WORKING CODE OF PROJECT

A) Code for Rescheduling Algorithm

```
#include<TimerOne.h>

int signal1[] = {23, 25, 27};
int signal2[] = {29, 31, 33};
int signal3[] = {35, 37, 39};
int signal4[] = {41, 43, 45};

int redDelay = 3000;
int yellowDelay = 1000;

volatile int triggerpin1 = 11;
volatile int echopin1 = 10;
volatile int triggerpin2 = 7;
volatile int echopin2 = 6;
volatile int triggerpin3 = 5;
volatile int echopin3 = 4;
volatile int triggerpin4 = 3;
volatile int echopin4 = 2;

volatile long time;                 // Variable for storing the time traveled
volatile int S1, S2, S3, S4;          // Variables for storing the distance covered

int t = 10;  // distance under which it will look for vehicles.

void setup(){
  Serial.begin(115200);
  Timer1.initialize(100000);  //Begin using the timer. This function must be called
first. "microseconds" is the period of time the timer takes.
  Timer1.attachInterrupt(softInterr); //Run a function each time the timer period
finishes.

  // Declaring LED pins as output
  for(int i=0; i<3; i++){
    pinMode(signal1[i], OUTPUT);
    pinMode(signal2[i], OUTPUT);
    pinMode(signal3[i], OUTPUT);
    pinMode(signal4[i], OUTPUT);
  }

  // Declaring ultrasonic sensor pins as output
  pinMode(triggerpin1, OUTPUT);
```

```
  pinMode(echopin1, INPUT);
  pinMode(triggerpin2, OUTPUT);
  pinMode(echopin2, INPUT);
  pinMode(triggerpin3, OUTPUT);
  pinMode(echopin3, INPUT);
  pinMode(triggerpin4, OUTPUT);
  pinMode(echopin4, INPUT);
}

void loop()
{
  // If there are vehicles at signal 1
  while (S1<t)
  {
    signal1Function();
  }
 /* if  (S1>t)
  {
    //signal01Function();
  }
  // If there are vehicles at signal 2
  */while (S2<t)
  {
    signal2Function();
  }
  /*if  (S2>t)
  {
    //signal02Function();
  }
  */
  // If there are vehicles at signal 3
  while (S3<t)
  {
    signal3Function();
  }
   /*if  (S3>t)
  {
    signal03Function();
  }
  */
  // If there are vehicles at signal 4
  while (S4<t)
  {
    signal4Function();
  }
  // If there are NO BUSY vehicles at signalS
```

```
/* if  (S4>t)
  {
    signal04Function();
  }
  */
}

// This is interrupt function and it will run each time the timer period finishes. The timer period is set at 100 milli seconds.
void softInterr()
{
  // Reading from first ultrasonic sensor
  digitalWrite(triggerpin1, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerpin1, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerpin1, LOW);
  time = pulseIn(echopin1, HIGH);
  S1= time*0.034/2;

  // Reading from second ultrasonic sensor
  digitalWrite(triggerpin2, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerpin2, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerpin2, LOW);
  time = pulseIn(echopin2, HIGH);
  S2= time*0.034/2;

  // Reading from third ultrasonic sensor
  digitalWrite(triggerpin3, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerpin3, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerpin3, LOW);
  time = pulseIn(echopin3, HIGH);
  S3= time*0.034/2;

  // Reading from fourth ultrasonic sensor
  digitalWrite(triggerpin4, LOW);
  delayMicroseconds(2);
  digitalWrite(triggerpin4, HIGH);
  delayMicroseconds(10);
  digitalWrite(triggerpin4, LOW);
  time = pulseIn(echopin4, HIGH);
  S4= time*0.034/2;
```

```
  // Print distance values on serial monitor for debugging
  Serial.print("S1: ");
  Serial.print(S1);
  Serial.print("  S2: ");
  Serial.print(S2);
  Serial.print("  S3: ");
  Serial.print(S3);
  Serial.print("  S4: ");
  Serial.println(S4);
}

void signal1Function()
{
  Serial.println("1");
  low();
  // Make RED LED LOW and make Green HIGH for 5 seconds
  digitalWrite(signal1[0], LOW);
  digitalWrite(signal1[2], HIGH);
  delay(redDelay);

  // if there are vehicels at other signals
  if(S2<t || S3<t || S4<t)
  {
    // Make Green LED LOW and make yellow LED HIGH for 2 seconds
    digitalWrite(signal1[2], LOW);
    digitalWrite(signal1[1], HIGH);
    delay(yellowDelay);
  }
}

void signal2Function()
{
  Serial.println("2");
  low();
  digitalWrite(signal2[0], LOW);
  digitalWrite(signal2[2], HIGH);
  delay(redDelay);

  if(S1<t || S3<t || S4<t)
  {
    digitalWrite(signal2[2], LOW);
    digitalWrite(signal2[1], HIGH);
    delay(yellowDelay);
  }
}
```

```
void signal3Function()
{
 Serial.println("3");
 low();
 digitalWrite(signal3[0], LOW);
 digitalWrite(signal3[2], HIGH);
 delay(redDelay);

 if(S1<t || S2<t || S4<t)
 {
  digitalWrite(signal3[2], LOW);
  digitalWrite(signal3[1], HIGH);
  delay(yellowDelay);
 }
}

void signal4Function()
{
 Serial.println("4");
 low();
 digitalWrite(signal4[0], LOW);
 digitalWrite(signal4[2], HIGH);
 delay(redDelay);

 if(S1<t || S2<t || S3<t)
 {
  digitalWrite(signal4[2], LOW);
  digitalWrite(signal4[1], HIGH);
  delay(yellowDelay);
 }

}

/*void signal01Function()
{
 Serial.println("01");
 low();
 digitalWrite(signal1[0], LOW);
 digitalWrite(signal1[2], HIGH);
 delay(3000);
 digitalWrite(signal1[2], LOW);
 digitalWrite(signal1[1], HIGH);
 delay(1000);
 digitalWrite(signal1[1], LOW);
 digitalWrite(signal1[0], HIGH);
```

```
 }

void signal02Function()
{
  Serial.println("02");
  low();
  digitalWrite(signal2[0], LOW);
  digitalWrite(signal2[2], HIGH);
  delay(3000);
  digitalWrite(signal2[2], LOW);
  digitalWrite(signal2[1], HIGH);
  delay(1000);
  digitalWrite(signal2[1], LOW);
  digitalWrite(signal2[0], HIGH);
}

void signal03Function()
{
  Serial.println("03");
  low();
  digitalWrite(signal3[0], LOW);
  digitalWrite(signal3[2], HIGH);
  delay(3000);
  digitalWrite(signal3[2], LOW);
  digitalWrite(signal3[1], HIGH);
  delay(1000);
  digitalWrite(signal3[1], LOW);
  digitalWrite(signal3[0], HIGH);
}

void signal04Function()
{
  Serial.println("04");
  low();
  digitalWrite(signal4[0], LOW);
  digitalWrite(signal4[2], HIGH);
  delay(3000);
  digitalWrite(signal4[2], LOW);
  digitalWrite(signal4[1], HIGH);
  delay(1000);
  digitalWrite(signal4[1], LOW);
  digitalWrite(signal4[0], HIGH);

 }
*/
// Function to make all LED's LOW except RED one's.
```

```
void low()
{
 for(int i=1; i<3; i++)
 {
  digitalWrite(signal1[i], LOW);
  digitalWrite(signal2[i], LOW);
  digitalWrite(signal3[i], LOW);
  digitalWrite(signal4[i], LOW);
 }
 for(int i=0; i<1; i++)
 {

  digitalWrite(signal2[i], HIGH);
  digitalWrite(signal3[i], HIGH);
  digitalWrite(signal4[i], HIGH);
 }
}
```

