rows
byte colPins[COLUMNS] = {7, 6, 5, 4}; // pins connected to the
columns

// Using the keypad library create a keyboard object
// to map the keys, pins, rows and columns
```

```
Keypad keypad =
Keypad(makeKeymap(Keys),rowPins,colPins,ROWS,COLUMNS);

void setup(){ // runs once when started or resetted
  Serial.begin(9600); // Set serial communication
}

void loop(){ // keeps running after setup()
  char pressedKey = keypad.getKey(); // get the key pressed
  Serial.println(pressedKey); // print the pressedKey in the monitor
  delay(100); // set a delay to be able to print input properly
}
```