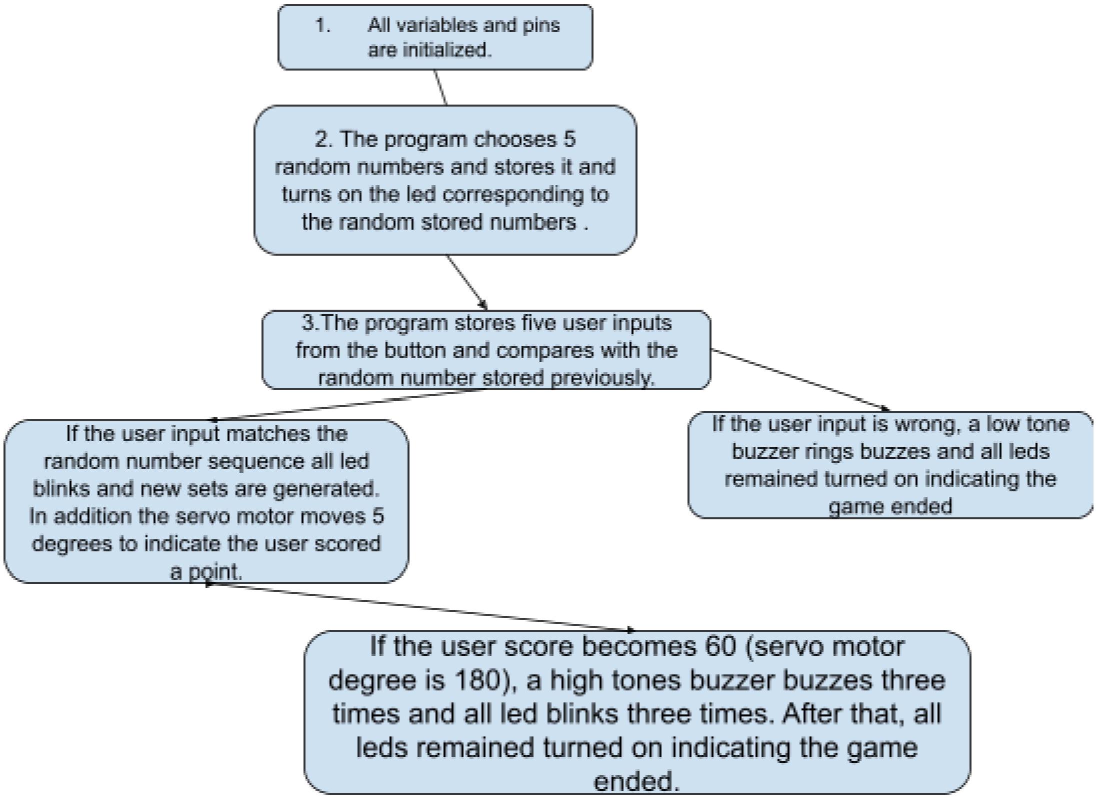
Simon Memory Game

Device outline:

The Arduino circuit I constructed is designed to serve as the Simon electronic memory game. First, the circuit shows a set of LED patterns with different colors. After that, the player should press a series of buttons corresponding to the given LED pattern that the circuit initially created. If the player does not push the appropriate buttons in order, the low-tone buzzer will buzz and all the LEDs will be turned on indicating the game ended and the user loses. However, if the player presses the correct series of buttons, the servo motor would move by 3 degrees to indicate the user has earned a point(the servo motor acts like a needle that points to a certain number in the custom-made sticker, it is a similar thing like a car gauge where the needle points a certain number). In addition, if the user survives the 60th stage of the game, the high-tone buzzer will buzz and all leads will blink three times indicating that the player won and the lead will remain turned on indicating the game is finished. In addition, the volume is adjustable using the potentiometer.

The code of this device works as following:



1. All variables and pins are initialized.

2. The program chooses 5 random numbers and stores it and turns on the led corresponding to the random stored numbers .

3.The program stores five user inputs from the button and compares with the random number stored previously.

If the user input matches the random number sequence all led blinks and new sets are generated. In addition the servo motor moves 5 degrees to indicate the user scored a point.

If the user input is wrong, a low tone buzzer rings buzzes and all leds remained turned on indicating the game ended

If the user score becomes 60 (servo motor degree is 180), a high tones buzzer buzzes three times and all led blinks three times. After that, all leds remained turned on indicating the game ended.

