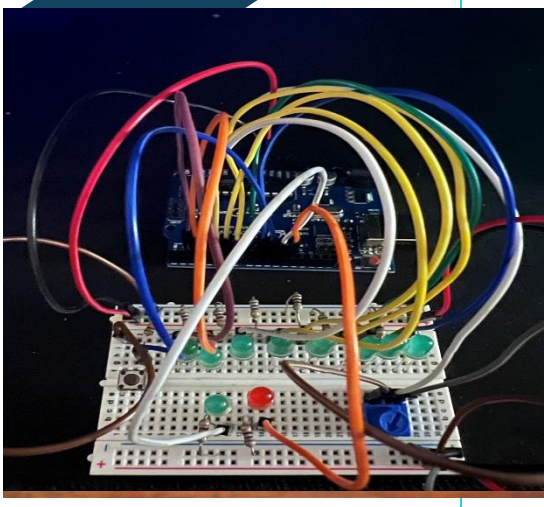
INDIVIDUAL MINI PROJECT: BATTLESHIPS

Noshad Boksh 210600322

BATTLESHIPS ARDUINO

By Noshad Boksh 210600322



CONTENTS

☐ Ships 10 LEDS –
8 Any color for Battleship 1 Green 1 Red for HIT MISS

☐ Resistors –
1 x 10k Ohm
10 x 560 Ohm

☐ Shot Confirm - Push Button x 1

☐ Shot Aimer - Potentiometer x 1

☐ Breadboard Wires

INTRODUCTION

☐The goal of this game is to guess where all the battleships are in 3 tries. The computer generates 3 ships at random. Each Led represents a ship. You have 5 attempts to pick and choose which ship and shoot it down.

HOW TO PLAY

☐ Twist the Potentiometer to pick between ships. Each LED will light up when you scroll through them. The LED that is lit is the ship you have targeted

☐ To confirm your hit, press the button and the LED will flash.

☐ After the chosen LED is shot and flashed, one of the two lights on the left will flash.

☐ The red light means you missed the ship

☐ The green light means you hit the ship.

☐ If you shoot all three ships before running out of shots the ships will light up and you won.

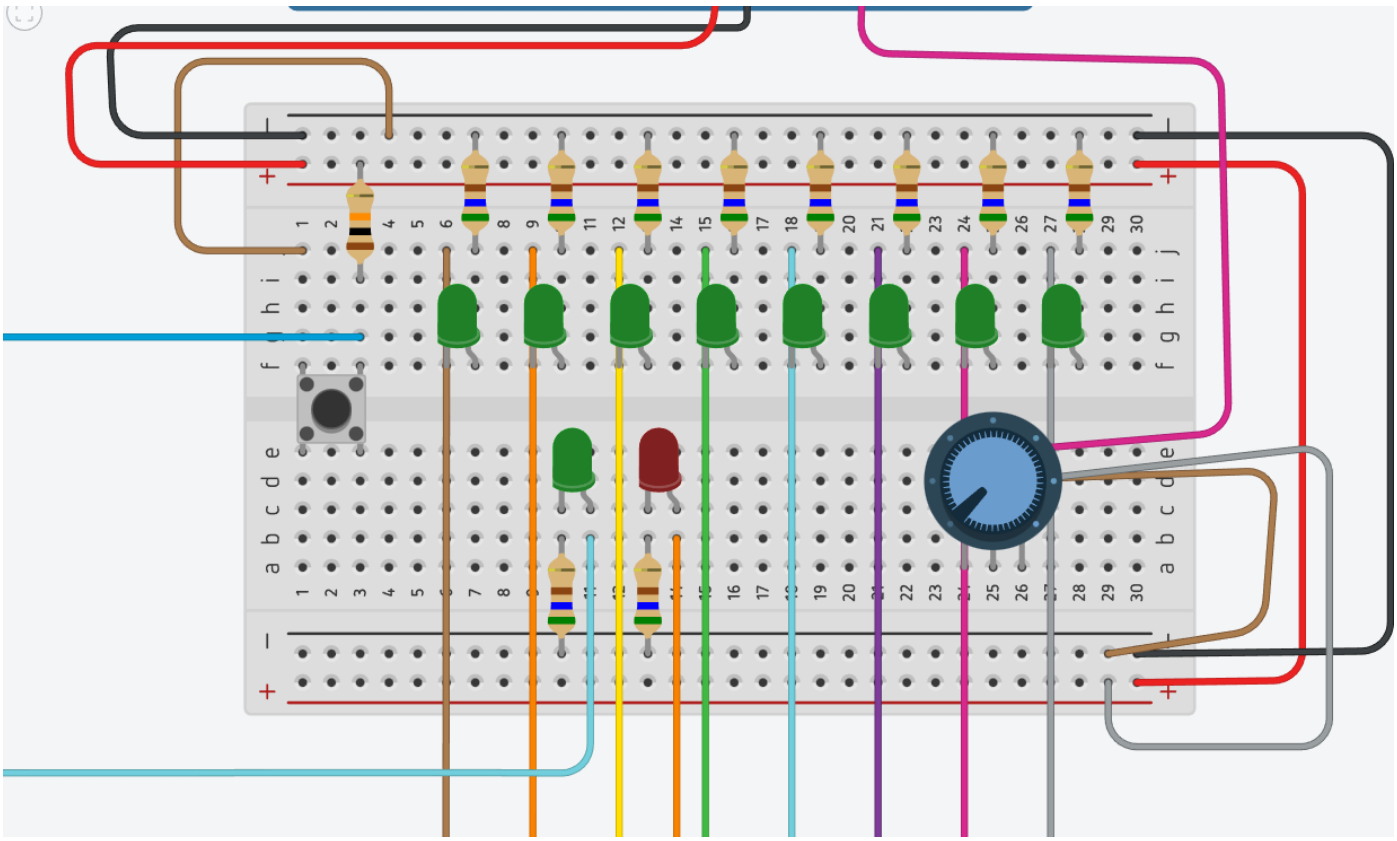
☐ However, if you use all shots and have not shot the ships, you lose, and the ships positions will be revealed to you.

PROGRAMMERS GUIDE

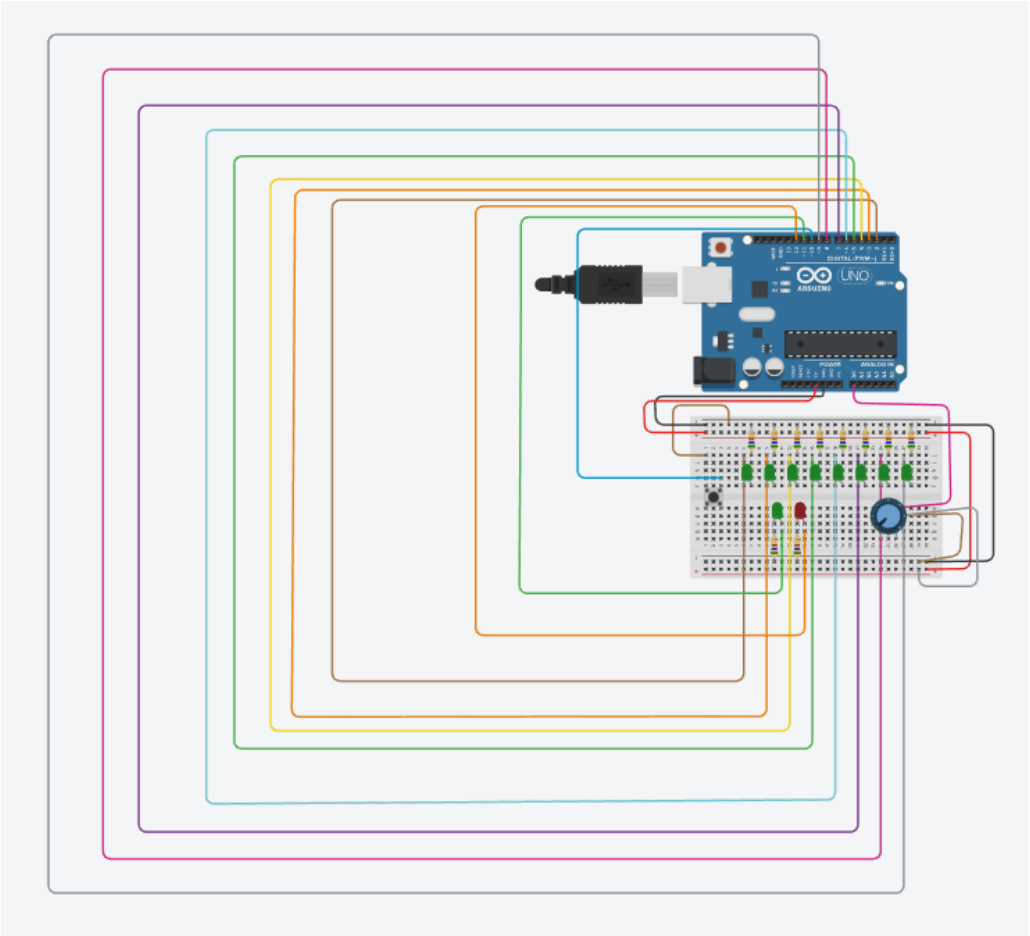
Design Decisions

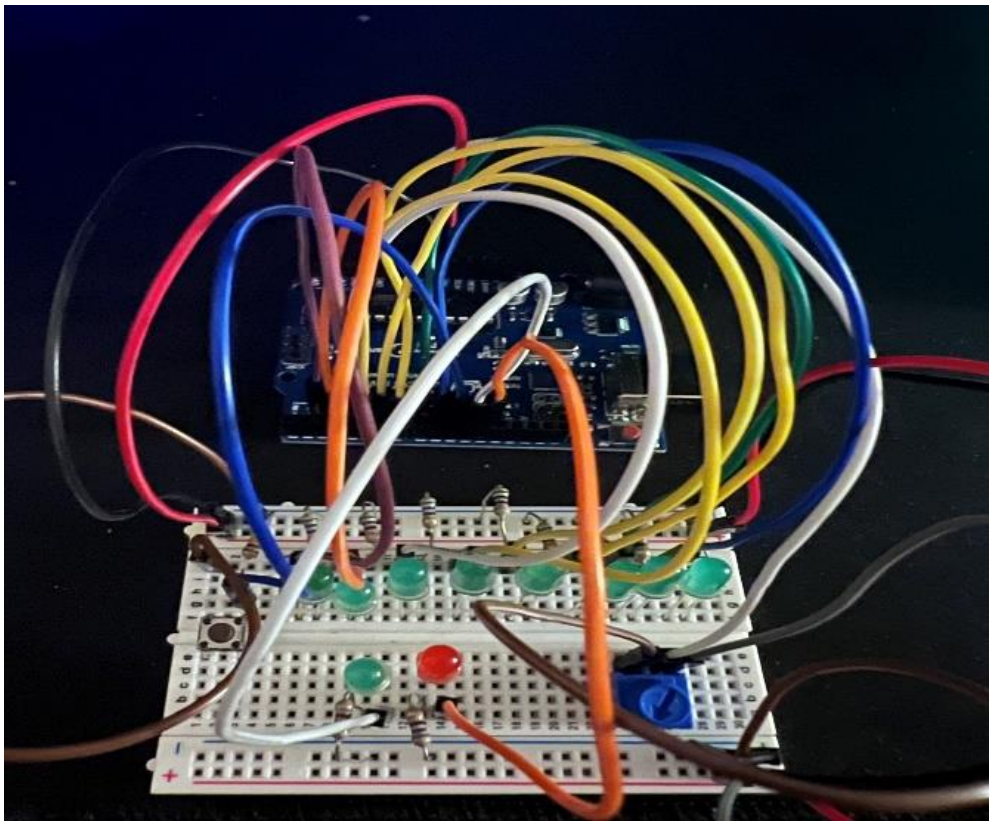
For the circuit I decided to use green LED's for the ships and wire each LED to pins starting from 2-9 on one half of the breadboard. To allow the user to focus on the ships easier I have the LED's in a line so scrolling through them won't be difficult for the user. On the other half of the breadboard I have connected the potentiometer and also the 2 LEDS to signal the hit or miss. At the other end of the board I have connected the push button to allow the user to twist the potentiometer with easy and press the button when they decide to shoot the ship.

Bread Board Layout



Zoomed out version:





Testing

```
3   const int greenLed = 11;
4   const int MAX_BOARD_SIZE= 8;
5   int sensorPin = A0;
6   int sensorValue;
7   int currentValue;
8   int previousValue;
9   int buttonPin = 10;
10  int buttonState;
11  int locationCells[] = {2,5,4};
12  int hitTracker[ MAX_BOARD_SIZE ][ 2 ] = {{2,0},{3,0},{4,0},{5,0},{6,0},{7,0},{8,0},{9,0}};
13  int numberOfHits = 0;
14
15  void setup() {
16    // put your setup code here, to run once:
17    pinMode(buttonPin, INPUT);
18    pinMode(sensorPin, INPUT);
19    pinMode(redLed,OUTPUT);
20      pinMode(greenLed,OUTPUT);
21   for ( int i = 0; i<8; i ++){
22     pinMode(ledPins[i],OUTPUT);
23   }
24   previousValue=analogRead(sensorPin);
25   previousValue=map(previousValue,0,1023,0,7);
26   Serial.begin(9600);
27  }
28
29  void loop() {
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM3')

```
current val: 5
current val: 4
current val: 5
current val: 4
current val: 5
current val: 4
current val: 5
current val: 4
current val: 5
current val: 4
current val: 5
current val: 4
current val: 3
```

Testing the potentiometer is being mapped correctly for each LED. Current value = Value potentiometer is creating once its turned

```
60        delay(150);
61        digitalWrite(ledPins[currentValue-1], LOW);
62        delay(150);
63        digitalWrite(ledPins[currentValue-1], HIGH);
64        delay(150);
65        digitalWrite(ledPins[currentValue-1], LOW);
66        digitalWrite(redLed,HIGH);
67        delay(500);
68        digitalWrite(redLed,LOW);
69          numberOfHits++;
70          Serial.print("numberOfhits: ");
71          Serial.println(numberOfHits);
72      }
73
74
75
76    //HIT
```

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on 'COM3')

```
current val: 2
current val: 3
0
BUTTON VAL: 0
Hit Tracker(0 not hit, 1 hit): 1
numberOfhits: 2
current val: 4
current val: 5
0
BUTTON VAL: 0
Hit Tracker(0 not hit, 1 hit): 1
numberOfhits: 3
current val: 6
current val: 7
```

Testing the Button. Whether the hits are being register and updated everytime the user presses the button – numberOfHits.

Testing the 2d Array Hit Tracker to see if the Array value hitTracker([currentLed][1]) specific value is being updated when the user shoots the LED

Testing Scenarios

Part A

- Checking if values are randomly generated for the ship
- Checking if the correct light is turned on when there is a hit
- Checking if the light stays on if it's a ship
- Checking if the red light turns on when the shot is a miss
- Checking if the green indicator light is turned on when the shot is hit
- Checking if the user can input a value greater than 8 or less than 1

Part B

- Turning the potentiometer cycles through the LED's correctly
- Pressing the button confirms the hit
- Checking If hit the red light is turned on if it's a miss
- Checking if the LED for the ship flashes when shot
- Checking if the green indicator light is turned on when the ship is hit
- Checking if the user can have more than 5 hits
- Checking if the ship positions are revealed if the user loses
- Checking if the game finishes if all ships are hit

Code A

```
const int ledPins[] = {2,3,4,5,6,7,8,9}; //LEDS

const int redLed = 12; //LED LIGHT TO MISS
const int greenLed = 11; //LED LIGHT FOR HIT
const int MAX_BOARD_SIZE= 8;  //MAX SIZE OF SHIPS
const int SHIP_SIZE = 3; //  AMOUNT OF SHIPS
int locationCells[SHIP_SIZE]; //SHIP LOCATIONS
int numberOfHits = 0;  //TRACKER FOR HITS




void setup() {
  // put your setup code here, to run once:
  for (int i = 0; i < MAX_BOARD_SIZE; i++) { //INITALISE ALL THE LIGHTS
    pinMode(ledPins[i], OUTPUT);
  }

  pinMode(redLed, OUTPUT); //INITALISE RED LED
  pinMode(greenLed, OUTPUT); // INITIALISE GREEN LED
  Serial.begin(9600);

  setLocationCells();

}

void loop() {


  String result = checkYourself(); //CHECK FOR LOCATION CELLS FOR TESTING
  Serial.println(result);
  delay(500);
  if (result == "kill") {  // CHECK IF ALL SHIPS ARE HIT

      digitalWrite(greenLed, HIGH);
      delay(250);
      digitalWrite(greenLed, LOW);



    delay(1000);
    exit(1);
  }
  if (result == "hit") {   //CHECK IF SHIP IS HIT
    digitalWrite(greenLed, HIGH);
```

```
    delay(500);
    digitalWrite(greenLed, LOW);
  }
  if (result == "miss") { //CHECK IF THE SHIP IS MISSED
    digitalWrite(redLed, HIGH);
    delay(500);
    digitalWrite(redLed, LOW);
  }
}

char getGuess() {  // FUNCTION TO FGET THE GUESS
  char readinput = '^'; // INITIALISE THE VARIABLE AND THE TYPE
  while ((readinput < '1') || (readinput > '8')) { //WHILE LOOP TO RUN IF THE USER DOES
NOT ENTER A CORRECT VALUE
    Serial.print("Enter a guess (1-8),  ");
    while (Serial.available() == 0) { }
    readinput = Serial.read();
    Serial.print("You have entered: ");
    Serial.println(readinput);
    if ((readinput < '1') || (readinput > '8')) { //PRINT THE INVALID
      Serial.println("Invalid input");
    }
  }
  return readinput;
}

void setLocationCells() {
  int startLocation = random(0, MAX_BOARD_SIZE - SHIP_SIZE-1);
  Serial.print("Location Cells generated are: ");


