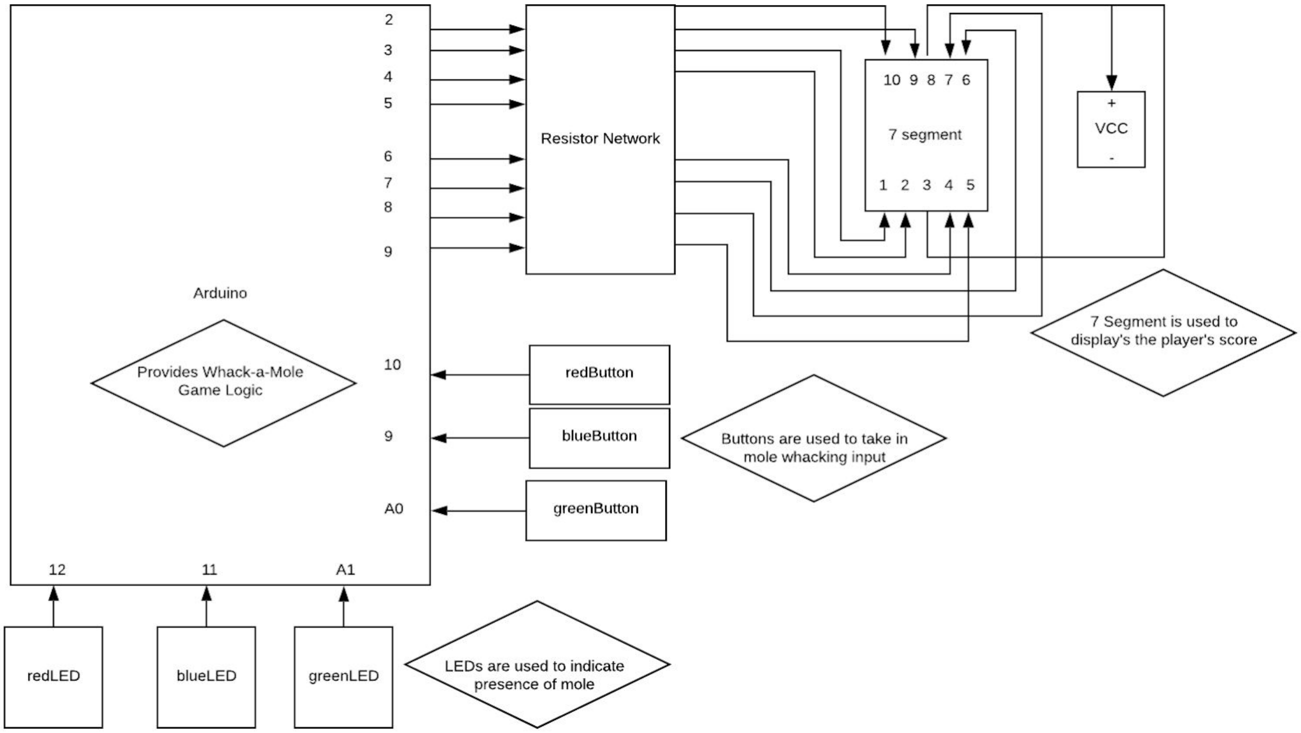
1. Title - Team Members

Whack A Mole - Kathleen Labog

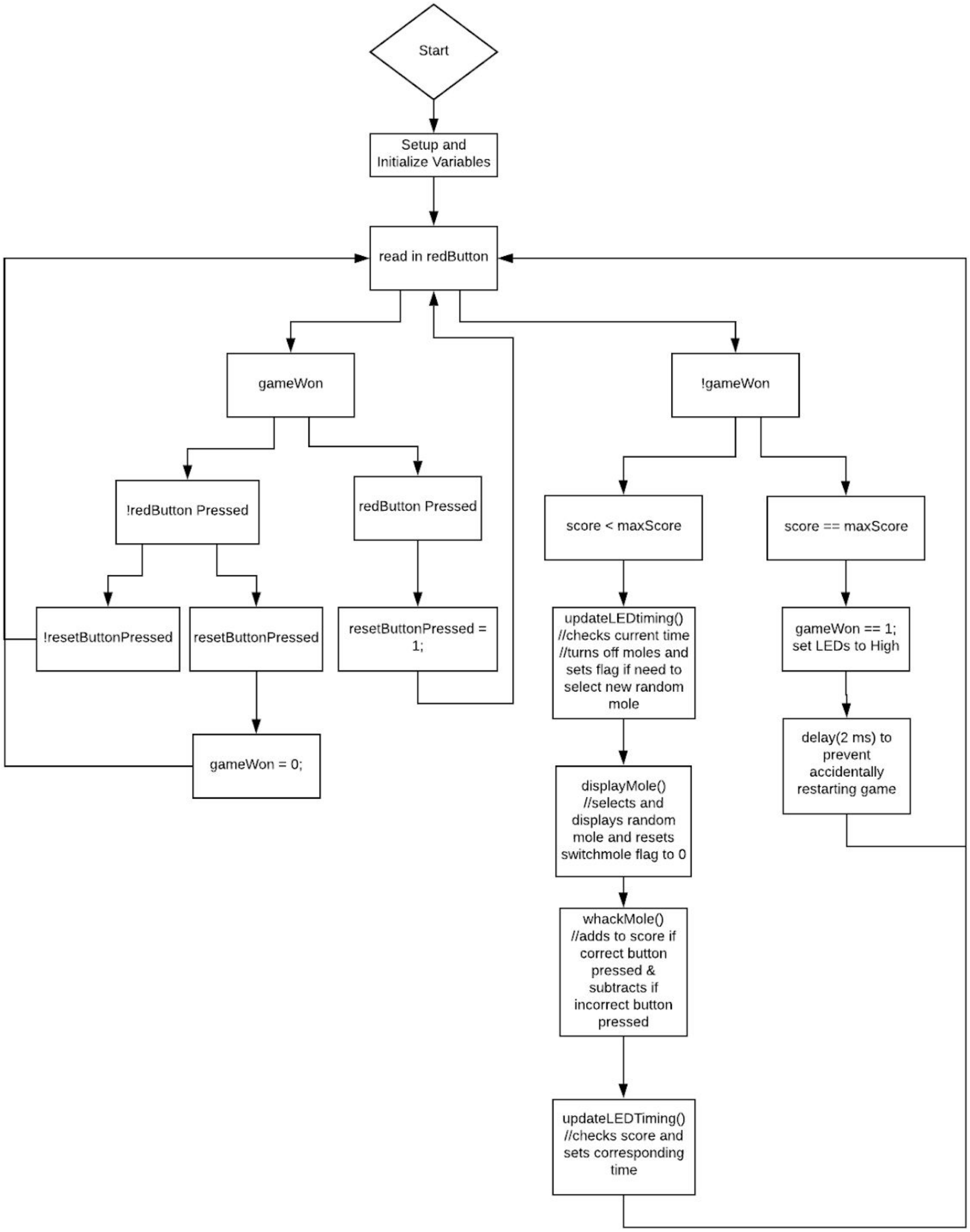
2. Project Description

My project is a whack-a-mole using 3 buttons and 3 LEDs. The game is started by pressing the red button. When an LED is lit , that corresponds to a mole being at that location. While the LED is lit, the player can press the button of matching color to whack the mole and earn a point. If the user misses the mole by pressing the incorrect color button or pressing the button too late, the user loses a point. The game speeds up when the user has earned 3 points and again when the user has earned 6 points and slows back down when the user loses enough points to be in a previous game speed. The current points earned are displayed on a 7 segment display. The player wins the game when they earn all 9 points and all LEDs light up. The player can start the game again by pressing the red button again.