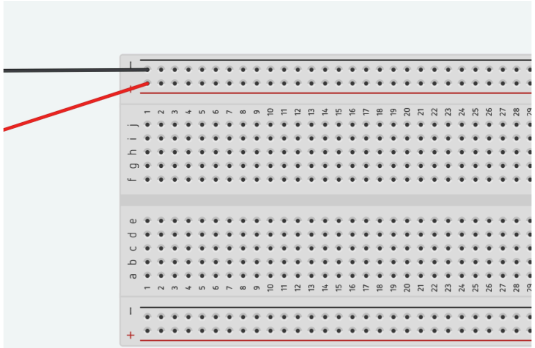
Tinkercad link: https://www.tinkercad.com/things/8OsIumJweLk-fantastic-tumelo

The electronics build:

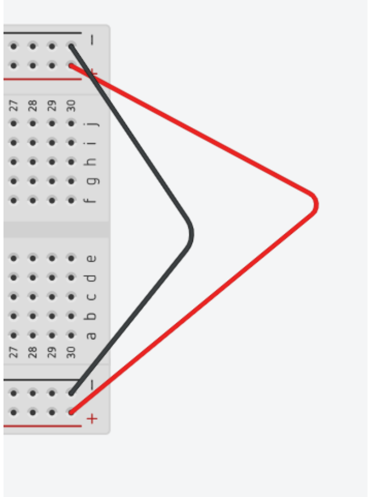
Parts used:

- Continuous servo motor-1
- Piezo buzzer-1
- Push buttons-5
- Leds-5
- 330Ω resistor-5
- 10kΩ resistor-5
- Arduino-1
- potentiometer-1
- Breadboard-1

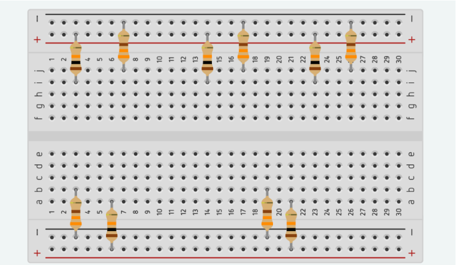
1. Connect a wire connecting the + rail to the 5v pin of the arduino and -reail to the GND pin of the arduino. (supplies power to the top rails)



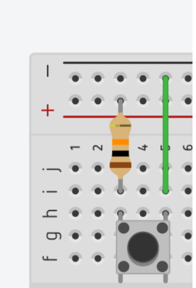
2. Connect the a wire from the upper + rail to the bottom + rail as well as for the -s.(supplies power to the bottom rails)



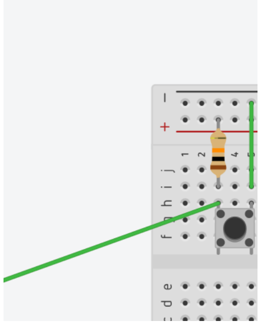
3. Connect all the 10kΩ resistor pins to the + rail of the arduino. AFter that, Place all the 330Ω resistors next to each 10kΩ resistor and connect to the - rail.(resistors prevent any explosions or fire of the components)



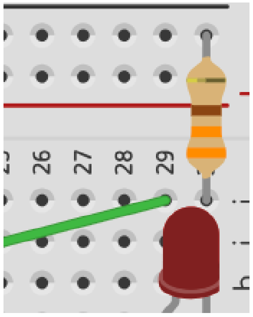
4. For all 10kΩ resistors, place the button as follows, and connect the other side of the button to the - rail.(supplies power to the button)



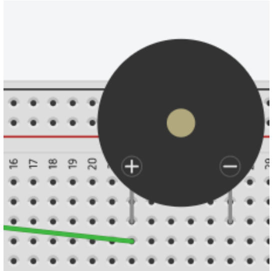
5. Connect a wire as the image for each button and attach it to the arduino pins (13, 12, 8, 7, 4) in order.(connects the push button to the arduino which lets it detect user input).



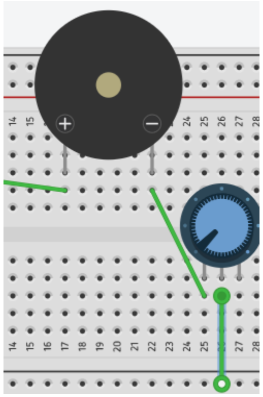
6. Connect a led so that the - pin of the led meets the 330Ω resistor. And connect a wire that will be attached to the arduino pins(11, 10, 9, 3, 5) in order. Repeat for all 5 leds(supplies power to led and lets the led to be programmed).