B) Code For Energy Efficient Street Light

```
int IR1 = 8;
int IR2 = 12;
int LDR = 7;
int led1 = 3;
int led2 = 5;
int val1;
int val2;
int val4;

void setup()
{
pinMode(IR1,INPUT);
pinMode(IR2,INPUT);
pinMode(LDR,INPUT);
pinMode(led1,OUTPUT);
pinMode(led2,OUTPUT);


}

void loop() {
val1 = digitalRead(IR1);
val2 = digitalRead(IR2);
val4 = digitalRead(LDR);

if(val1==1&&val4==0&&val2==1)
```

```
{
digitalWrite(3,LOW);
digitalWrite(5,LOW);


}
else if(val1==1&&val4==1&&val2==1)
{
analogWrite(3,20);
analogWrite(5,20);

}

else if(val1==0&&val4==1&&val2==1)
{
analogWrite(3,500);
analogWrite(5,20);

}
else if(val1==1&&val4==1&&val2==0)
{
analogWrite(3,20);
analogWrite(5,500);

}
else if(val1==1&&val4==1&&val2==1)
{
analogWrite(3,20);
analogWrite(5,20);

}
}
```

C) Code For Data send to cloud for analysis

```
import requests
import json
import random
import time
import matplotlib.pyplot as plt

# Replace with your ThingsBoard parameters
accessToken = "7a9ZB5AHuouEu4mYy32J"
thingsboard_url = "http://demo.thingsboard.io/api/v1/" + accessToken + "/telemetry"

# Initialize lists to store sensor data
```

```
S1_data = []
S2_data = []
S3_data = []
S4_data = []
time_data = []

def send_data_to_thingsboard(S1, S2, S3, S4):
    payload = {
        "S1": S1,
        "S2": S2,
        "S3": S3,
        "S4": S4
    }

    headers = {
        "Content-Type": "application/json"
    }

    try:
        response = requests.post(thingsboard_url, headers=headers,
data=json.dumps(payload))
        response.raise_for_status()
        print("Data sent successfully:", payload)
    except requests.exceptions.RequestException as e:
        print("Error sending data:", e)

def plot_data(averages):
    lanes = ['Lane 1', 'Lane 2', 'Lane 3', 'Lane 4']
    plt.bar(lanes, averages)
    plt.xlabel('Lane')
    plt.ylabel('Average Sensor Value')
    plt.title('Average Sensor Values After 30 Seconds')
    plt.show()

def calculate_averages(data):
    return sum(data) / len(data) if len(data) > 0 else 0

if _name_ == "_main_":
    start_time = time.time()
    while True:
        # Simulate sensor readings
        S1 = random.randint(1, 10)
        S2 = random.randint(1, 10)
        S3 = random.randint(1, 10)
        S4 = random.randint(1, 10)
```

```
# Append data to lists
S1_data.append(S1)
S2_data.append(S2)
S3_data.append(S3)
S4_data.append(S4)
time_data.append(time.time() - start_time)

# Send simulated data to ThingsBoard
send_data_to_thingsboard(S1, S2, S3, S4)

# Stop after 30 seconds
if time.time() - start_time >= 30:
    avg_S1 = calculate_averages(S1_data)
    avg_S2 = calculate_averages(S2_data)
    avg_S3 = calculate_averages(S3_data)
    avg_S4 = calculate_averages(S4_data)

    print("Average Sensor Values:")
    print("Lane 1:", avg_S1)
    print("Lane 2:", avg_S2)
    print("Lane 3:", avg_S3)
    print("Lane 4:", avg_S4)

    plot_data([avg_S1, avg_S2, avg_S3, avg_S4])
    break

# Adjust the delay based on your requirements
time.sleep(5)
```