CODE EXAMPLES FOR MESH

1.

```
#include <Arduino.h>
// IR Sensors
int sensor1 = 2;      // Left most sensor
int sensor2 = 3;
int sensor3 = 4;
int sensor4 = 5;      // Right most sensor

// Initial Values of Sensors
int sensor[4] = {0, 0, 0, 0};

// Motor Variables
int ENA = 6;
int motorInput1 = 7;
int motorInput2 = 8;
int motorInput3 = 9;
int motorInput4 = 10;
int ENB = 11;

//Initial Speed of Motor
int initial_motor_speed = 140;

// Output Pins for Led
int ledPin1 = A3;
int ledPin2 = A4;

// PID Constants
float Kp = 25;
float Ki = 0;
float Kd = 15;

float error = 0, P = 0, I = 0, D = 0, PID_value = 0;
float previous_error = 0, previous_I = 0;

int flag = 0;

void setup()
{
  pinMode(sensor1, INPUT);
  pinMode(sensor2, INPUT);
```

```
  pinMode(sensor3, INPUT);
  pinMode(sensor4, INPUT);

  pinMode(motorInput1, OUTPUT);
  pinMode(motorInput2, OUTPUT);
  pinMode(motorInput3, OUTPUT);
  pinMode(motorInput4, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(ENB, OUTPUT);

  pinMode(ledPin1, OUTPUT);
  pinMode(ledPin2, OUTPUT);

  digitalWrite(ledPin1, LOW);
  digitalWrite(ledPin2, LOW);

  Serial.begin(9600);                 //setting serial monitor at a default baund rate of 9600
  delay(500);
  Serial.println("Started !!");
  delay(1000);
}

void read_sensor_values()
{
  sensor[0] = !digitalRead(sensor1);
  sensor[1] = !digitalRead(sensor2);
  sensor[2] = !digitalRead(sensor3);
  sensor[3] = !digitalRead(sensor4);

  /*
    Serial.print(sensor[0]);
    Serial.print("\t");
    Serial.print(sensor[1]);
    Serial.print("\t");
    Serial.print(sensor[2]);
    Serial.print("\t");
    Serial.println(sensor[3]);*/

  if ((sensor[0] == 1) && (sensor[1] == 0) && (sensor[2] == 0) && (sensor[3] == 0))
    error = 3;
  else if ((sensor[0] == 1) && (sensor[1] == 1) && (sensor[2] == 0) && (sensor[3] == 0))
    error = 2;
  else if ((sensor[0] == 0) && (sensor[1] == 1) && (sensor[2] == 0) && (sensor[3] == 0))
    error = 1;
```

```
  else if ((sensor[0] == 0) && (sensor[1] == 1) && (sensor[2] == 1) && (sensor[3] == 0))
    error = 0;
  else if ((sensor[0] == 0) && (sensor[1] == 0) && (sensor[2] == 1) && (sensor[3] == 0))
    error = -1;
  else if ((sensor[0] == 0) && (sensor[1] == 0) && (sensor[2] == 1) && (sensor[3] == 1))
    error = -2;
  else if ((sensor[0] == 0) && (sensor[1] == 0) && (sensor[2] == 0) && (sensor[3] == 1))
    error = -3;
  else if ((sensor[0] == 1) && (sensor[1] == 1) && (sensor[2] == 1) && (sensor[3] == 0)) // Turn
robot left side
    error = 100;
  else if ((sensor[0] == 0) && (sensor[1] == 1) && (sensor[2] == 1) && (sensor[3] == 1)) // Turn
robot right side
    error = 101;
  else if ((sensor[0] == 0) && (sensor[1] == 0) && (sensor[2] == 0) && (sensor[3] == 0)) // Make U
turn
    error = 102;
  else if ((sensor[0] == 1) && (sensor[1] == 1) && (sensor[2] == 1) && (sensor[3] == 1)) // Turn
left side or stop
    error = 103;
}
void forward()
{
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, LOW);
  digitalWrite(motorInput2, HIGH);
  digitalWrite(motorInput3, LOW);
  digitalWrite(motorInput4, HIGH);
}

void reverse()
{
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, HIGH);
  digitalWrite(motorInput2, LOW);
  digitalWrite(motorInput3, HIGH);
  digitalWrite(motorInput4, LOW);
}
void right()
{
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, LOW);
  digitalWrite(motorInput2, HIGH);
  digitalWrite(motorInput3, LOW);
```

```
  digitalWrite(motorInput4, LOW);
}
void left()
{
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, LOW);
  digitalWrite(motorInput2, LOW);
  digitalWrite(motorInput3, LOW);
  digitalWrite(motorInput4, HIGH);
}
void sharpRightTurn() {
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, LOW);
  digitalWrite(motorInput2, HIGH);
  digitalWrite(motorInput3, HIGH);
  digitalWrite(motorInput4, LOW);
}