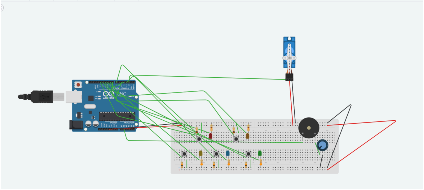
7. Place a piezo buzzer at an empty space of your breadboard and connect a wire at the positive pin and attach it to the A1 pin at the arduino(connecting the piezo buzzer to the arduino lets the piezo buzzer be programmed).



8. Place a potentiometer right below the piezo buzzer and connect the - pin of the piezo buzzer to the first pin of the potentiometer.After that, connect the second pin of the potentiometer to the - rail.(the potentiometer does not have direct connection to the arduino, however, the circuit is designed so that the potentiometer still functions without any programming of the potentiometer).



End result:



Physical build:

- 5- Arduino button caps for the push button cover. - $1.42(https://www.aliexpress.com/item/32852734000.html?src=google&pdp_npi=4%40dis!CAD!1.46!1.42!!!!!%40!65346216976!ppc!!!)

- Custom made sticker for the servo motor gauge score system. -$0 (craftable make it with paper, tape and protractor)

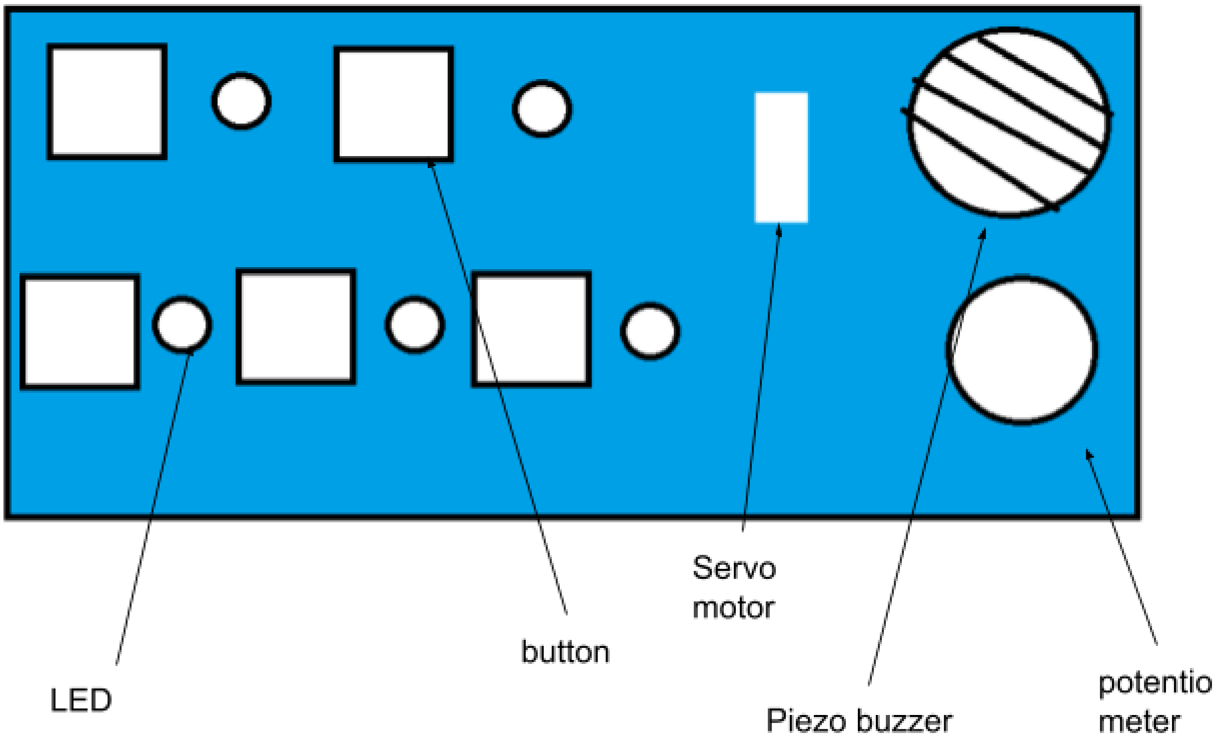
- Filament/3dn printer -$1.50 for filament and use of toronto public library's 3d printer(free)(https://www.torontopubliclibrary.ca/using-the-library/computer-services/innovation-spaces/3D-design-print.jsp#:~:text=Printing%20costs,of%20%240.15%20per%20print%20job.)