  //ASSIGN LOCATION
  for (int i = 0; i < SHIP_SIZE; i++) {
    locationCells[i] = startLocation + i;
    Serial.print(locationCells[i]+1);
  }
  Serial.println();
}

String checkYourself() {
  //GET THE USERS INPUT
  char userGuess = getGuess();
  //char userGuess = '1';

  String result = "miss";

  int guess = (userGuess - '0');
  guess -= 1;
```

```
  for (int i = 0; i < SHIP_SIZE; i++) {
    //if the user's guess matches
    if ((guess) == locationCells[i]) {
      numberOfHits++;
      locationCells[i] = -1;
      digitalWrite(ledPins[guess], HIGH);
      if (numberOfHits == SHIP_SIZE) {
        result = "kill";
      } else {
        result = "hit";
      }
      break;
    }
  }
  return result;
}
```

Code B

```
const int ledPins[] = { 2, 3, 4, 5, 6, 7, 8, 9 }; //STORING LED PINS IN ARRAY
const int redLed = 12; //RED LED FOR MISS
const int greenLed = 11; // GREEN LED FOR HIT
const int MAX_BOARD_SIZE = 8;
const int SHIP_SIZE = 3;
int sensorPin = A0; // PIN FOR POTENTIOMETER
int sensorValue; // STORE VALUE OF POTENTIOMETER

// VARIABLES TO ALLOW THE POTENTIOMETER TO SCROLL THROUGH THE LEDS
int currentValue;
int previousValue;

int buttonPin = 10; // BUTTON PIN VARIABLE
int buttonState; // STORE THE BUTTON VALUE
int locationCells[SHIP_SIZE]; // STORE THE POSITIONS OF THE SHIPS
int hitTracker[MAX_BOARD_SIZE][2] = { { 2, 0 }, { 3, 0 }, { 4, 0 }, { 5, 0 }, { 6, 0 },
{ 7, 0 }, { 8, 0 }, { 9, 0 } }; // 2D ARRAY FOR CHECKING IF SHIP HAS BEEN HIT
int numberOfHits = 0; // TRACKING HOW MANY HITS THE USER HAS USED

int hitMade; // CHECKING IF USER HAS SHOT
//int difficulty;
int shipHit = 0; //TRACKING HOW MANY OF THE SHIPS HIT
```

```
void setup() {
  // put your setup code here, to run once:
  //INITALISING LEDS AND BUTTONS
  pinMode(buttonPin, INPUT);
  pinMode(sensorPin, INPUT);
  pinMode(redLed, OUTPUT);
  pinMode(greenLed, OUTPUT);

  for (int i = 0; i < 8; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
  //STORING PREVIOUS VALUE OF POTENTIOMETER AND MAPPING TO ALLOW SCROLLING THROUGH LEDS
  previousValue = analogRead(sensorPin);
  previousValue = map(previousValue, 0, 1023, 0, 9);
  Serial.begin(9600);
  setLocationCells(); //GENERATE VALUES OF THE SHIP POSITIONS
//  difficulty = getDifficulty();
}

void loop() {
  //STORING CURRENT VALUE TO MAKE SURE PREVIOUS LED IS TURNED OFF AND NOT ALL LIGHTS
ARE TURNED ON
  currentValue = analogRead(sensorPin);
  currentValue = map(currentValue, 0, 1023, 0, 9);
  while (currentValue != previousValue) {
    for (int i = 0; i < 8; i++) {
      digitalWrite(ledPins[i], LOW);
    }
    if (currentValue == previousValue) {

