Team id:PNT2022TMID40101

Project Name:Smart Waste Management System for Metropolitan Cities

Publish data to the IBM cloud

Simon is a simple electronic memory game: the user has to repeat a growing sequence of colors. The sequence is displayed by lighting up the LEDs. Each color also has a corresponding tone.

In each turn, the game will play the sequence, and then wait for the user to repeat the sequence by pressing the buttons according to the color sequence. If the user repeated the sequence correctly, the game will play a "leveling-up" sound, add a new color at the end of the sequence, and move to the next turn.

The game continues until the user has made a mistake. Then a game over sound is played, and the game restarts.

Hardware

Item	Quantity	Notes
Arduino Uno R3	1	
5mm LED	4	Red, Green, Blue, and Yellow
12mm Push button	4	Red, Green, Blue, and Yellow
Resistor	4	220Ω
Piezo Buzzer	1	

<figure> <figcaption> Hardware for the Simon game, kit available on Tindie </figcaption> </figure>

Diagram

<figure> <figcaption>Simon connection diagram</figcaption> </figure>

Pin Connections

Arduino Pin	Device
12	Red LED
11	Green LED
10	Blue LED
9	Yellow LED
8	Buzzer
5	Red Button
4	Green Button
3	Blue Button
2	Yellow Button

- The LEDs are connected through a 220Ω resistor each.

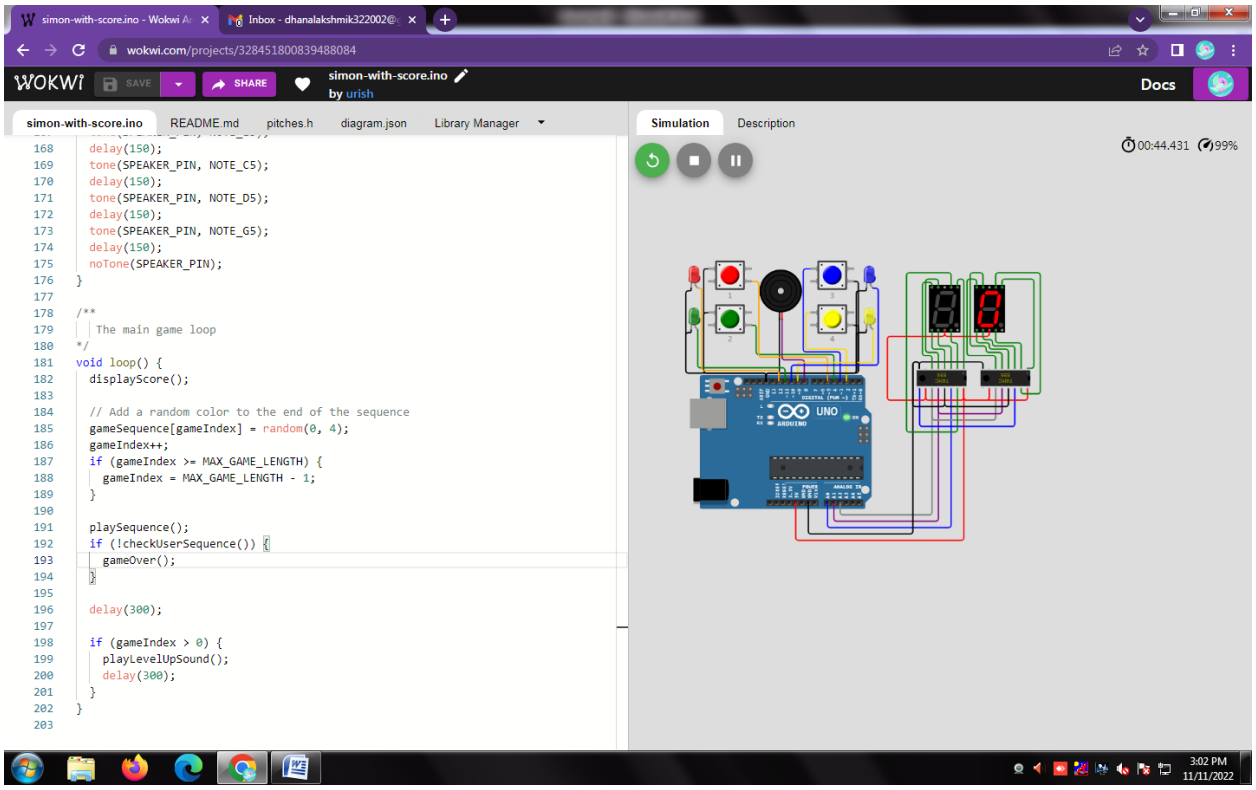
```
#include "pitches.h"

/* Constants - define pin numbers for LEDs,
   buttons and speaker, and also the game tones: */
const uint8_t ledPins[] = {9, 10, 11, 12};
const uint8_t buttonPins[] = {2, 3, 4, 5};
#define SPEAKER_PIN 8

// These are connected to 74HC595 shift register (used to show game score):
const int LATCH_PIN = A1;  // 74HC595 pin 12
const int DATA_PIN = A0;  // 74HC595pin 14
const int CLOCK_PIN = A2;  // 74HC595 pin 11

#define MAX_GAME_LENGTH 100

const int gameTones[] = { NOTE_G3, NOTE_C4, NOTE_E4, NOTE_G5};

/* Global variables - store the game state */
```