Total - $ 2.92

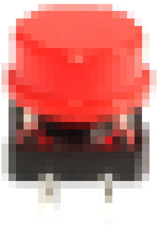
steps:

1. Create the custom made sticker using a protractor and mark every 3 degrees of the angle. After that, write number 1-60 at the marked points.
2. Using the 3d printer print the base part of the plastic case and place the circuit above the base.
3. 3d Print the upper cover of the plastic case and fit the buttons, servo motor and led piezo buzzer in the appropriate location.

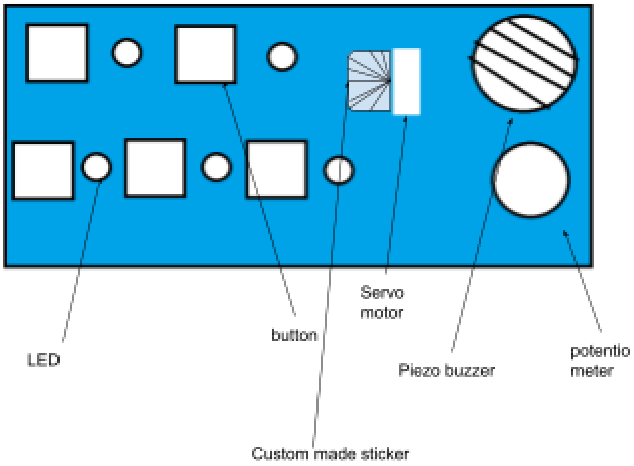
How the upper case looks like:



4. Attach the button cover to the push buttons like this:

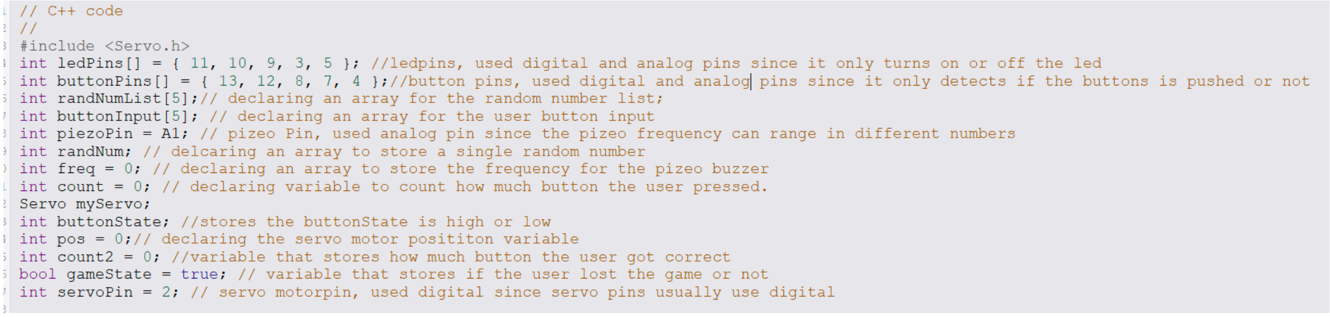


5. Stick the custom made sticker right besides the servo motor.

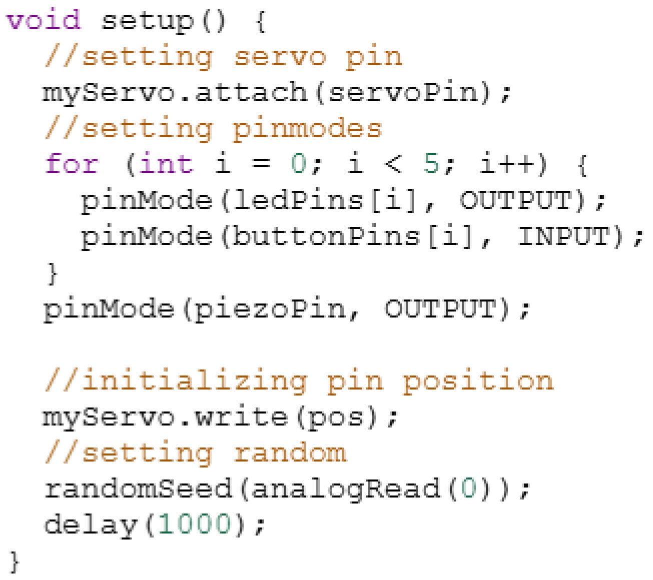


Code review(plz also look that the comments):

variables:

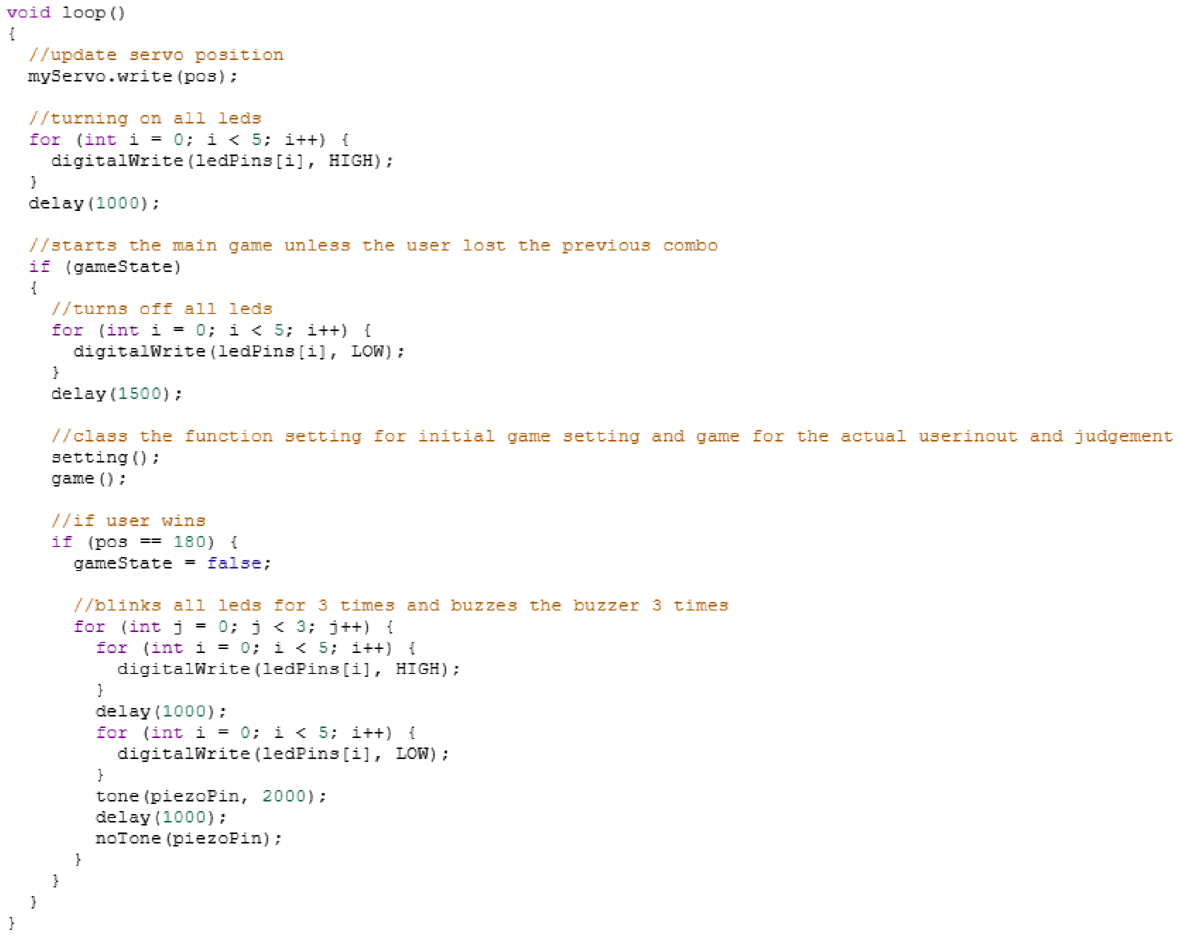

```
// C++ code
//
#include <Servo.h>
int ledPins[] = { 11, 10, 9, 3, 5 }; //ledpins, used digital and analog pins since it only turns on or off the led
int buttonPins[] = { 13, 12, 8, 7, 4 };//button pins, used digital and analog pins since it only detects if the buttons is pushed or not
int randNumList[5];// declaring an array for the random number list;
int buttonInput[5]; // declaring an array for the user button input
int piezoPin = A1; // pizeo Pin, used analog pin since the pizeo frequency can range in different numbers
int randNum; // delcaring an array to store a single random number
int freq = 0; // declaring an array to store the frequency for the pizeo buzzer
int count = 0; // declaring variable to count how much button the user pressed.
Servo myServo;
int buttonState; //stores the buttonState is high or low
int pos = 0;// declaring the servo motor posititon variable
int count2 = 0; //variable that stores how much button the user got correct
bool gameState = true; // variable that stores if the user lost the game or not
int servoPin = 2; // servo motorpin, used digital since servo pins usually use digital
```

