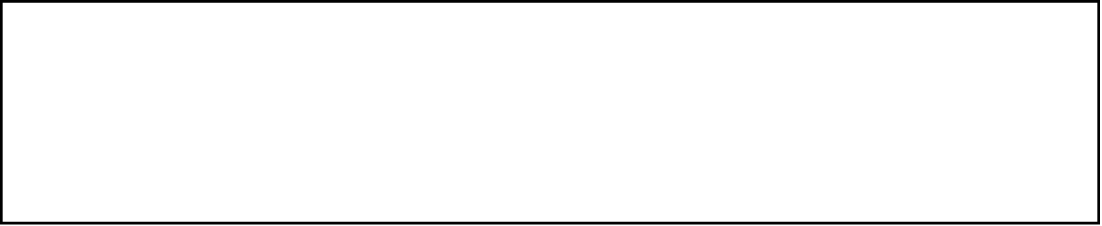
A picture containing text, clipart

Description automatically generatedText, logo

Description automatically generated

# CSA0989 PROGRAMMING IN JAVA

## JAVA ASSIGNMENT

**Smart Traffic Signal Optimization:**

Smart Traffic Signal Optimization aims to improve traffic flow and reduce congestion using real-time data and adaptive signal control. The system uses sensors and cameras to gather traffic data, processes it with algorithms, and adjusts traffic signals dynamically. Here are key components and benefits of such a system:

**Pseudo Code Implementation**

Initialize sensors and cameras

Initialize traffic signal controllers

Initialize central processing unit (CPU)

Function collectTrafficData():

For each sensor and camera:

Read traffic data

Send data to CPU

Function processTrafficData(data):

Analyze traffic patterns

Predict traffic flow

Calculate optimal signal timings

Function updateTrafficSignals(timings):

For each traffic signal controller:

Set signal timings based on calculated timings

Function main():

While system is running:

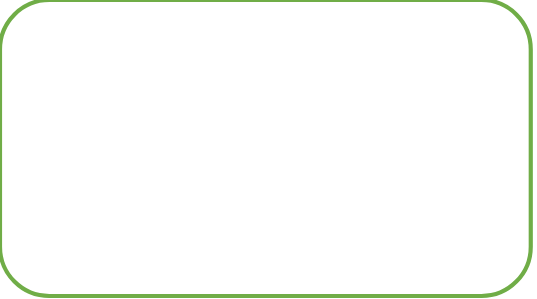
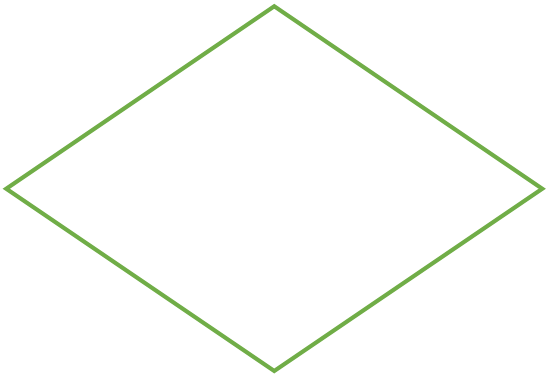
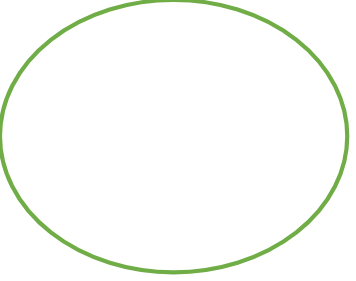
trafficData = collectTrafficData()

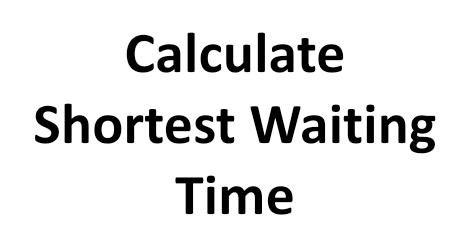
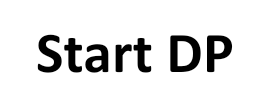
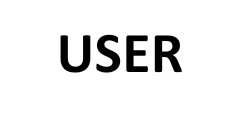
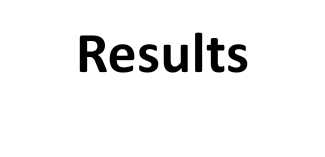
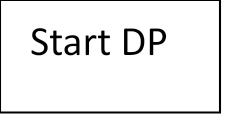
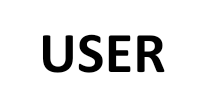
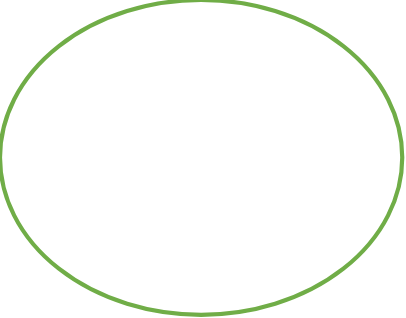
signalTimings = processTrafficData(trafficData)

updateTrafficSignals(signalTimings)

