Smart Traffic Signal Optimization

Scenario: You are part of a team working on an initiative to optimize traffic signal management in a busy city to reduce congestion and improve traffic flow efficiency using smart technologies.

Tasks:

1. **Data Collection and Modelling:**

Define a data structure to collect real-time traffic data from sensors

Code:

```
public class TrafficData {

        private int vehicleCount;

        private double averageSpeed;

        private int intersectionId;

        private long timestamp;


        public TrafficData(int vehicleCount, double averageSpeed, int
intersectionId,    long timestamp) {

       this.vehicleCount = vehicleCount;

       this.averageSpeed = averageSpeed;

       this.intersectionId = intersectionId;

       this.timestamp = timestamp;

         }

}
```

2. Algorithm Design:

Develop algorithms to analyze the collected data and optimize traffic signal timings.

CODE:

```
public class TrafficSignalOptimizer {

    public static int calculateGreenTime(TrafficData data) {

    int baseGreenTime = 30; // Base green time in seconds

    int maxGreenTime = 120; // Maximum green time in seconds

    int greenTime = baseGreenTime + data.getVehicleCount() / 10;

    return Math.min(greenTime, maxGreenTime);

    }

    public static void optimizeSignal(TrafficData[] trafficDataArray) {

    for (TrafficData data : trafficDataArray) {

    int greenTime = calculateGreenTime(data);

        System.out.println("Intersection " + data.getIntersectionId() + ": Set green time to " + greenTime + " seconds");

    }

    }

}
```

3.Implementation:

Implement a Java application that integrates with traffic sensors and controls traffic signals.

CODE:

```
 import java.util.Timer;

import java.util.TimerTask;

public class TrafficSignalController {

    private TrafficSensor trafficSensor;

    private TrafficSignal trafficSignal;


    public TrafficSignalController(TrafficSensor trafficSensor, TrafficSignal
trafficSignal) {

    this.trafficSensor = trafficSensor;

    this.trafficSignal = trafficSignal;

    }

    public void start() {

    Timer timer = new Timer();

    timer.schedule(new TimerTask() {

            @Override

            public void run() {

            TrafficData data = trafficSensor.collectData();

            int greenTime = TrafficSignalOptimizer.calculateGreenTime(data);

             trafficSignal.setGreenTime(greenTime);

            }

    }, 0, 10000); // Adjust traffic signals every 10 seconds

    }
```

}

4. Visualization and Reporting:

Develop visualizations and generate reports.

CODE:

```
package com.example.trafficsignalsnew;


import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.scene.layout.VBox;
import javafx.stage.Stage;
import javafx.util.Duration;


import java.io.IOException;



public class HelloApplication extends Application {
    @Override
    public void start(Stage primaryStage) {
```

```
Circle redLight = new Circle(50, Color.RED);
Circle yellowLight = new Circle(50, Color.GRAY);
Circle greenLight = new Circle(50, Color.GRAY);
VBox root = new VBox(10);
root.getChildren().addAll(redLight, yellowLight, greenLight);
Scene scene = new Scene(root, 200, 600);
primaryStage.setTitle("Traffic Signal Animation");
primaryStage.setScene(scene);
primaryStage.show();

Timeline timeline = new Timeline(
        new KeyFrame(Duration.seconds(0), e -> {
        redLight.setFill(Color.RED);
        yellowLight.setFill(Color.GRAY);
        greenLight.setFill(Color.GRAY);
        }),
        new KeyFrame(Duration.seconds(3), e -> {
        redLight.setFill(Color.GRAY);
        yellowLight.setFill(Color.YELLOW);
        greenLight.setFill(Color.GRAY);
        }),
        new KeyFrame(Duration.seconds(6), e -> {
        redLight.setFill(Color.GRAY);
```

```
            yellowLight.setFill(Color.GRAY);
            greenLight.setFill(Color.GREEN);
            }),
            new KeyFrame(Duration.seconds(9), e -> {
            redLight.setFill(Color.RED);
            yellowLight.setFill(Color.GRAY);
            greenLight.setFill(Color.GRAY);
            })
        );
      timeline.setCycleCount(Timeline.INDEFINITE);
        timeline.play();
        }


        public static void main(String[] args) {
        launch();
        }
}
```

5.User Interaction: