**Dawson College**

**Electrical Engineering Technology Department**

**Introduction to Internet of Things**

**Project Name:**

Number Memory Game

**Team Members:**

Cromwell Austria (ID: 2338117)

&

Christine George Bondesto (ID: 2135699)

May 12, 2025

2. Project Description

We came up with a game inspired by a game from the popular website “Human Benchmark”. The aim of the game is to memorize a random sequence of numbers to the best of the player’s ability. The player will then be prompted with a single digit shortly after, called a ‘KEY’. The goal is to then determine whether the KEY was part of the previous sequence of numbers or not. The game continues endlessly until the player make a mistake.

**What problem does it solve or what functionality does it offer?**

The game involves short-term memory, focus, and quick decision-making by challenging players to recall if a digit was in a sequence.

**How it works:**

* The **LCD** displays a randomly generated 7-digit sequence for a limited time, which the player must memorize.
* After the sequence disappears, a single digit called the **“KEY”** is shown on the LCD.
* The player must decide if the KEY was part of the sequence:
* Press **buttonPin1** if the KEY **was in** the sequence (IN).
* Press **buttonPin2** if the KEY **was not in** the sequence (OUT).
* If the player is **correct**:
* The **RGB LED** lights **green**.
* The LCD displays **“Correct!”**
* The **display time shortens by 100ms**, making the game progressively harder.
* If the player is **wrong**:
* The **RGB LED** lights **red**.
* The **buzzer** sounds twice using the tone() function.
* The LCD displays **“Wrong!”** and the player's **final score**.
* A new game begins after returning to the main menu, which shows the **current high score** and waits for input.

**A person thinking about a computer

AI-generated content may be incorrect.Final Assembly Diagram:**

**FIGURE 1 - PROJECT DIAGRAM.**

3. Circuit Diagram

**Inputs:**

* Push Button 1
* Push Button 2

**Outputs:**

* LCD Module (16x2 LCD)
* Buzzer
* RGB LED

Below is a description of the hardware connections, which are also shown graphically using Tinkercad.

Table 1 - Circuit Connections

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Part** | **Arduino Uno Pin** | **Notes** |
| **S1, S2** | Push Button Module (x2) | Signal pin → D3 and D4  VCC → 5V  GND → GND | Modules have internal pull-up; set pinMode to INPUT\_PULLUP. |
| **D1** | RGB LED Module | Red → D2  Green → D6  GND → GND | Module has built-in resistors. Only red and green pins used. |
| **BZ1** | Passive Buzzer Module | Signal (S) → D5  VCC → 5V  GND → GND | Since it’s passive, tone() is required to generate sound. |
| **LCD** | 1602 LCD Module (with header) | RS → D7, EN → D8, D4 → D9, D5 → D10, D6 → D11, D7 → D12  VSS → GND, VDD → 5V, V0 → 820 Ω → GND | Standard 16x2 display. V0 controls contrast. |
| — | +5V Rail | From Arduino 5V pin | Powers modules and LCD. Do not use 3.3 V. |