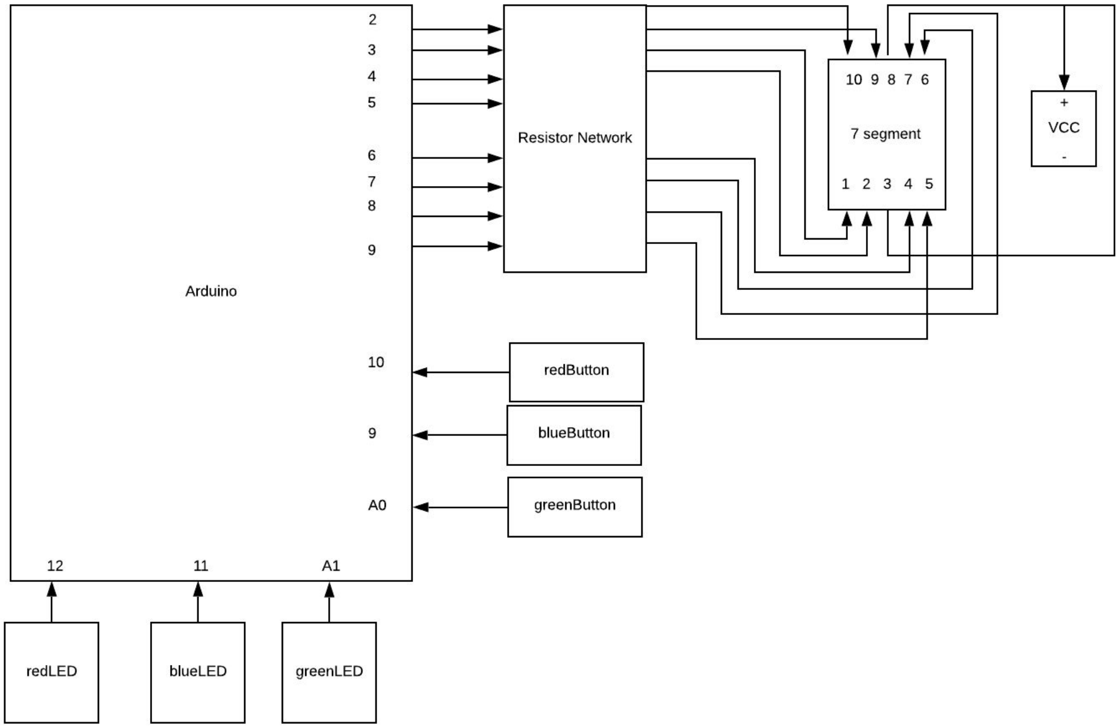
3. Block Diagram & Flow Diagram which clearly shows where parts of your system were implemented (Arduino or K64F)



Arduino
Provides Whack-a-Mole Game Logic
2
3
4
5
6
7
8
9
Resistor Network
10 9 8 7 6
7 segment
1 2 3 4 5
+
VCC
-
7 Segment is used to display's the player's score
10
9
A0
redButton
blueButton
greenButton
Buttons are used to take in mole whacking input
12
11
A1
redLED
blueLED
greenLED
LEDs are used to indicate presence of mole


Start
Setup and Initialize Variables
read in redButton
gameWon
!gameWon
!redButton Pressed
redButton Pressed
!resetButtonPressed
resetButtonPressed
resetButtonPressed = 1;
gameWon = 0;
score < maxScore
score == maxScore
updateLEDtiming() //checks current time //turns off moles and sets flag if need to select new random mole
displayMole() //selects and displays random mole and resets switchmole flag to 0
whackMole() //adds to score if correct button pressed & subtracts if incorrect button pressed
updateLEDTiming() //checks score and sets corresponding time
gameWon == 1; set LEDs to High
delay(2 ms) to prevent accidentally restarting game

4. Circuit Schematic (hand-drawing is ok, if you don't feel like a professional) indicating all components and PIN NUMBERS.



5. Testing Evaluation

Using the Arduino Com monitor and a DMM I was able to debug and test my code and hardware to ensure the correct timing and inputs were being read.

6. Discussions

It was very difficult to do a mini-project by myself but I had worked on the labs by myself and did not feel I would be able to find a partner especially because personal issues had set me so far behind. The system is limited by the number of pins allowed and could have been enhanced by using an LCD screen to increase the amount of points given. A winning LED sequence, sound buzzer, and photosensors could have been added to provide the user with a whacking movement and provide more realistic Whack-a-mole arcade experience but the function timing is very smooth. I was able to apply my knowledge of the 7-segment interfacing to the project and my understanding of the importance of timing. I mismanaged my time and as a result was only able to implement the system on arduino.