void sharpLeftTurn() {
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, HIGH);
  digitalWrite(motorInput2, LOW);
  digitalWrite(motorInput3, LOW);
  digitalWrite(motorInput4, HIGH);
}
void stop_bot()
{
  /*The pin numbers and high, low values might be different depending on your connections */
  digitalWrite(motorInput1, LOW);
  digitalWrite(motorInput2, LOW);
  digitalWrite(motorInput3, LOW);
  digitalWrite(motorInput4, LOW);
}

void calculate_pid()
{
  P = error;
  I = I + previous_I;
  D = error - previous_error;

  PID_value = (Kp * P) + (Ki * I) + (Kd * D);

  previous_I = I;
  previous_error = error;
}
```

```
void motor_control()
{
  // Calculating the effective motor speed:
  int left_motor_speed = initial_motor_speed - PID_value;
  int right_motor_speed = initial_motor_speed + PID_value;

  // The motor speed should not exceed the max PWM value
  left_motor_speed = constrain(left_motor_speed, 0, 255);
  right_motor_speed = constrain(right_motor_speed, 0, 255);

  /*Serial.print(PID_value);
    Serial.print("\t");
    Serial.print(left_motor_speed);
    Serial.print("\t");
    Serial.println(right_motor_speed);*/

  analogWrite(ENA, left_motor_speed); //Left Motor Speed
  analogWrite(ENB, right_motor_speed - 30); //Right Motor Speed

  //following lines of code are to make the bot move forward
  forward();
}
void loop()
{
  read_sensor_values();
  Serial.print(error);
  if (error == 100) {              // Make left turn untill it detects straight path
    //Serial.print("\t");
    //Serial.println("Left");
    do {
      read_sensor_values();
      analogWrite(ENA, 110); //Left Motor Speed
      analogWrite(ENB, 90); //Right Motor Speed
      sharpLeftTurn();
    } while (error != 0);

  } else if (error == 101) {          // Make right turn in case of it detects only right path (it will go into
forward direction in case of staright and right "|--")
                                // untill it detects straight path.
    //Serial.print("\t");
    //Serial.println("Right");
    analogWrite(ENA, 110); //Left Motor Speed
    analogWrite(ENB, 90); //Right Motor Speed
```

```
    forward();
    delay(200);
    stop_bot();
    read_sensor_values();
    if (error == 102) {
      do {
        analogWrite(ENA, 110); //Left Motor Speed
        analogWrite(ENB, 90); //Right Motor Speed
        sharpRightTurn();
        read_sensor_values();
      } while (error != 0);
    }
  } else if (error == 102) {        // Make left turn untill it detects straight path
    //Serial.print("\t");
    //Serial.println("Sharp Left Turn");
    do {
      analogWrite(ENA, 110); //Left Motor Speed
      analogWrite(ENB, 90); //Right Motor Speed
      sharpLeftTurn();
      read_sensor_values();
      if (error == 0) {
        stop_bot();
        delay(200);
      }
    } while (error != 0);
  } else if (error == 103) {        // Make left turn untill it detects straight path or stop if dead end
reached.
    if (flag == 0) {
      analogWrite(ENA, 110); //Left Motor Speed
      analogWrite(ENB, 90); //Right Motor Speed
      forward();
      delay(200);
      stop_bot();
      read_sensor_values();
      if (error == 103) {     /* Dead End Reached, Stop! */
        stop_bot();
        digitalWrite(ledPin1, HIGH);
        digitalWrite(ledPin2, HIGH);
        flag = 1;
      } else {        /* Move Left */
        analogWrite(ENA, 110); //Left Motor Speed
        analogWrite(ENB, 90); //Right Motor Speed
        sharpLeftTurn();
        delay(200);
```

```
      do {
        //Serial.print("\t");
        //Serial.println("Left Here");
        read_sensor_values();
        analogWrite(ENA, 110); //Left Motor Speed
        analogWrite(ENB, 90); //Right Motor Speed
        sharpLeftTurn();
      } while (error != 0);
    }
   }
  } else {
   calculate_pid();
   motor_control();
  }
}
```

.2.

```
//Program to

//define the arduino pin numbers connected to motor

int L1 = 2;

int L2 = 3;

int r1 = 4;

int r2 = 5;

int en1 = 9;

int en2 = 10;

// array to store five sensor values

int a[5] = {0, 0, 0, 0, 0};

int last_proportional = 0;

int integral = 0;

//function prototype of different functions

char select_turn(unsigned char found_left, unsigned char found_right, unsigned char found_st);

int mod(int v);
```

```
int set_motors(int a, int b);
void turn(char dir);
void PID();
int right = 0;
int left = 0;
//setup function to define the pin Mode is input or output
void setup()
{
  pinMode(L1, OUTPUT);
  pinMode(L2, OUTPUT);
  pinMode(r1, OUTPUT);
  pinMode(r2, OUTPUT);
  pinMode(en1, OUTPUT);
  pinMode(en2, OUTPUT);
  Serial.begin(9600);
}
//infinite loop function
void loop()
{
  PID();  // calling the pid function

  unsigned char found_left = 0;
  unsigned char found_right = 0;
  unsigned char found_st = 0;
  readline();  //calling readline function to read sensor values
  if(a[0] == HIGH)  // condition check for left senor is high or not
  {
```

```
   found_left = 1;
  }
  else if(a[0] == HIGH && a[1] == HIGH && a[2] == HIGH) // condition check  for left sensor is high
  {
   found_left = 1;
  }
  else if(a[4] == HIGH && a[3] == HIGH)   // condition check for right sensor is high or not
  {
   found_right == 1;
  }
  if(a[2]== HIGH)    // condition check for middle sensor is high or not
  {
   found_st = 1;
  }
  unsigned char dir;
  dir = select_turn(found_left, found_right, found_st);
  turn(dir);
}
void PID()     // PID function definition
{
// Variable initialization
  int i;
  int power_difference = 0;
  float Kp, Ki, Kd;
  unsigned int position;
  int derivative,proportional;
  while(1)
```

```
{
  position = readline();            //reading the sensor value and storing in variable position
   Serial.println("position = ");
  Serial.println(position);
   //replace value 2000 with your position by placing your robot at the dead centre and read the value
  proportional = ((int)position - 2000);       derivative = proportional - last_proportional;
  integral = integral+proportional;
  last_proportional = proportional;
// value of kp ki kd, you have to make changes to make your robot move without oscillation (hit and trial method)
  Kp = 0.08;
  Ki = 0.0002;
  Kd = 1.0;
//formula for pid to calculate error (i.e power difference)
  power_difference = proportional * Kp + integral * Ki + derivative *Kd;
  const int max = 180;  //setting the maximum speed of motor
  if(power_difference>max)
   power_difference = max;
  if(power_difference < -max)
    power_difference = (-1*max);
    if(power_difference < 0)
    {
      //set_motors(max, max-power_difference);
      set_motors(max+power_difference, max);
    }
    else
    {
      //set_motors(max+power_difference, max);
```

```
        set_motors(max, max-power_difference);
      }
      readline();
      if(a[0] == LOW && a[1] == LOW && a[2] == LOW && a[3] == LOW && a[4] == LOW)
      return;
      else if(a[0]== HIGH || a[4] == HIGH)
      return;
  }
}
int readline()
{
//Read the sensor values
  a[0] = digitalRead(6);
  a[1] = digitalRead(7);
  a[2] = digitalRead(11);
  a[3] = digitalRead(12);
  a[4] = digitalRead(13);
  int v;
//calculating the average value of sensors
  v = (4000*a[0] + 3000*a[1] + 2000*a[2] + 1000*a[3] + 0*a[4])/(a[0] + a[1] + a[2] + a[3] + a[4]);
  Serial.println(v);
  return v;
}
char select_turn(unsigned char found_left, unsigned char found_right, unsigned char found_st)
{
  if(found_left == 1)
  return 'L';
```

```
  if(found_st == 1)
  return 'S';
  if(found_right == 1)
  return 'R';
  else
  return 'B';
}
void turn(char dir)
{
  switch(dir)
  {
    case 'L':
    set_motors(0,150);
    delay(350);
    break;
    case 'R':
    set_motors(150,0);
    delay(350);
    break;
     case 'B':
    set_motors(-150,150);
    delay(200);
    break;
    case 'S':
    break;
  }
}
```

```
int set_motors(int l, int r)
{
  Serial.println(r);
  Serial.println(l);
  if(l>=0 && r>=0)
  {
    analogWrite(en1,mod(l));
    analogWrite(en2, mod(r));
    digitalWrite(L1, HIGH);
    digitalWrite(L2, LOW);
    digitalWrite(r1, HIGH);
    digitalWrite(r2, LOW);
  }
 else if(l<=0 && r>=0)
  {
    analogWrite(en1,mod(l));
    analogWrite(en2, mod(r));
    digitalWrite(L1, LOW);
    digitalWrite(L2, HIGH);
    digitalWrite(r1, HIGH);
    digitalWrite(r2, LOW);
  }
  else if(l>=0 && r<=0)
  {
    analogWrite(en1,mod(l));
    analogWrite(en2, mod(r));
   digitalWrite(L1, HIGH);
   digitalWrite(L2, LOW);
 digitalWrite(r1, LOW);
```

```
    digitalWrite(r2, HIGH);
  }
  else if(l==0 && r==0)
  {
    analogWrite(en1,0);
    analogWrite(en2, 0);
    digitalWrite(L1, LOW);
    digitalWrite(L2, LOW);
    digitalWrite(r1, LOW);
    digitalWrite(r2, LOW);
}
}
int mod(int v)
{
  if(v<0)
  return -1*v;
  else if(v>0)
  return v;
}
```

###

3.