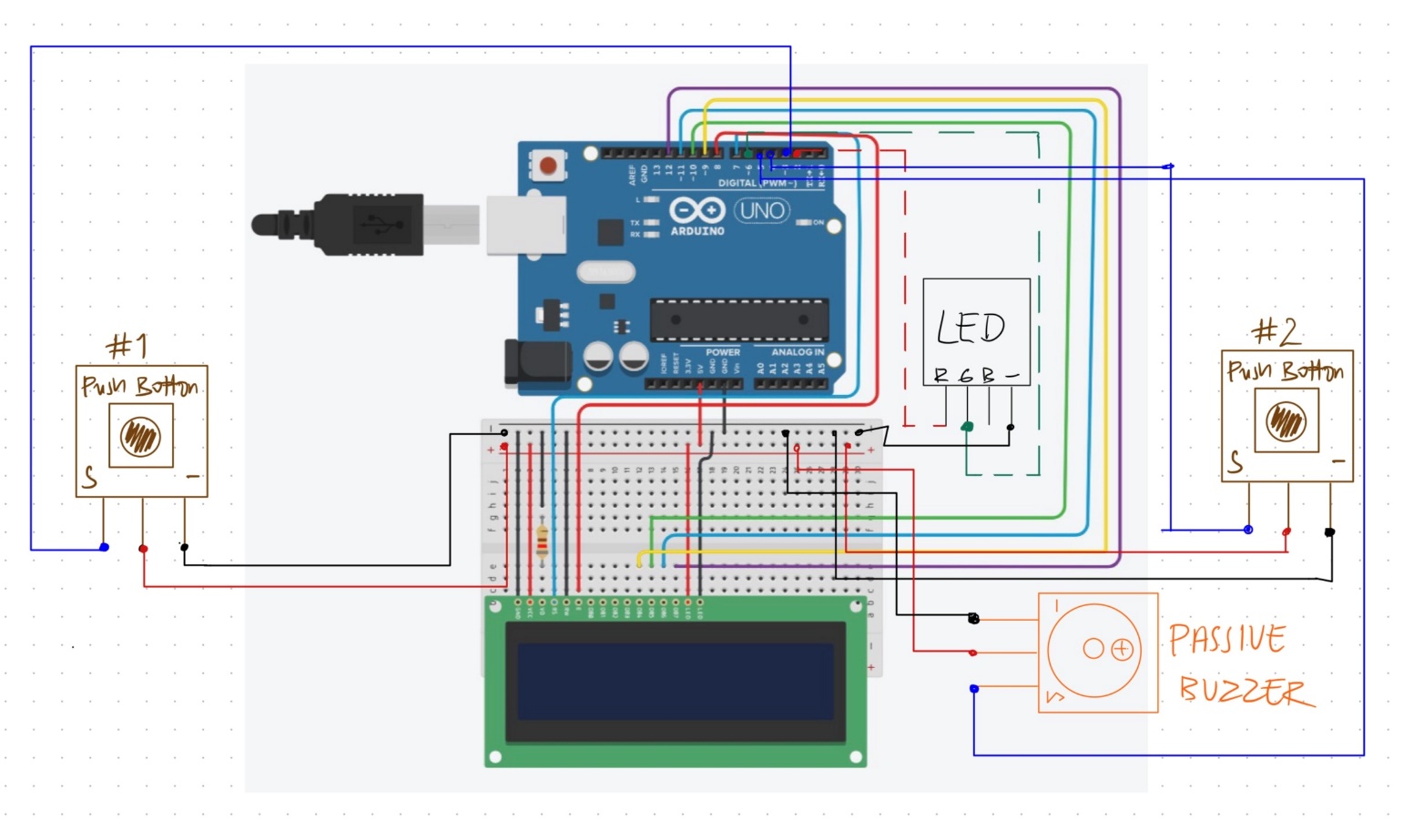
****

Figure 2 - Circuit Connections Diagram. Image from tinkercad and GOODNOTES, EDUCATIONAL purpose only.

4. Code Documentation

**4.1 Library Used**

#include <LiquidCrystal.h> //Controls LCD display; allows to print messages, moving cursors, clear the screen and others.

**4.2 Global Constants & Pins**

Below is the list of constant variables used in the code. As constants they do not change during the running of the code, and as global they can be used inside and outside functions.

Table 2 - Global constants and Pins.

|  |  |  |
| --- | --- | --- |
| **Constant / Pin** | **Purpose** | **Typical Value / Assignment** |
| *buttonPin1* | “IN” decision button | D3 |
| *buttonPin2* | “OUT” decision button | D4 |
| *green* | Green output (RGB LED) | D6 |
| *red* | Red output (RGB LED) | D2 |
| *buzzer* | Passive buzzer tone output | D5 |
| *rs* | LCD RS (Register Select) | D7 |
| *en* | LCD Enable | D8 |
| *d4, d5, d6, d7* | LCD data lines (4-bit mode) | D9, D10, D11, D12 |
| *displayTime* | Time to show sequence (decreases with difficulty) | Starts at 2000 ms, min 400 ms |
| *highscore* | Tracks best score in a session | Initialized at 0 |

**4.3 Main Code Structure: Function Descriptions and Responsibilities**

We separated our code into 3 sections: Initialization and setup, Main loop, Game functions.