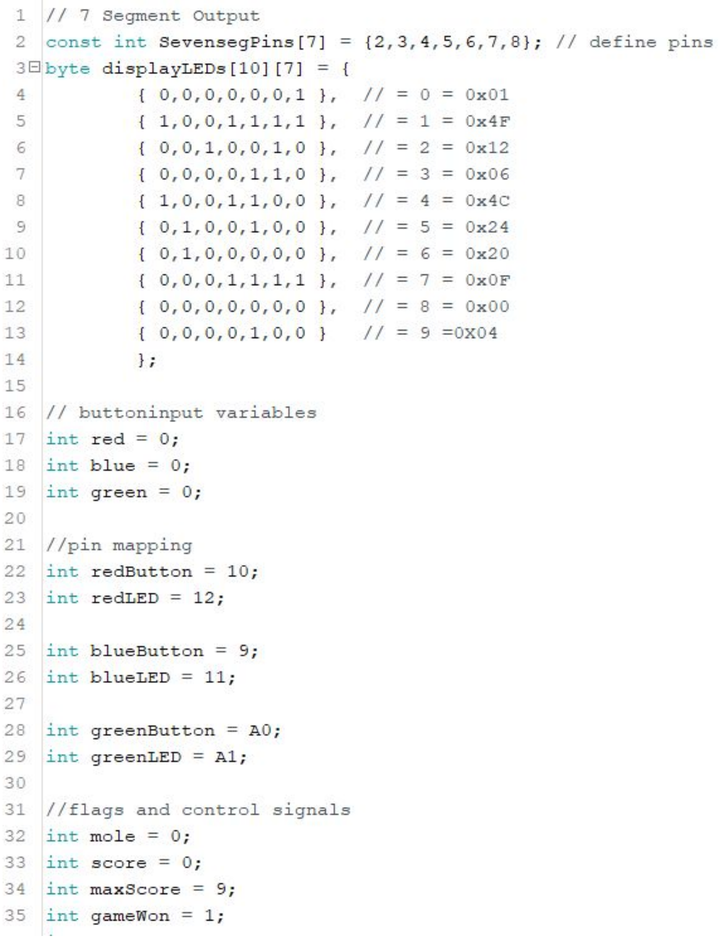
7. Roles and responsibilities

The project was done completely on my own.

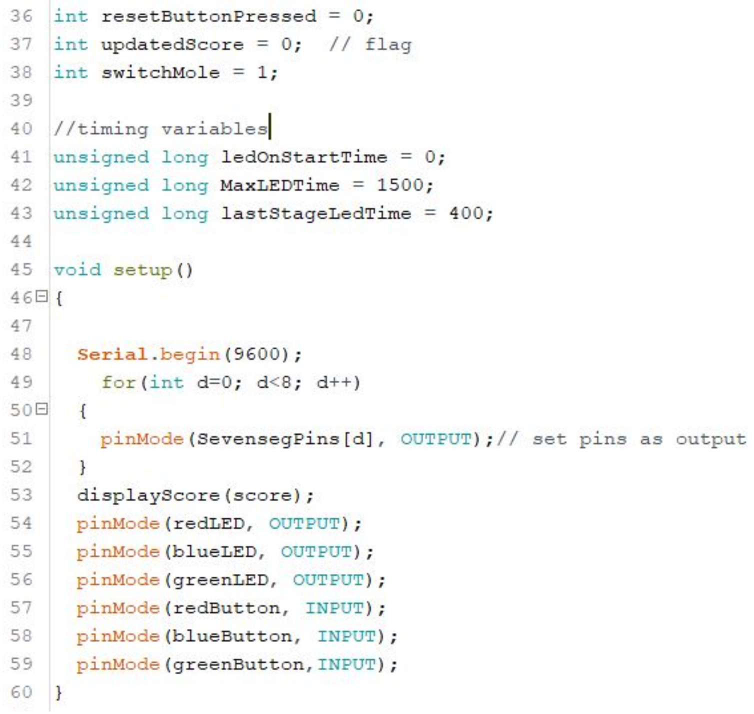
8. Conclusion

The system functions well as Whack-a-mole however is very basic and could be improved a lot.