```
#define l1 5
#define l2 2
#define r1 4
#define r2 3

#define enl 9
#define enr 10

#define mode 12
```

```
String path = "";

int a[6];
int last_proportional = 0;
int integral = 0;
int spd = 200;
int c = 0;
float distance;



char select_turn(unsigned char found_left, unsigned char found_right, unsigned char found_st);
int mod(int v); // Modulus Function
int set_motors(int a, int b);
void turn(char dir);
void PID();

int right = 0;
int left = 0;


//setup function to define the pin Mode is input or output
void setup()
{
  pinMode(l1, OUTPUT);
  pinMode(l2, OUTPUT);
  pinMode(r1, OUTPUT);
  pinMode(r2, OUTPUT);
  pinMode(enl, OUTPUT);
  pinMode(enr, OUTPUT);
  pinMode(mode, INPUT);

  Serial.begin(9600);
}


int readline() {
  a[0] = digitalRead(A5);
  a[1] = digitalRead(A4);
  a[2] = digitalRead(A3);
  a[3] = digitalRead(A2);
  a[4] = digitalRead(A1);
  a[5] = digitalRead(A0);
  int v;
  v = (5000*a[0] + 4000*a[1] + 3000*a[2] + 2000*a[3] + 1000*a[4] + 0*a[5])/
      (a[0] + a[1] + a[2] + a[3] + a[4] + a[5]);
      Serial.println("from readline v:");
      Serial.println(v);
  return v;
```

```
}

void turn(char dir) { //Turning setup
  switch(dir) {
    case 'L':
      set_motors(-spd, spd);
      delay(90);
      break;
    case 'R':
      set_motors(spd, -spd);
      delay(90);
      break;
    case 'B':
      set_motors(spd, -spd);
      delay(160);
      break;
    case 'S':
      break;
  }
}

int set_motors(int l, int r) { // Motor setup
  if(l > 0 && r > 0)  {
    analogWrite(enl, mod(l));
    analogWrite(enr, mod(r));

    digitalWrite(l1, LOW);
    digitalWrite(l2, HIGH);

    digitalWrite(r1, LOW);
    digitalWrite(r2, HIGH);
  }

  else if(l < 0 && r > 0) {
    analogWrite(enl, mod(l));
    analogWrite(enr, mod(r));

    digitalWrite(l1, HIGH);
    digitalWrite(l2, LOW);

    digitalWrite(r1, LOW);
    digitalWrite(r2, HIGH);
  }

  else if(l > 0 && r < 0) {
    analogWrite(enl, mod(l));
    analogWrite(enr, mod(r));

    digitalWrite(l1, LOW);
```

```
    digitalWrite(l2, HIGH);

    digitalWrite(r1, HIGH);
    digitalWrite(r2, LOW);
  }

  else if(l == 0 && r == 0) {
    analogWrite(enl, 0);
    analogWrite(enr, 0);

    digitalWrite(l1, HIGH);
    digitalWrite(l2, HIGH);

    digitalWrite(r1, HIGH);
    digitalWrite(r2, HIGH);
  }
}


int mod(int v)
{
  if(v<0)
    return -1*v;
  else if(v>0)
    return v;
}



char select_turn(unsigned char found_left, unsigned char found_right, unsigned char found_st)
{
    if(digitalRead(mode)){
        char r = path[0];
        path = path.substring(1, path.length()-1);
        return r;
    }
    else{
        if(found_left == 1){
            path += 'L';
            return 'L';
        }
        if(found_st == 1){
            path += 'S';
            return 'S';
        }
        if(found_right == 1){
            path += 'R';
            return 'R';
        }
```

```
        else{
            path += 'B';
            return 'B';
        }
    }
}


void PID() {
  int i;       // Control function
  int power_difference = 0;
  float Kp, Ki, Kd;
  unsigned int position;
  int derivative, proportional;
  while(1) {
    position = readline();
    Serial.println(position);
    proportional = ((int)position - 3000);

    derivative = proportional - last_proportional;
    integral = integral+proportional;

    last_proportional = proportional;
    // use the tutorial to set initial values of Kp, Ki, and Kd
    Kp = 5.5;
    Ki = 1;
    Kd = 0;