    } else if ((hitTracker[currentValue - 1][1]) == 0 && (shipHit != 3 || numberOfHits
!= 5 )) { //CHECK IF THE LED HAS BEEN SHOT, IF IT HAS NOT BEEN SHOT LIGHT IT. IF IT HAS
BEEN SHOT SKIP IT
      Serial.print("current val: ");
      Serial.println(currentValue);
      digitalWrite(ledPins[currentValue - 1], HIGH);
    }
    previousValue = currentValue;
  }

  buttonState = digitalRead(buttonPin); //GETTING THE BUTTON VALUE


  if (buttonState == LOW && (hitTracker[currentValue - 1][1] == 0) ) { //CHECKING IF
THE BUTTON IS PRESSED AND THE LED HAS NOT BEEN HIT ALREADY AND THE USER HAS NOT SHOT 5
TIMES
    Serial.println(hitTracker[currentValue - 1][1]);
```

```
    Serial.print("BUTTON VAL: ");
    Serial.println(buttonState);
    hitTracker[currentValue - 1][1] = 1;  //SAVING INTO THE 2D ARRAY THAT THE LED HAS
BEEN HIT
    Serial.print("Hit Tracker(0 not hit, 1 hit): ");
    Serial.println(hitTracker[currentValue - 1][1]);
    delay(150);

    //FLASH THE LED
    digitalWrite(ledPins[currentValue - 1], LOW);
    delay(150);
    digitalWrite(ledPins[currentValue - 1], HIGH);
    delay(150);
    digitalWrite(ledPins[currentValue - 1], LOW);
    delay(150);
    digitalWrite(ledPins[currentValue - 1], HIGH);
    delay(150);
    digitalWrite(ledPins[currentValue - 1], LOW);

    //FOR LOOP TO CHECK IF THE SHIP IS A CORRECT SHIP
    for (int l = 0; l < 4; l++) {
      hitMade = 0;
      if (ledPins[currentValue - 1] == locationCells[l]) {
        Serial.print(locationCells[l]);
        Serial.print(ledPins[currentValue - 1]);
        madeHit();  //GREEN LIGHT FUNCTION TO SHOW IT IS A HITT
        l = 3;
        hitMade = 1; //SAVING THE VALUE THAT THE HIT HAS BEEN MADE SO THAT
        shipHit++;
      } else if ((l == 3) && (hitMade == 0)) {
        missedHit(); //RED LIGHT FUNCTION TO SHOW IT IS A MISS
      }
    }
  }

  //IF THE USER HAS MADE 5 GUEESES OR ALL SHIPS HAVE BEEN SHOT
 else if ((numberOfHits > 4) || (shipHit == 3)){
   currentValue==previousValue;
   sensorPin=0;
   for(int m =0; m<4; m++){
      digitalWrite(locationCells[m],HIGH);
   }
 }



}
```

```
//FUNCTION TO TURH THE RED LIGHT ON IF ITS A MISS AND STORE THE VALUE IN THE 2D ARRAY
THAT A SHIP HAS BEEN HIT
void missedHit() {
  digitalWrite(redLed, HIGH);
  delay(500);
  digitalWrite(redLed, LOW);
  numberOfHits++;
  Serial.print("numberOfhits: ");
  Serial.println(numberOfHits);
  hitTracker[currentValue - 1][1] = 1;
}

//FUNCTION TO TURN THE GREEN LIGHT ON IF ITS A HIT AND STORE THE VALUE IN THE 2D ARRAY
void madeHit() {
  digitalWrite(greenLed, HIGH);
  delay(500);
  digitalWrite(greenLed, LOW);
  numberOfHits++;
  Serial.print("numberOfhits: ");
  Serial.println(numberOfHits);
  hitTracker[currentValue - 1][1] = 1;
}




void setLocationCells(){
    int startLocation = random(0, MAX_BOARD_SIZE - SHIP_SIZE-1);
    Serial.print("Location Cell generated are: ");
    //
    for (int i = 0; i<SHIP_SIZE; i ++){
      locationCells[i] = startLocation + i;
      Serial.print(locationCells[i]);
    }
    Serial.println();
}


// char getDifficulty() {
//   char readinput = '^';
//   while ((readinput < '3') || (readinput > '8')) {
//     Serial.print("Enter difficulty 3-8, 3 = 3 shots),  ");
//     while (Serial.available() == 0) { }
//     readinput = Serial.read();
//     Serial.print("You chose : ");
//     Serial.println(readinput);
//     if ((readinput < '3') || (readinput > '8')) {
//       Serial.println("Invalid input");
//     }
```

```
//   }
//   return readinput;
// }
```

Shortcomings

- Once the game is over the potentiometer does not turn any lights as, as desired however the last light if the potentiometer value has reached it. It will turn on
- Sometimes due to the potentiometers value not being the same even without moving it sometimes the led will switch/flash back and forth without user touching the potentiometer.