**Function Explanations:**

Table 3 - Function descriptions and Responsibilities.

|  |  |
| --- | --- |
| **Functions** | **Descriptions and Responsibilities** |
| void setup() | * Initializes pins, LCD, and serial * Seeds random number generator |
| void loop() | * Loops mainMenu() and numberGame() |
| void mainMenu() | * Displays “Press to Start” * Displays the player’s highscore for the duration of play * Waits for player input to begin game |
| void numberGame() | * Resets playerscore to zero * Resets display time (duration the sequence is being displayed to the player) to 2 seconds * In while, Calls generateSequence(), displaySequence(), increaseDifficulty(), checkTheKey(), and checkPlayerInput() in order * If player is correct -> call playerRight()   Else call playerWrong() and break while |
| void generateSequence() | * Generates 7 random digits and stores them in a variable |
| void displaySequence() | * Clears LCD * Displays “Remember” * Waits 1 second before displaying the entire 7 digit sequence |
| void increaseDifficulty() | * Sets the duration of how long the sequence is being displayed to the player for |
| void checkTheKey() | * Generates a random digit from 0-10, called a KEY * Clears display * Displays “Digit” * Displays KEY * verifies if KEY is contained in the 7 digit sequence (correct = true or false depending..) |
| void checkPlayerInput() | * Waits indefinitely until an input is received from the player (if button 1 or 2 is pressed) |
| void playerRight() | * Adds a point to the players score * Decreases the sequence display time by 100ms IF the display time is greater than 400ms * Clears display * LED lights up green * LCD displays “Correct!” * Waits 1500ms then turns LED turns off |
| void playerWrong() | * Clears LCD * LED lights up red * Calls errorNoise() * Prints “Wrong!” and the player’s score right under that * Waits 1500ms and turns LED off * Updates the highscore shown in the main menu IF the player’s score is greater than the current highscore |
| void errorNoise() | * Activates the buzzer for 150ms at a frequency of 50 * Waits for 200ms * Activates the buzzer again for 150ms at a frequency of 50 |

**Input and Output Logic:**

* buttonPin1 is the "IN" button (pressed when the KEY is **IN** the 7 digit sequence)
* ButtonPin2 is the "NOT" button (pressed when the KEY is **NOT** in the 7 digit sequence)
* LCD displays messages like the 7 digit sequence, digit to check, score, etc.
* Green LED lights up when the right input was received from the player.
* Red LED lights up and Buzzer activates when the wrong input was received from the player.

**Code Explanation**

While the code can be found on our GitHub, the explanation of how the code works is shown below.

1. **Main Menu –** LCD displays “Press to Start” and the player’s highscore right under while the game waits for a button press (indicating the player is ready to start the game, in which case, it then calls numberGame().
2. **Number Game** – Runs a “while” that executes generateSequence(), displaySequence(), increaseDifficulty(), checkTheKey(), checkPlayerInput(). IF the player presses the right button, call playerRight(), else playerWrong() then break the while (indicating the game is over)

5. Ethics, Privacy, or Security Disclaimer

We considered the ethical implications of our number memory game project. There are no immediate privacy or security threats because the program runs completely offline and doesn't gather, store, or send any personal information. However, it is still important to make sure that players are in a distraction-free, safe setting when playing the game. This project is not meant for use in any crucial decision-making applications; rather, it is meant to be entertaining and cognitively challenging.

6. References

* 37 SENSOR KIT TUTORIAL FOR UNO V2.0: <https://www.elegoo.com/>
* Human Benchmark: https://humanbenchmark.com/
* Arduino LiquidCrystal Library: https://github.com/arduino-libraries/LiquidCrystal
* Tinkercad – https://tinkercad.com