Name: Shaan Yadav
NetID: ay140
Honor Code: *I have adhered to the Duke Community Standard in completing this assignment.*

1. Code for Simple Line Following

• The QTI code for simple line following from the exploration (following a line, regard-less of hashmarks).

```
//Pins for QTI connections on board
#define leftQTI 51
#define middleQTI 53
#define rightQTI 52

#include <Servo.h>                            // Include servo library
Servo servoLeft;                              // Declare left servo signal
Servo servoRight;

// Define pins for built-in RGB LED
#define redpin 45
#define greenpin 46
#define bluepin 44

int hashCount = 0;
int reds[5] = {0, 0, 255, 255, 100};
int greens[5] = {255, 0, 0, 255, 255};
int blues[5] = {255, 255, 255, 0, 0};

void setup() {
 Serial.begin(9600); //start the serial monitor so we can view the output

 servoLeft.attach(12);                       // Attach left signal to P13
 servoRight.attach(11);                       // Attach left signal to P12

 servoLeft.writeMicroseconds(1500);          // 1.5 ms stay still sig, pin 13
 servoRight.writeMicroseconds(1500);         // 1.5 ms stay still sig, pin 12

 pinMode(redpin, OUTPUT);
 pinMode(greenpin, OUTPUT);
 pinMode(bluepin, OUTPUT);

 // start with light off
 analogWrite(redpin, 255);
 analogWrite(greenpin, 255);
 analogWrite(bluepin, 255);
}
void loop() {
 int lQTI = rcTime(leftQTI);
 int mQTI = rcTime(middleQTI);
 int rQTI = rcTime(rightQTI);

 // Serial.println("left: " + String(lQTI) + " middle: " + String(mQTI) + " right: " +
String(rQTI));

 int state = 4*(lQTI < 200) + 2*(mQTI < 200) + (rQTI < 150);
 // Serial.println(state);

 switch(state) {
   // not on line
   case 7:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1550);
     break;
   // right sensor --> turn right
   case 6:
     servoRight.writeMicroseconds(1550);
     servoLeft.writeMicroseconds(1550);
     delay(30);
     break;
   // middle sensor --> go forward
   case 5:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1550);
     break;
   // middle + right sensor --> turn right, slight
   case 4:
     servoRight.writeMicroseconds(1550);
     servoLeft.writeMicroseconds(1550);
     delay(40);
     break;
   // left sensor --> turn left
   case 3:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1450);
     delay(25);
     break;
   // left + middle sensor --> turn left, slight
   case 1:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1550);
     break;
   // everything else
   default:
     break;
 }
 delay(50);
}

//Defines funtion 'rcTime' to read value from QTI sensor
// From Ch. 6 Activity 2 of Robotics with the BOE Shield for Arduino
long rcTime(int pin)
{
 pinMode(pin, OUTPUT);     // Sets pin as OUTPUT
 digitalWrite(pin, HIGH); // Pin HIGH
 delay(1);                 // Waits for 1 millisecond
 pinMode(pin, INPUT);      // Sets pin as INPUT
 digitalWrite(pin, LOW);   // Pin LOW
 long time = micros();     // Tracks starting time
 while(digitalRead(pin)); // Loops while voltage is high
 time = micros() - time;   // Calculate decay time
 return time;              // Return decay time
}
```

