**Activity 4 : Programming Embedded Systems (Having Fun with Arduino)**

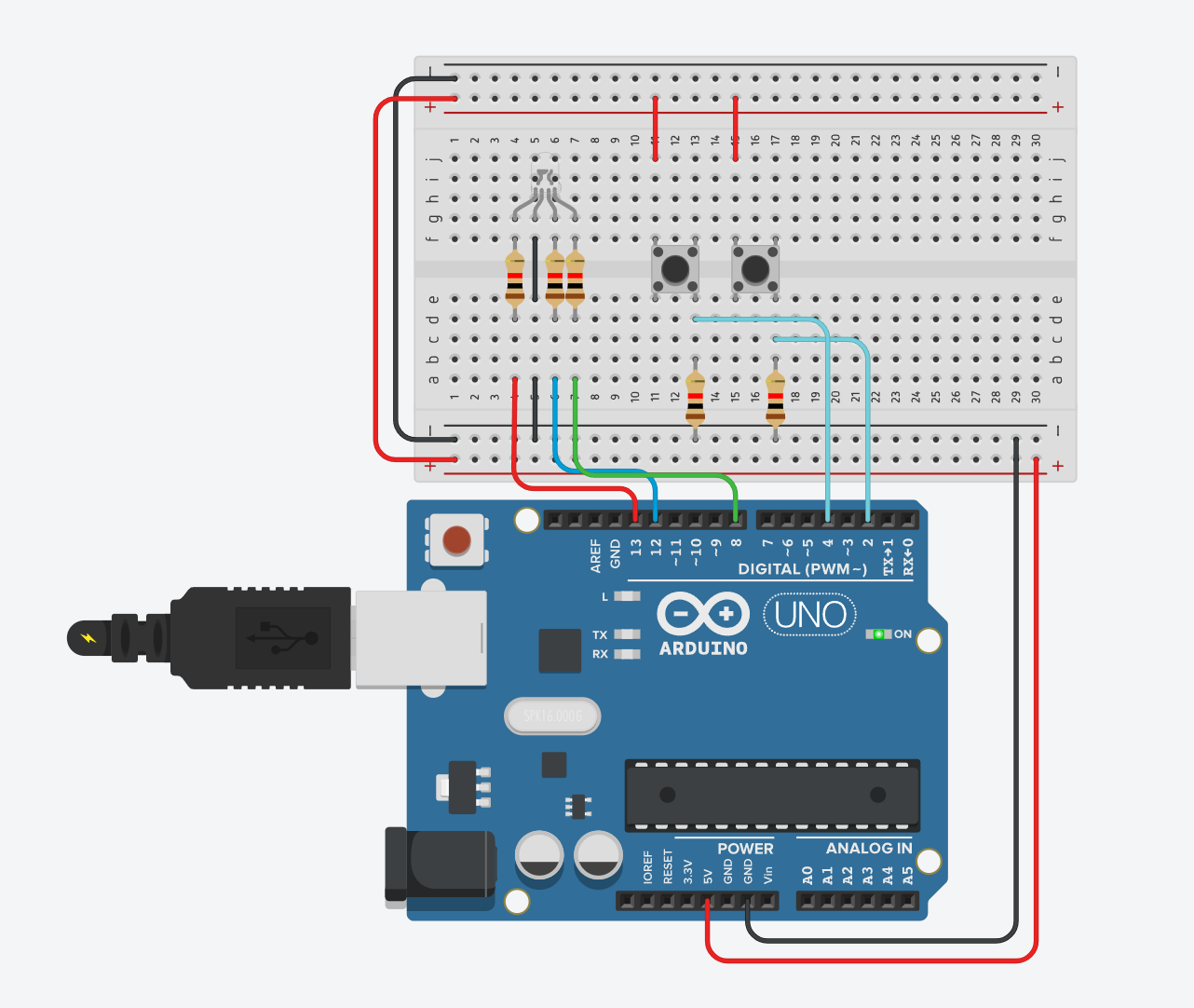
| **Group No :** XX  **Group Member :**   1. Name Surname ID 2. Name Surname ID 3. Name Surname ID 4. Name Surname ID |
| --- |

# **A Chance to be “Outstanding”**

**Prerequisite**: You have to complete all parts before doing this part.

This activity applies core concepts in **embedded systems, real-time processing, and user interaction**, making it relevant for **IoT applications, digital locks, and user-input devices**. It reinforces your understanding of **button debouncing, LED control, and state machine design**—skills essential for developing smart electronic systems.

In this activity, you will develop an **Arduino-based memory game** using push buttons and an RGB LED. The game randomly generates a **sequence of 4 button presses (A and B)** and flashes LEDs to display the sequence. The user must **repeat the sequence correctly** by pressing the buttons in the correct order. If the user succeeds, they will receive a **winning LED animation**; otherwise, they must try again.



### **Expected Output Example**

* **Game starts:**
  + "New sequence: A B A B"
  + (LED flashes Green-Red-Green-Red)
* **User Input:**
  + Button A pressed → "Input: A"
  + Button B pressed → "Input: B"
  + Button A pressed → "Input: A"
  + Button B pressed → "Input: B"
* **Correct Sequence:**
  + "Correct! You win!"
  + Blue LED blinks 5 times
* **Incorrect Sequence:**
  + "Incorrect! Try again!"
  + Red LED blinks 5 times

After you have finished, call a TA or an instructor to check your output. Also, put the code in the answer box.

Example Video Clip : [YouTUBE](https://youtu.be/F_S5z5rCuxE?si=LIYMkg2sXgKm_Fw3)

Code Example:

// --- Constants and Pin Definitions ---

enum class GameState {

STATE\_DISPLAY,

STATE\_INPUT,

STATE\_RESULT

};

GameState state = GameState::STATE\_DISPLAY;

char sequence[4]; // Stores the randomly generated sequence (e.g., 'A', 'B', 'A', 'B')

char userInput[4]; // Stores the sequence entered by the user

void setup() {

Serial.begin(9600);

// Initialize button pins and LED

//----------Fill your code here----------//

//--------------------------------------//

// Seed the random generator (using an unconnected analog pin for noise).

randomSeed(analogRead(0));

// Generate the first random sequence and prepare for display.

generateSequence();

state = GameState::STATE\_DISPLAY;

}

void loop() {

switch(state) {

case GameState::STATE\_DISPLAY: {

// This state displays the sequence one flash at a time.

//----------Fill your code here----------//

//--------------------------------------//

}

case GameState::STATE\_INPUT: {

// This state is read the pushbuttons and record user input (using edge detection).

//----------Fill your code here----------//

//--------------------------------------//

}

case GameState::STATE\_RESULT: {

// Show the outcome using LED animations.

//----------Fill your code here----------//

//--------------------------------------//

}

}

}

// Function to generate a new random sequence of 'A's and 'B's.

void generateSequence() {

for (int i = 0; i < 4; i++) {

int r = random(0, 2); // Generates 0 or 1.

sequence[i] = (r == 0) ? 'A' : 'B';

}

// For debugging: print the new sequence to the Serial Monitor.

Serial.print("New sequence: ");

for (int i = 0; i < 4; i++) {

Serial.print(sequence[i]);

Serial.print(" ");

}

Serial.println();

}

| Put your code here |
| --- |

*— THIS IS THE END OF OUTSTANDING PART —*