(Set up) function which sets up all the settings and pinmode



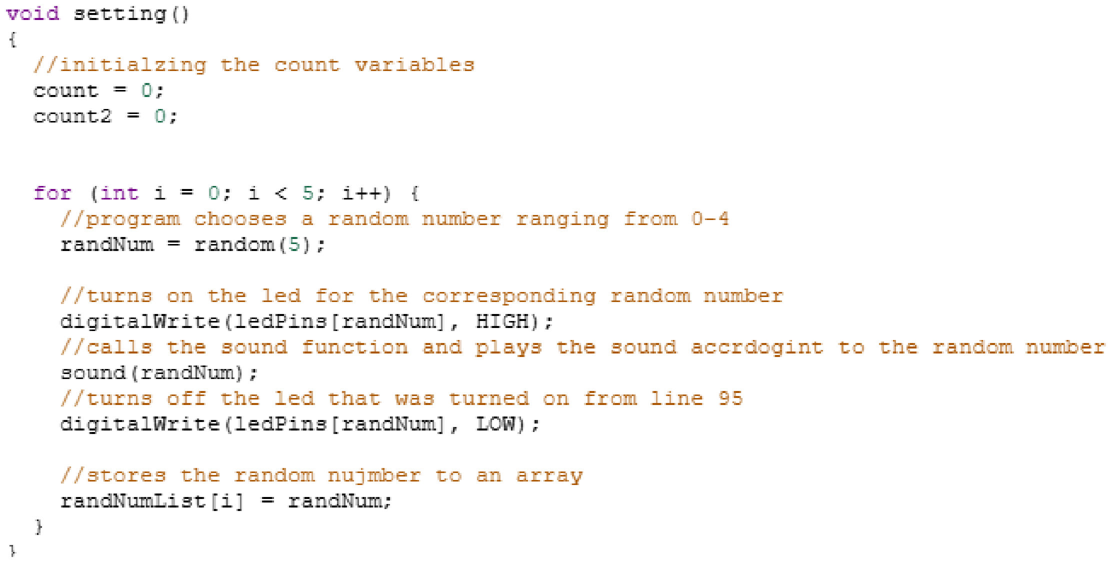
```
void setup() {
  //setting servo pin
  myServo.attach(servoPin);
  //setting pinmodes
  for (int i = 0; i < 5; i++) {
    pinMode(ledPins[i], OUTPUT);
    pinMode(buttonPins[i], INPUT);
  }
  pinMode(piezoPin, OUTPUT);

  //initializing pin position
  myServo.write(pos);
  //setting random
  randomSeed(analogRead(0));
  delay(1000);
}
```

Main function which runs the game and judges if the user is winning or not. And accordingly, the program decides what to do with the leds and the buzzer.