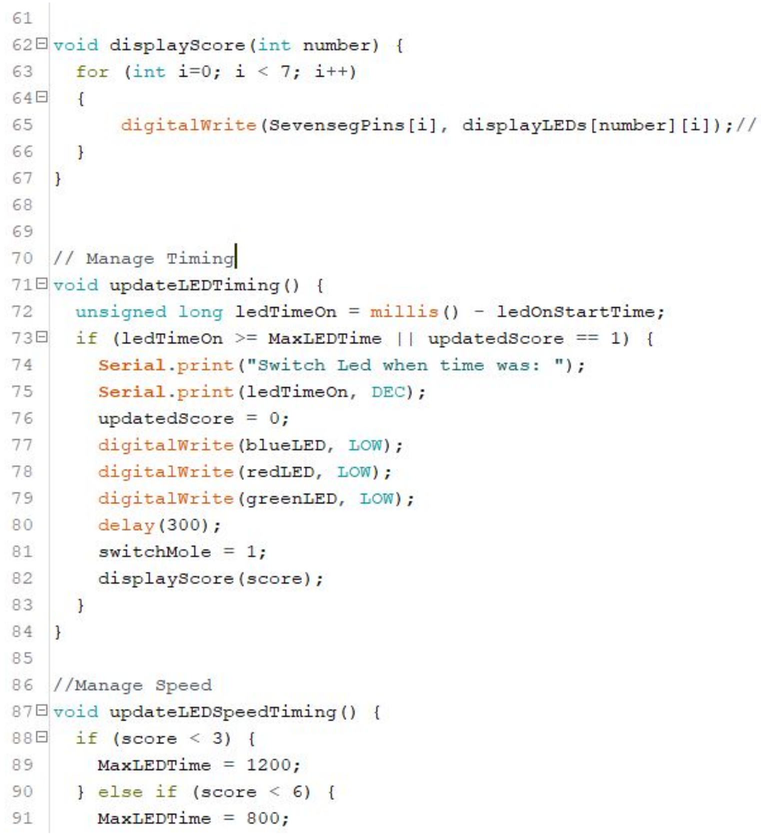
9. Appendix with the source code you wrote (not codes external libraries)



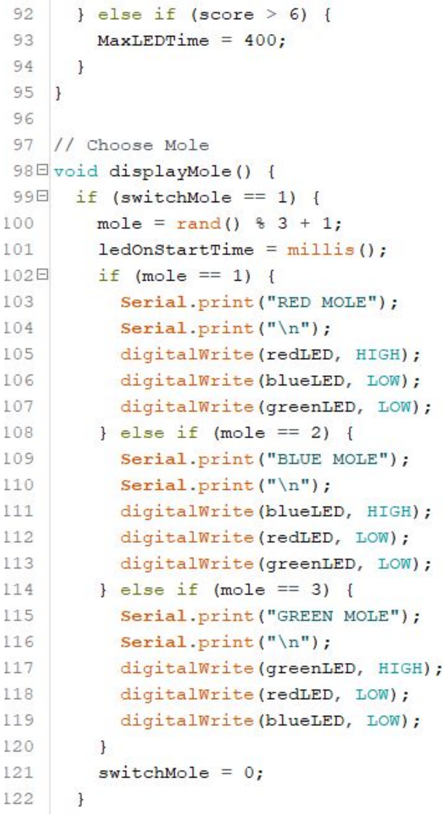
```
// 7 Segment Output
const int SevensegPins[7] = {2,3,4,5,6,7,8}; // define pins
byte displayLEDs[10][7] = {
        { 0,0,0,0,0,0,1 },  // = 0 = 0x01
        { 1,0,0,1,1,1,1 },  // = 1 = 0x4F
        { 0,0,1,0,0,1,0 },  // = 2 = 0x12
        { 0,0,0,0,1,1,0 },  // = 3 = 0x06
        { 1,0,0,1,1,0,0 },  // = 4 = 0x4C
        { 0,1,0,0,1,0,0 },  // = 5 = 0x24
        { 0,1,0,0,0,0,0 },  // = 6 = 0x20
        { 0,0,0,1,1,1,1 },  // = 7 = 0x0F
        { 0,0,0,0,0,0,0 },  // = 8 = 0x00
        { 0,0,0,0,1,0,0 }   // = 9 =0X04
        };

// buttoninput variables
int red = 0;
int blue = 0;
int green = 0;

//pin mapping
int redButton = 10;
int redLED = 12;

int blueButton = 9;
int blueLED = 11;

int greenButton = A0;
int greenLED = A1;

//flags and control signals
int mole = 0;
int score = 0;
int maxScore = 9;
int gameWon = 1;
```