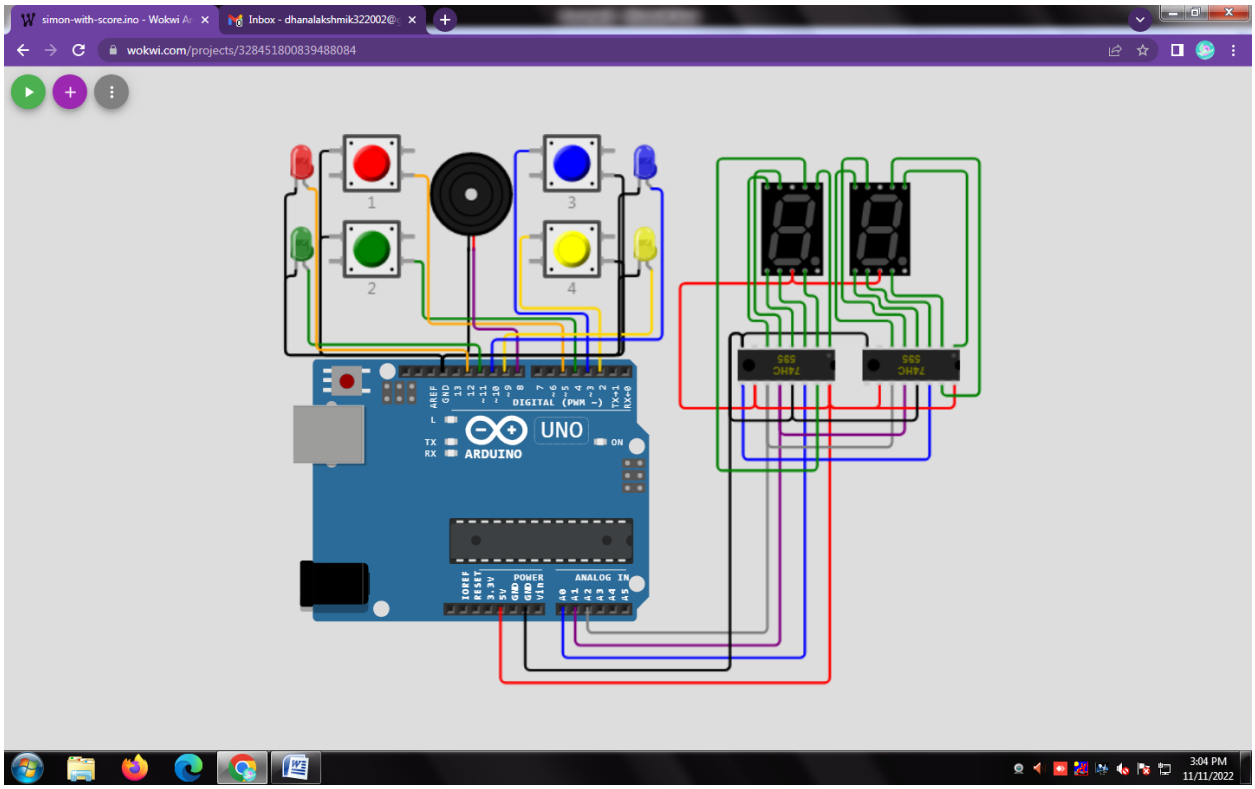
```
uint8_t gameSequence[MAX_GAME_LENGTH] = {0};
uint8_t gameIndex = 0;

/**
   Set up the Arduino board and initialize Serial communication
*/
void setup() {
  Serial.begin(9600);
  for (byte i = 0; i < 4; i++) {
    pinMode(ledPins[i], OUTPUT);
    pinMode(buttonPins[i], INPUT_PULLUP);
  }
  pinMode(SPEAKER_PIN, OUTPUT);
  pinMode(LATCH_PIN, OUTPUT);
  pinMode(CLOCK_PIN, OUTPUT);
  pinMode(DATA_PIN, OUTPUT);

  // The following line primes the random number generator.
  // It assumes pin A3 is floating (disconnected):
  randomSeed(analogRead(A3));
}

/* Digit table for the 7-segment display */
const uint8_t digitTable[] = {
  0b11000000,
  0b11111001,
  0b10100100,
  0b10110000,
  0b10011001,
  0b10010010,
  0b10000010,
  0b11111000,
  0b10000000,
  0b10010000,
};
const uint8_t DASH = 0b10111111;

void sendScore(uint8_t high, uint8_t low) {
  digitalWrite(LATCH_PIN, LOW);
  shiftOut(DATA_PIN, CLOCK_PIN, MSBFIRST, low);
  shiftOut(DATA_PIN, CLOCK_PIN, MSBFIRST, high);
  digitalWrite(LATCH_PIN, HIGH);
}

void displayScore() {
  int high = gameIndex % 100 / 10;
  int low = gameIndex % 10;
  sendScore(high ? digitTable[high] : 0xff, digitTable[low]);
}

/**
   Lights the given LED and plays a suitable tone
*/
void lightLedAndPlayTone(byte ledIndex) {
```

```
  digitalWrite(ledPins[ledIndex], HIGH);
  tone(SPEAKER_PIN, gameTones[ledIndex]);
  delay(300);
  digitalWrite(ledPins[ledIndex], LOW);
  noTone(SPEAKER_PIN);
}

/**
   Plays the current sequence of notes that the user has to repeat
*/
void playSequence() {
  for (int i = 0; i < gameIndex; i++) {
    byte currentLed = gameSequence[i];
    lightLedAndPlayTone(currentLed);
    delay(50);
  }
}

/**
    Waits until the user pressed one of the buttons,
    and returns the index of that button
*/
byte readButtons() {
  while (true) {
    for (byte i = 0; i < 4; i++) {
      byte buttonPin = buttonPins[i];
      if (digitalRead(buttonPin) == LOW) {
        return i;
      }
    }
    delay(1);
  }
}

/**
  Play the game over sequence, and report the game score
*/
void gameOver() {
  Serial.print("Game over! your score: ");
  Serial.println(gameIndex - 1);
  gameIndex = 0;
  delay(200);

  // Play a Wah-Wah-Wah-Wah sound
  tone(SPEAKER_PIN, NOTE_DS5);
  delay(300);
  tone(SPEAKER_PIN, NOTE_D5);
  delay(300);
  tone(SPEAKER_PIN, NOTE_CS5);
  delay(300);
  for (byte i = 0; i < 10; i++) {
    for (int pitch = -10; pitch <= 10; pitch++) {
      tone(SPEAKER_PIN, NOTE_C5 + pitch);
      delay(5);
```

```
    }
  }
  noTone(SPEAKER_PIN);

  sendScore(DASH, DASH);
  delay(500);
}

/**
   Get the user's input and compare it with the expected sequence.
*/
bool checkUserSequence() {
  for (int i = 0; i < gameIndex; i++) {
    byte expectedButton = gameSequence[i];
    byte actualButton = readButtons();
    lightLedAndPlayTone(actualButton);
    if (expectedButton != actualButton) {
      return false;
    }
  }

  return true;
}

/**
   Plays a hooray sound whenever the user finishes a level
*/
void playLevelUpSound() {
  tone(SPEAKER_PIN, NOTE_E4);
  delay(150);
  tone(SPEAKER_PIN, NOTE_G4);
  delay(150);
  tone(SPEAKER_PIN, NOTE_E5);
  delay(150);
  tone(SPEAKER_PIN, NOTE_C5);
  delay(150);
  tone(SPEAKER_PIN, NOTE_D5);
  delay(150);
  tone(SPEAKER_PIN, NOTE_G5);
  delay(150);
  noTone(SPEAKER_PIN);
}

/**
   The main game loop
*/
void loop() {
  displayScore();

  // Add a random color to the end of the sequence
  gameSequence[gameIndex] = random(0, 4);
  gameIndex++;
  if (gameIndex >= MAX_GAME_LENGTH) {
    gameIndex = MAX_GAME_LENGTH - 1;
```

```
  }

  playSequence();
  if (!checkUserSequence()) {
    gameOver();
  }

  delay(300);

  if (gameIndex > 0) {
    playLevelUpSound();
    delay(300);
  }
}
```




simon-with-score.ino - Wokwi
Inbox - dhanalakshmik322002@
wokwi.com/projects/328451800839488084
1
2
3
4
UNO
ARDUINO
DIGITAL (PWM ~)
ANALOG IN
POWER
74HC
595
3:04 PM
11/11/2022