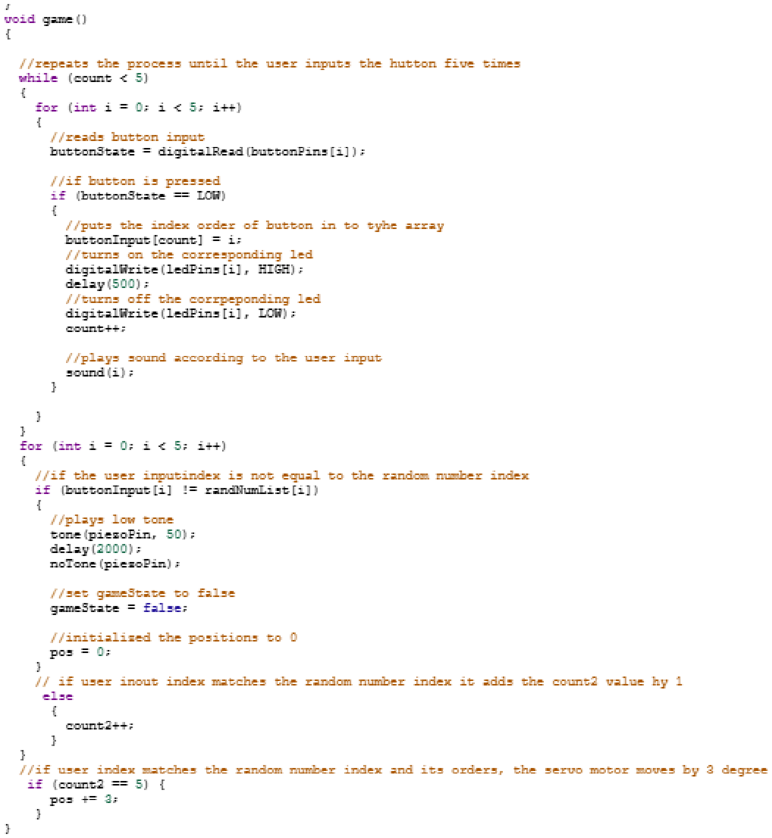
```
void loop()
{
  //update servo position
  myServo.write(pos);

  //turning on all leds
  for (int i = 0; i < 5; i++) {
    digitalWrite(ledPins[i], HIGH);
  }
  delay(1000);

  //starts the main game unless the user lost the previous combo
  if (gameState)
  {
    //turns off all leds
    for (int i = 0; i < 5; i++) {
      digitalWrite(ledPins[i], LOW);
    }
    delay(1500);

    //class the function setting for initial game setting and game for the actual userinout and judgement
    setting();
    game();

    //if user wins
    if (pos == 180) {
      gameState = false;

      //blinks all leds for 3 times and buzzes the buzzer 3 times
      for (int j = 0; j < 3; j++) {
        for (int i = 0; i < 5; i++) {
          digitalWrite(ledPins[i], HIGH);
        }
        delay(1000);
        for (int i = 0; i < 5; i++) {
          digitalWrite(ledPins[i], LOW);
        }
        tone(piezoPin, 2000);
        delay(1000);
        noTone(piezoPin);
      }
    }
  }
}
```

Function that sets the random numbers, stores them and turns on and off leds according to it.



```
void setting()
{
  //initialzing the count variables
  count = 0;
  count2 = 0;


  for (int i = 0; i < 5; i++) {
    //program chooses a random number ranging from 0-4
    randNum = random(5);

    //turns on the led for the corresponding random number
    digitalWrite(ledPins[randNum], HIGH);
    //calls the sound function and plays the sound accrdogint to the random number
    sound(randNum);
    //turns off the led that was turned on from line 95
    digitalWrite(ledPins[randNum], LOW);

    //stores the random nujmber to an array
    randNumList[i] = randNum;
  }
}
```

Function that gets the user input and compares it to the random number, after the program returns values such as the score or gameState which decides if the player won or lost.



```
;
void game()
{

  //repeats the process until the user inputs the hutton five times
  while (count < 5)
  {
    for (int i = 0; i < 5; i++)
    {
      //reads button input
      buttonState = digitalRead(buttonPins[i]);

      //if button is pressed
      if (buttonState == LOW)
      {
        //puts the index order of button in to tyhe array
        buttonInput[count] = i;
        //turns on the corresponding led
        digitalWrite(ledPins[i], HIGH);
        delay(500);
        //turns off the corrpeponding led
        digitalWrite(ledPins[i], LOW);
        count++;

        //plays sound according to the user input
        sound(i);
      }

    }
  }
  for (int i = 0; i < 5; i++)
  {
    //if the user inputindex is not equal to the random number index
    if (buttonInput[i] != randNumList[i])
    {
      //plays low tone
      tone(piezoPin, 50);
      delay(2000);
      noTone(piezoPin);

      //set gameState to false
      gameState = false;

      //initialized the positions to 0
      pos = 0;
    }
    // if user inout index matches the random number index it adds the count2 value hy 1
     else
      {
        count2++;
      }
  }
  //if user index matches the random number index and its orders, the servo motor moves by 3 degree
   if (count2 == 5) {
      pos += 3;
    }
}
```