FACULTY OF ENGINEERING SCIENCES AND TECHNOLOGY

Department: **Computer Science**
Program: **BS(CS)**

Data Structures and Algorithm

Announced date: 12-05-2025 **Due Date: 15-06-2025**
Total Marks = 10

Complex Computing Problem (CCP)		
Mapped CLO	**SDG**	**Complex Problem Solving Mapped**
CLO3	4 & 9	WP1 (Depth of knowledge required) WP2 (Range of conflicting requirements required) WP3 (Depth of analysis required) WP4 (Familiarity of Issues)

Traffic Signal Simulation Using JavaFX

By Syed Ibtihaj Ali

Course: Data Structures and Algorithms

Department: Computer Science

Program: BS(CS)

Problem Statement

Urban congestion is a pressing issue in modern cities, often caused by poorly managed traffic signals. This project aims to simulate a simple grid-based traffic light and vehicle movement system using JavaFX to better understand traffic signal behavior and emergency vehicle handling in a simulated 5x5 grid of intersections.

Objectives

- Develop a graphical simulation of urban traffic signal behavior.
- Randomly assign traffic light states and emergency vehicles.
- Use color-coded rectangles and circles to represent vehicles and signals.
- Simulate simple vehicle movement across intersections.
- Update vehicle and signal states in real-time using JavaFX's Timeline.

Tools & Technologies Used

- Programming Language: Java
- GUI Framework: JavaFX
- IDE: Eclipse
- Animation API: Timeline and KeyFrame
- Deployment: https://github.com/Ibtihaj51/Traffic-Simulation

System Design

Core Components:
- Rectangle (Vehicle): Represents a moving vehicle (gray = normal, red = emergency).
- Circle (Signal): Traffic light signal at an intersection (green/red).
- Grid Layout: 5x5 matrix simulating urban intersections.
- Timeline Animation: Controls periodic updates to simulate real-time movement and light switching.

Data Structures Used

- 2D Array of Rectangles (Rectangle[][]): Represents vehicles in the grid.
- 2D Array of Circles (Circle[][]): Represents traffic signals.
- 2D Boolean Array (boolean[][]): Stores signal status (green/red).

- Random Generator: Used to initialize light states and emergency vehicle status.

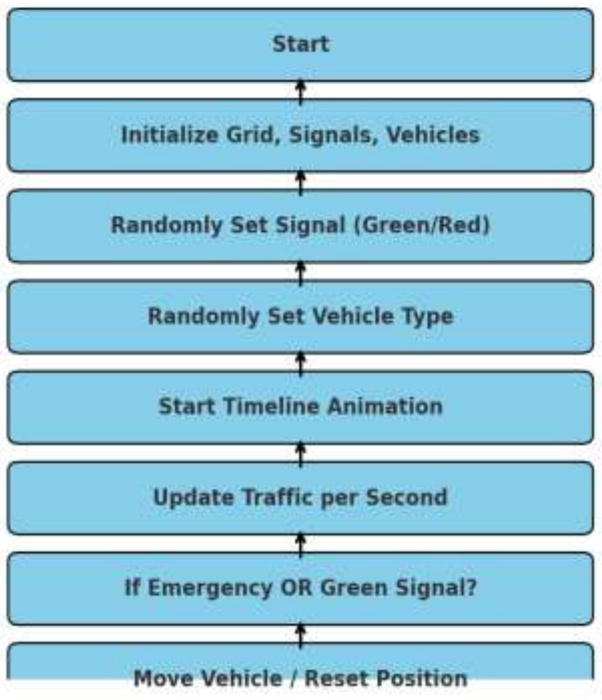
Implementation Summary

Main.java
- `start()` Method: Sets up the JavaFX scene, grid, and initializes signals and vehicles.
- `updateTraffic()` Method: Updates vehicle position and signal state every second.
  - Emergency vehicles (Color.RED) can move regardless of signal state.
  - Regular vehicles move only on green signals.
- Timeline: Animates the entire simulation with 1-second intervals.

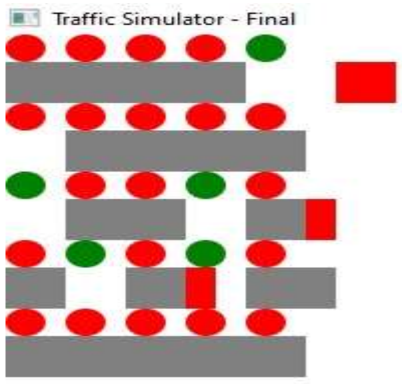
Testing & Results

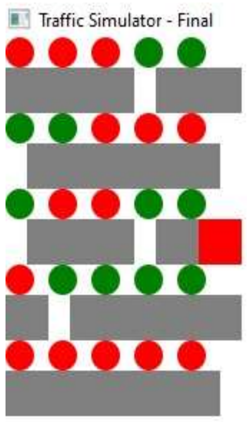
Feature	Status
Grid Initialization	✅ 5x5 grid rendered correctly
Signal Randomization	✅ Signals are randomly red/green
Vehicle Assignment	✅ Emergency vehicles (20% chance)
Movement Logic	✅ Emergency/green signal movement handled
GUI Animation	✅ Working as expected via Timeline
Reset on Edge	✅ Vehicles reset position after reaching boundary

Flow Chart:


Start
Initialize Grid, Signals, Vehicles
Randomly Set Signal (Green/Red)
Randomly Set Vehicle Type
Start Timeline Animation
Update Traffic per Second
If Emergency OR Green Signal?
Move Vehicle / Reset Position

Output:





Code Snippet

```
if (emergency || isGreen[i][j]) {
    vehicles[i][j].setTranslateX(vehicles[i][j].getTranslateX() + 5);
    if (vehicles[i][j].getTranslateX() > 50) {
        vehicles[i][j].setTranslateX(0);
        vehicles[i][j].setFill(rand.nextDouble() < 0.2 ? Color.RED :
Color.GRAY);
        isGreen[i][j] = rand.nextBoolean();
        signals[i][j].setFill(isGreen[i][j] ? Color.GREEN : Color.RED);
    }
}
```

Key Learnings

- Understanding of 2D data structures (arrays) in real-time simulations.
- Hands-on experience with JavaFX GUI components and animation.

- Practical insight into event-driven programming using Timeline.
- Simulating priority logic for emergency handling.

Conclusion

This simplified simulation demonstrates how basic data structures can be used to emulate traffic signal behavior and vehicle movement in a grid. Though minimal compared to a full-scale simulation, it successfully introduces core ideas like emergency priority, traffic flow control, and real-time visualization using JavaFX.