Wait for a predefined interval

**Data Flow Diagram**







**Java Code**

**import java.util.\*;**

**class TS {**

**String name;**

**int greenTime;**

**int redTime;**

**TS(String x) {**

**name = x;**

**greenTime = 30;**

**redTime = 30;**

**}**

**void timing(int Density) {**

**if (Density > 100) {**

**greenTime = 45;**

**redTime = 15;**

**} else if (Density > 50) {**

**greenTime = 35;**

**redTime = 25;**

**} else {**

**greenTime = 30;**

**redTime = 30;**

**}**

**}**

**void dispTiming() {**

**System.out.println("Traffic Signal"+name);**

**System.out.println("Green Time:"+greenTime+"seconds");**

**System.out.println("Red Time:"+redTime+"seconds");**

**}**

**}**

**class TD{**

**Random random;**

**TD() {**

**random = new Random();**

**}**

**int getTD() {**

**return random.nextInt(150);**

**}**

**}**

**public class SmartTS{**

**public static void main(String[] args) {**

**TS sig1 = new TS("Signal 1");**

**TS sig2 = new TS("Signal 2");**

**TD sensor = new TD();**

**for (int i=0;i<5;i++) {**

**int den1 = sensor.getTD();**

**int den2 = sensor.getTD();**

**System.out.println("Cycle"+(i + 1));**

**System.out.println("Traffic Density at Signal 1:"+den1);**

**System.out.println("Traffic Density at Signal 2:"+den2);**

**sig1.timing(den1);**

**sig2.timing(den2);**

**sig1.dispTiming();**

**sig2.dispTiming();**

**System.out.println();**

**}**

**}**

**}**

**Documentation**

Smart Traffic Signal Optimization is a technically advanced conceptual project that is designed to enhance the efficacy of traffic signal systems. It strives to ease traffic flow more efficiently, reduce congestion rate and minimize waiting time caused due to congestion caused. Traffic signals monitor the roads using sensor-incorporated cameras and adjust their timers based on prevailing traffic conditions. The sensor-incorporated cameras collect data such as vehicle count, speed and queue length. This data helps in comprehending congestion rate and take required actions.

Data Fusion i.e., integrating data collected from various sources like traffic cameras GPS from vehicles assist in better traffic control management.

Dynamic Programming is an algorithm used to solve complex problems by dividing them into smaller subproblems (divide-and-conquer methodology). By dynamically storing the results of subproblems, chances of redundancy are avoided. In this project, DP is used to calculate the optimal signal timings by considering various state of traffic congestion, breaking them down based on previous reference data and finding optimal solutions to minimize congestion. It optimizes signal timings help to reduce traffic jams and improve overall traffic flow. Efficient signal control minimizes waiting times at intersections, leading to faster congestion rates. Better signal management can reduce accidents and improve pedestrian safety.

Dynamic Programming provides a robust framework for optimizing traffic signal timings, improving traffic flow, reducing congestion, and enhancing safety. Addressing challenges such as data accuracy and real-time processing is crucial for effective implementation.

**Conclusion**

Dynamic programming provides a good way to optimize smart traffic signals by systematically calculating lighting duration according to time and vehicle queue lengths. This method improves traffic flow, reduces congestion, and increases safety by dynamically adjusting signal levels and improving intersection synchronization. Although useful, issues such as data accuracy, energy management, and timely processing of data need to be addressed to achieve its potential. Continuous progress, including the integration of machine learning and advanced data fusion, will strengthen the efficiency and flexibility of signal processing optimization.

Techniques such as dynamic programming and machine learning can improve signal level based on traffic patterns, while vehicle-to-infrastructure (V2I) communication and network coordination can improve accurate synchronization of multiple signals. This approach can reduce traffic congestion, reduce travel time, and increase safety, but it also faces challenges such as data accuracy, scalability, and speed.