```
 #include <ECE3.h>

//pins
 uint16_t sensorValues[8];      //right to left, 0->7
 const int left_nslp_pin=31;    // nslp ==> awake & ready for PWM
 const int right_nslp_pin=11;   // nonsleep for the right
 const int left_dir_pin=29;     //dir
 const int right_dir_pin=30;
 const int right_pwm_pin=39;    // (pulse width modulation)
 const int left_pwm_pin=40;     // left wheel
 const int LED_RF = 41;  //rightfront
 const int LED_RB = 58;  //rightback
 const int LED_LF = 51;  //leftfront
 const int LED_LB = 57;  //leftback

 const int straight = 24;  //bumper 0
 const int ribbon = 25;    // bumper 1




//arrays and variables
 double min[8]={607.4, 706.8, 562.8, 562.2, 607.6, 607.2, 562.4, 629.8};
//min and max values in arrays
 double max[8]={1892.6, 1793.2, 1937.2, 1937.8, 1892.4, 1892.8, 1937.6, 1870.2};
 double Kp = 0.013;
 double Kd = 0.09;
 double prev_weight = 0; //previous weight
 double weight;         //weight for sensor fusion
 double pid;            //PD term
 int leftSpd;
 int rightSpd;
 int initialSpd = 90;

//functions
 void calculatePD();
 void readIR();
 void turnaround();
 void halt();

void setup() {
  ECE3_Init();
// internal pullup resistor 'pulls' input pin 'high' when not pressed
// by default, button is HIGH (LOW is 0, HIGH is 1)
  pinMode(straight, INPUT_PULLUP);
  pinMode(ribbon, INPUT_PULLUP);
 pinMode(left_nslp_pin,OUTPUT);
 pinMode(left_dir_pin,OUTPUT);
 pinMode(left_pwm_pin,OUTPUT);
  digitalWrite(left_dir_pin,LOW);   //LOW is forward, HIGH is backwards
 digitalWrite(left_nslp_pin,HIGH);

 pinMode(right_nslp_pin,OUTPUT);
 pinMode(right_dir_pin,OUTPUT);
 pinMode(right_pwm_pin,OUTPUT);
  digitalWrite(right_dir_pin,LOW);  //HIGH is backwards
 digitalWrite(right_nslp_pin,HIGH);

  pinMode(LED_RF, OUTPUT);
  pinMode(LED_RF, OUTPUT);
  pinMode(LED_RF, OUTPUT);
  pinMode(LED_RF, OUTPUT);

  leftSpd = constrain(leftSpd,0,255);     //speed limit
  rightSpd = constrain(rightSpd,0,255);

 Serial.begin(9600);
  // set the data rate in bits per second for serial data transmission
  delay(2000);



}

void loop() {
  int checkpoint = 0;                 // # of black cross

  //if the bump switch 0 is pressed, do this
  if(digitalRead(straight) == 0){
     digitalWrite(LED_RF, HIGH);
      delay(2000);

      while(checkpoint < 2){


        readIR();
        calculatePD();
        leftSpd = initialSpd - pid;
        rightSpd = initialSpd + pid;
       analogWrite(left_pwm_pin,leftSpd);
       analogWrite(right_pwm_pin,rightSpd);

        if(weight == 0.00){
          checkpoint++;
          halt();
          if(checkpoint == 1){
          turnaround();
          }
        }
      delay(50);
     }
  }


//if bump switch 5 is pressed, do full ribbon
if(digitalRead(ribbon) == 0){
     digitalWrite(LED_RF, LOW);
      delay(2000);

      while(checkpoint != 4){
        readIR();
        calculatePD();
        leftSpd = initialSpd - pid;
        rightSpd = initialSpd + pid;
       analogWrite(left_pwm_pin,leftSpd);
       analogWrite(right_pwm_pin,rightSpd);

        if(weight == 0.00){
         //light indication for when car detects black cross
         digitalWrite(LED_LB, HIGH);
         digitalWrite(LED_RB, HIGH);
          delay(100);
         digitalWrite(LED_LB, LOW);
         digitalWrite(LED_RB, LOW);
          checkpoint++;

          if(checkpoint == 2){
          halt;
          turnaround();
          }
        }
      delay(50);
     }
     halt();
}

  digitalWrite(LED_RF, LOW);
  digitalWrite(straight, 1);
  digitalWrite(ribbon, 1);
  delay(50);
```

```
}

void readIR(){
 ECE3_read_IR(sensorValues);
  double IR[8];
  for (unsigned char i = 0; i < 8; i++)            //checks IR Sensor readings
  {
    //calculate weighting scheme (8-4-2-1)/4
   IR[i]=(sensorValues[i]-min[i])*1000/max[i];
    //Serial.print(IR[i]);
    //Serial.print('\t'); // tab to format the raw data into columns
  }

    weight = (-8*IR[0] - 4*IR[1] - 2*IR[2] - IR[3] + IR[4] + 2*IR[5] + 4*IR[6] +
8*IR[7])/4;
}

void calculatePD(){
  pid = Kp*weight + Kd*(weight-prev_weight);
  prev_weight = weight;
}

void turnaround(){
 digitalWrite(left_nslp_pin,HIGH);
 digitalWrite(right_nslp_pin,HIGH);
  digitalWrite(right_dir_pin,HIGH);   //LOW is forward, HIGH is backwards
 analogWrite(left_pwm_pin,110);
 analogWrite(right_pwm_pin,110);
  delay(490);                         //new battery: 530 > 570 > 600 > 635
 analogWrite(left_pwm_pin,0);
 analogWrite(right_pwm_pin,0);
 digitalWrite(right_dir_pin,LOW);
}

void halt(){
 digitalWrite(left_nslp_pin,LOW);
 digitalWrite(right_nslp_pin,LOW);
}
```