    power_difference = proportional*Kp + integral*Ki + derivative*Kd;
    const int max = spd/2 + 30;
    if(power_difference > max)
     power_difference = max;
    if(power_difference < -max)
     power_difference = (-1*max);

    if(power_difference < 0)  //left
     set_motors(max+power_difference, max);
    else  //right
     set_motors(max, max-power_difference);

    readline();

    if(a[0] == HIGH && a[1]==HIGH && a[2] == HIGH && a[3] == HIGH && a[4] == HIGH && a[5] == HIGH)
      return;
    else if(a[0] == LOW || a[5] == LOW)
      return;
  }
}
```

```
void findPath(){
   int changes = 1;
   while(changes){
      changes = 0;
      if (path.indexOf("LBR") != -1){
         path.replace("LBR", "B");
         changes++;
      }
      if (path.indexOf("LBS") != -1){
         path.replace("LBS", "R");
         changes++;
      }
      if (path.indexOf("RBL") != -1){
         path.replace("RBL", "B");
         changes++;
      }
      if (path.indexOf("SBL") != -1){
         path.replace("SBL", "R");
         changes++;
      }
      if (path.indexOf("SBS") != -1){
         path.replace("SBS", "B");
         changes++;
      }
      if (path.indexOf("LBL") != -1){
         path.replace("LBL", "S");
         changes++;
      }

   }
}


void loop()
{
  PID();  // calling the pid function

  unsigned char found_left = 0;
  unsigned char found_right = 0;
  unsigned char found_st = 0;

  readline();  //calling readline function to read sensor values

  if(a[0] == HIGH)  // condition check for left senor is high or not
  {
    found_left = 1;
  }
  else if(a[0] == HIGH && a[1] == HIGH && a[2] == HIGH) // condition check  for left sensor is high
```

```
  {
    found_left = 1;
  }
  else if(a[4] == HIGH && a[3] == HIGH)   // condition check for right sensor is high or not
  {
    found_right == 1;
  }
  if(a[2]== HIGH)    // condition check for middle sensor is high or not
  {
    found_st = 1;
  }

  unsigned char dir;

  dir = select_turn(found_left, found_right, found_st);

  turn(dir);
}
```

.4.

```
//define the arduino pin numbers connected to motor

int L1 = 2;

int L2 = 3;

int r1 = 4;

int r2 = 5;

int en1 = 9;

int en2 = 10;

// array to store five sensor values

int a[6] = {0, 0, 0, 0, 0,0};

int last_proportional = 0;

int integral = 0;

//function prototype of different functions

char select_turn(unsigned char found_left, unsigned char found_right, unsigned char found_st);
```

```
int mod(int v);
int set_motors(int a, int b);
void turn(char dir);
void PID();
int right = 0;
int left = 0;
//setup function to define the pin Mode is input or output
void setup()
{
  pinMode(L1, OUTPUT);
  pinMode(L2, OUTPUT);
  pinMode(r1, OUTPUT);
  pinMode(r2, OUTPUT);
  pinMode(en1, OUTPUT);
  pinMode(en2, OUTPUT);
  Serial.begin(9600);
}
//infinite loop function
void loop()
{
  PID();  // calling the pid function

  unsigned char found_left = 0;
  unsigned char found_right = 0;
```

```
  unsigned char found_st = 0;
  readline();  //calling readline function to read sensor values
  if(a[0] == HIGH)  // condition check for left senor is high or not
  {
    found_left = 1;
  }
  else if(a[0] == HIGH && a[1] == HIGH && a[2] == HIGH) // condition check  for left sensor is high
  {
    found_left = 1;
  }
  else if(a[5] == HIGH && a[4] == HIGH)   // condition check for right sensor is high or not
  {
    found_right == 1;
  }
  if(a[3]== HIGH)    // condition check for middle sensor is high or not
  {
    found_st = 1;
  }
  unsigned char dir;