2. Code for Line Following with Hashmarks

• The code for following lines, stopping at hashmarks, turning on the appropriate RGB LED colors, and turning them off before continuing. Code should show that the ’bot does not move on past the final hash mark. Make sure your TA has marked your group as finishing IDC Checkpoint 1.

```
//Pins for QTI connections on board
#define leftQTI 51
#define middleQTI 53
#define rightQTI 52

#include <Servo.h>                             // Include servo library
Servo servoLeft;                               // Declare left servo signal
Servo servoRight;

// Define pins for built-in RGB LED
#define redpin 45
#define greenpin 46
#define bluepin 44

int hashCount = 0;
int reds[5] = {0, 0, 255, 255, 100};
int greens[5] = {255, 0, 0, 255, 255};
int blues[5] = {255, 255, 255, 0, 0};

void setup() {
 Serial.begin(9600); //start the serial monitor so we can view the output

 servoLeft.attach(12);                        // Attach left signal to P13
 servoRight.attach(11);                        // Attach left signal to P12

 servoLeft.writeMicroseconds(1500);           // 1.5 ms stay still sig, pin 13
 servoRight.writeMicroseconds(1500);          // 1.5 ms stay still sig, pin 12

 pinMode(redpin, OUTPUT);
 pinMode(greenpin, OUTPUT);
 pinMode(bluepin, OUTPUT);

 // start with light off
 analogWrite(redpin, 255);
 analogWrite(greenpin, 255);
 analogWrite(bluepin, 255);
```

```
}
void loop() {
 int lQTI = rcTime(leftQTI);
 int mQTI = rcTime(middleQTI);
 int rQTI = rcTime(rightQTI);

 // Serial.println("left: " + String(lQTI) + " middle: " + String(mQTI) + " right: " +
String(rQTI));

 int state = 4*(lQTI < 200) + 2*(mQTI < 200) + (rQTI < 150);
 // Serial.println(state);

 switch(state) {
   // not on line
   case 7:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1550);
     break;
   // right sensor --> turn right
   case 6:
     servoRight.writeMicroseconds(1550);
     servoLeft.writeMicroseconds(1550);
     delay(30);
     break;
   // middle sensor --> go forward
   case 5:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1550);
     break;
   // middle + right sensor --> turn right, slight
   case 4:
     servoRight.writeMicroseconds(1550);
     servoLeft.writeMicroseconds(1550);
     delay(40);
     break;
   // left sensor --> turn left
   case 3:
     servoRight.writeMicroseconds(1450);
     servoLeft.writeMicroseconds(1450);
     delay(25);
     break;
   // left + middle sensor --> turn left, slight
```

```
  case 1:
    servoRight.writeMicroseconds(1450);
    servoLeft.writeMicroseconds(1450);
    delay(25);
    break;
  // at HASHMARK --> stop, forward
  case 0:
    servoRight.writeMicroseconds(1500);
    servoLeft.writeMicroseconds(1500);

    // turn on light
    // analogWrite(redpin, hashmarkColours[hashCount][0]);
    // analogWrite(greenpin, hashmarkColours[hashCount][1]);
    // analogWrite(bluepin, hashmarkColours[hashCount][2]);

    analogWrite(redpin, reds[hashCount % 5]);
    analogWrite(greenpin, greens[hashCount % 5]);
    analogWrite(bluepin, blues[hashCount % 5]);

    delay(2000);

    hashCount+=1;


    // turn light off
    analogWrite(redpin, 255);
    analogWrite(greenpin, 255);
    analogWrite(bluepin, 255);

    servoRight.writeMicroseconds(1300);
    servoLeft.writeMicroseconds(1700); // right 13 is forward, left 17 is forward

    delay(500);

    break;
  // everything else
  default:
    break;
 }
 delay(50);
}
```

```
//Defines funtion 'rcTime' to read value from QTI sensor
// From Ch. 6 Activity 2 of Robotics with the BOE Shield for Arduino
long rcTime(int pin)
{
 pinMode(pin, OUTPUT);     // Sets pin as OUTPUT
 digitalWrite(pin, HIGH); // Pin HIGH
 delay(1);                 // Waits for 1 millisecond
 pinMode(pin, INPUT);      // Sets pin as INPUT
 digitalWrite(pin, LOW);   // Pin LOW
 long time = micros();     // Tracks starting time
 while(digitalRead(pin)); // Loops while voltage is high
 time = micros() - time;   // Calculate decay time
 return time;              // Return decay time
}
```

3. Reflection Paragraph

• Brief paragraph reflecting on your experience with this lab.

I really enjoyed this lab, especially iteratively debugging and improving the bots to follow the line. I feel like I learnt a lot by having a final objective and just working towards the final goal with my partner was an extremely helpful learning experience.