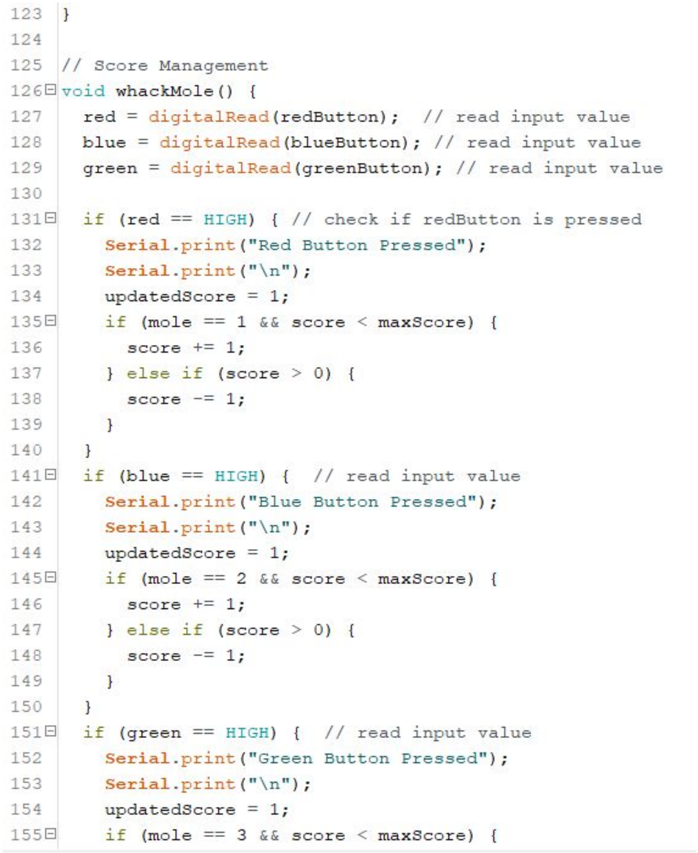
```
int resetButtonPressed = 0;
int updatedScore = 0;  // flag
int switchMole = 1;

//timing variables
unsigned long ledOnStartTime = 0;
unsigned long MaxLEDTime = 1500;
unsigned long lastStageLedTime = 400;

void setup()
{

  Serial.begin(9600);
    for(int d=0; d<8; d++)
  {
    pinMode(SevensegPins[d], OUTPUT);// set pins as output
  }
  displayScore(score);
  pinMode(redLED, OUTPUT);
  pinMode(blueLED, OUTPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(redButton, INPUT);
  pinMode(blueButton, INPUT);
  pinMode(greenButton,INPUT);
}
```