  dir = select_turn(found_left, found_right, found_st);
  turn(dir);
}
void PID()     // PID function definition
{
// Variable initialization
```

```
 int i;

 int power_difference = 0;

 float Kp, Ki, Kd;

 unsigned int position;

 int derivative,proportional;

 while(1)

 {

   position = readline();            //reading the sensor value and storing in variable position

    Serial.println("position = ");

   Serial.println(position);

    //replace value 2000 with your position by placing your robot at the dead centre and read the value

   proportional = ((int)position - 3000);       derivative = proportional - last_proportional;

   integral = integral+proportional;

   last_proportional = proportional;

  // value of kp ki kd, you have to make changes to make your robot move without oscillation (hit and trial
method)

   Kp = 0.08;

   Ki = 0.0002;

   Kd = 1.0;

  //formula for pid to calculate error (i.e power difference)

   power_difference = proportional * Kp + integral * Ki + derivative *Kd;

   const int max = 180;  //setting the maximum speed of motor

   if(power_difference>max)

    power_difference = max;

   if(power_difference < -max)
```

```
    power_difference = (-1*max);
    if(power_difference < 0)
    {
      //set_motors(max, max-power_difference);
      set_motors(max+power_difference, max);
    }
    else
    {
      //set_motors(max+power_difference, max);
      set_motors(max, max-power_difference);
    }
    readline();
    if(a[0] == LOW && a[1] == LOW && a[2] == LOW && a[3] == LOW && a[4] == LOW && a[5]==LOW)
    return;
    else if(a[0]== HIGH || a[5] == HIGH)
    return;
  }
}
int readline()
{
//Read the sensor values
  a[0] = digitalRead(6);
  a[1] = digitalRead(7);
  a[2] = digitalRead(11);
```

```
  a[3] = digitalRead(12);

  a[4] = digitalRead(13);
  a[5] = digitalRead(14);

  int v;

//calculating the average value of sensors

  v = (5000*a[0] + 4000*a[1] + 3000*a[2] + 2000*a[3] + 1000*a[4] + 0*a[5])/(a[0] + a[1] + a[2] + a[3] + a[4]
+a[5]);

  Serial.println(v);

  return v;

}

char select_turn(unsigned char found_left, unsigned char found_right, unsigned char found_st)

{

  if(found_left == 1)

  return 'L';

  if(found_st == 1)

  return 'S';

  if(found_right == 1)

  return 'R';

  else

  return 'B';

}

void turn(char dir)

{

  switch(dir)

  {

    case 'L':
```

```
    set_motors(0,150);
    delay(350);
    break;
    case 'R':
    set_motors(150,0);
    delay(350);
    break;
     case 'B':
    set_motors(-150,150);
    delay(200);
    break;
    case 'S':
    break;
  }
}
int set_motors(int l, int r)
{
  Serial.println(r);
  Serial.println(l);
  if(l>=0 && r>=0)
  {
    analogWrite(en1,mod(l));
    analogWrite(en2, mod(r));
    digitalWrite(L1, HIGH);
```

```
  digitalWrite(L2, LOW);
  digitalWrite(r1, HIGH);
  digitalWrite(r2, LOW);
 }
else if(l<=0 && r>=0)
 {
  analogWrite(en1,mod(l));
  analogWrite(en2, mod(r));
  digitalWrite(L1, LOW);
  digitalWrite(L2, HIGH);
  digitalWrite(r1, HIGH);
  digitalWrite(r2, LOW);
 }
 else if(l>=0 && r<=0)
 {
  analogWrite(en1,mod(l));
  analogWrite(en2, mod(r));
  digitalWrite(L1, HIGH);
  digitalWrite(L2, LOW);
  digitalWrite(r1, LOW);
  digitalWrite(r2, HIGH);
 }
 else if(l==0 && r==0)
 {
  analogWrite(en1,0);
```

```
    analogWrite(en2, 0);
    digitalWrite(L1, LOW);
    digitalWrite(L2, LOW);
    digitalWrite(r1, LOW);
    digitalWrite(r2, LOW);
}
}
int mod(int v)
{
  if(v<0)
  return -1*v;
  else if(v>0)
  return v;
}
```