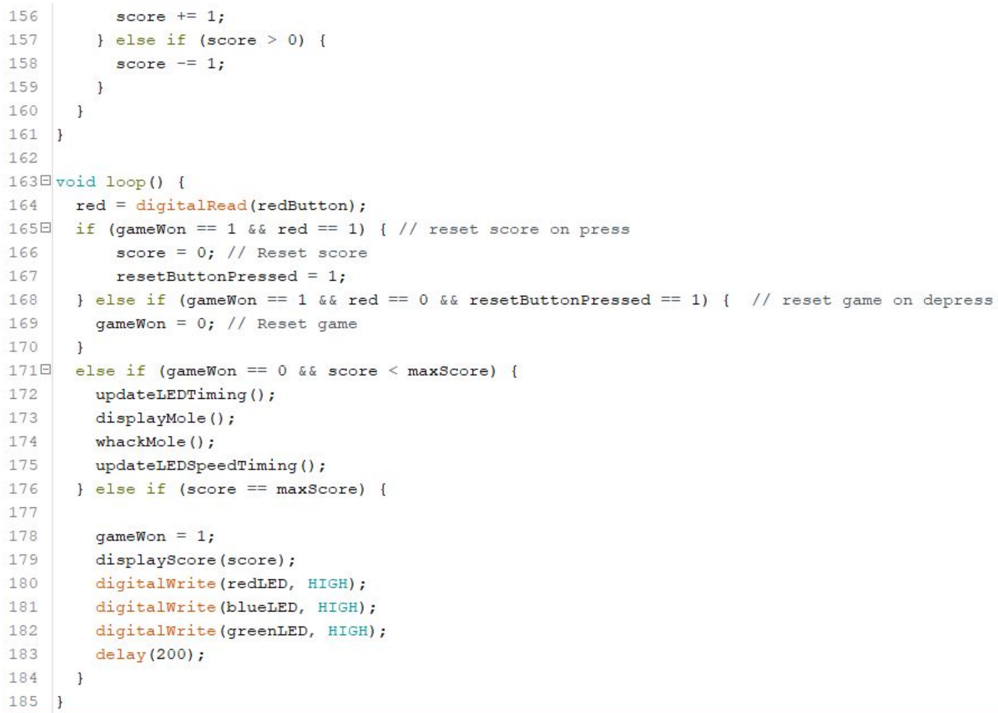
```
void displayScore(int number) {
  for (int i=0; i < 7; i++)
  {
      digitalWrite(SevensegPins[i], displayLEDs[number][i]);//
  }
}


// Manage Timing
void updateLEDTiming() {
  unsigned long ledTimeOn = millis() - ledOnStartTime;
  if (ledTimeOn >= MaxLEDTime || updatedScore == 1) {
    Serial.print("Switch Led when time was: ");
    Serial.print(ledTimeOn, DEC);
    updatedScore = 0;
    digitalWrite(blueLED, LOW);
    digitalWrite(redLED, LOW);
    digitalWrite(greenLED, LOW);
    delay(300);
    switchMole = 1;
    displayScore(score);
  }
}

//Manage Speed
void updateLEDSpeedTiming() {
  if (score < 3) {
    MaxLEDTime = 1200;
  } else if (score < 6) {
    MaxLEDTime = 800;
```



```
  } else if (score > 6) {
    MaxLEDTime = 400;
  }
}

// Choose Mole
void displayMole() {
  if (switchMole == 1) {
    mole = rand() % 3 + 1;
    ledOnStartTime = millis();
    if (mole == 1) {
      Serial.print("RED MOLE");
      Serial.print("\n");
      digitalWrite(redLED, HIGH);
      digitalWrite(blueLED, LOW);
      digitalWrite(greenLED, LOW);
    } else if (mole == 2) {
      Serial.print("BLUE MOLE");
      Serial.print("\n");
      digitalWrite(blueLED, HIGH);
      digitalWrite(redLED, LOW);
      digitalWrite(greenLED, LOW);
    } else if (mole == 3) {
      Serial.print("GREEN MOLE");
      Serial.print("\n");
      digitalWrite(greenLED, HIGH);
      digitalWrite(redLED, LOW);
      digitalWrite(blueLED, LOW);
    }
    switchMole = 0;
  }
```



```
}

// Score Management
void whackMole() {
  red = digitalRead(redButton);  // read input value
  blue = digitalRead(blueButton); // read input value
  green = digitalRead(greenButton); // read input value

  if (red == HIGH) { // check if redButton is pressed
    Serial.print("Red Button Pressed");
    Serial.print("\n");
    updatedScore = 1;
    if (mole == 1 && score < maxScore) {
      score += 1;
    } else if (score > 0) {
      score -= 1;
    }
  }
  if (blue == HIGH) {  // read input value
    Serial.print("Blue Button Pressed");
    Serial.print("\n");
    updatedScore = 1;
    if (mole == 2 && score < maxScore) {
      score += 1;
    } else if (score > 0) {
      score -= 1;
    }
  }
  if (green == HIGH) {  // read input value
    Serial.print("Green Button Pressed");
    Serial.print("\n");
    updatedScore = 1;
    if (mole == 3 && score < maxScore) {
```



```
      score += 1;
    } else if (score > 0) {
      score -= 1;
    }
  }
}

void loop() {
  red = digitalRead(redButton);
  if (gameWon == 1 && red == 1) { // reset score on press
      score = 0; // Reset score
      resetButtonPressed = 1;
  } else if (gameWon == 1 && red == 0 && resetButtonPressed == 1) {  // reset game on depress
    gameWon = 0; // Reset game
  }
  else if (gameWon == 0 && score < maxScore) {
    updateLEDTiming();
    displayMole();
    whackMole();
    updateLEDSpeedTiming();
  } else if (score == maxScore) {

    gameWon = 1;
    displayScore(score);
    digitalWrite(redLED, HIGH);
    digitalWrite(blueLED, HIGH);
    digitalWrite(greenLED, HIGH);
    delay(200);
  }
}
```