Test Plan

Introduction:

The objective and extent of the test is to cover the functional and non functional system requirements.

System Overview:

The operators should be able to determine the start and the end points of each vehicle. They should be able to set intersections by clicking on a button. They may run the stimulation and start a timer. When all cars reach their destination, the timer will stop and the time taken will be recorded.

Features to be Tested:

Testing Requirements	System Requirement	Short Description
1	F3	The system is reset correctly
2	F1	Set the types of intersections
3	F2	Start and end points of vehicles can be set
4	F4	All cars obey the rules of the road, and traffic lights behave as they should
5	N1	All vehicles moving at once
6	N2	Cars take same route if destination is unchanged
7	N4	Tracks time throughout the session
8	N5	Intersections cannot be changed mid simulation
9	N3	User inputs processed in 1 second

Test Environment:

The most current version of eclipse with a loaded copy of the program and provided source files. Installing the WindowBuilder plugin for eclipse will help with the gui design and testing, but is not required for the testing.

Test Cases:

Test case 1

Component under test

City Traffic Simulator

Features to be tested

2,3,4,6,7,8,9

Initial Conditions

The system is currently running and awaiting input from the user relating to the types of intersections, as well as the start and end points of cars.

Expected Behavior

1. The user selects the mode of the intersections
    a. The intersections, from left to right, top to bottom, are: Traffic Light, E-W Stop Signs, N-S Stop Signs, All-Way Stop Signs, Traffic Light, Traffic Light, N Stop Sign, S Stop Sign, Traffic Light, W Stop Sign, Traffic Light, Traffic Light, All-Way Stop Signs, All-Way Stop Signs, E Stop Sign, S Stop Sign, Traffic Light, All-Way Stop Signs, S Stop Sign, W Stop Sign, Traffic Light, All-Way Stop Sign, N Stop Sign, N Stop Sign
2. The user adds cars to the traffics by pressing the Add Cars Button
3. The user sets the start and end points of the cars by uploading a file
    a. The start and end points of each car are grouped together, separated by a comma, and each new car is placed on a new line of the file. The points are as follows, with each car's points grouped inside parentheses: (1, 1, 1, 29), (1, 29, 1, 1), (17,29,1,1), (17,29,1,3), (5,29,11,8), (1,2, 9, 15), (17,29, 1,3), (1, 29, 18,29), (1, 22, 4,8), (1,7, 15, 24),
4. The user runs the simulator by pressing the start button

    a. After pressing the start button, the drop-down box to change the type of an intersection disappears, so the type of an intersection can't be changed mid-simulation
5. The system displays bars representing the flow of traffic in each intersection, and displays the cars moving around the map
6. The system starts the timer and records time throughout the session, and the timer stops when all cars have reached their destination
7. All cars should move to their destinations
8. The simulation will stop when all cars have reached their endpoints
9. Each time the user presses a button, measure the time it takes the system to process the input.

Test case 2

Component under test

City Traffic Simulator

Features to be tested

2,3,4,5,6,7,8

Initial Condition

The system is running and awaiting input from the user

Expected Behavior

1. The user selects the type of each intersection
    a. The selections of intersections are arbitrary for this test case, as they were all tested in the previous test case
2. The user adds cars by pressing the Add Cars button
3. The user sets the start and end points of the cars by uploading a file
    a. The start and end points of each car are grouped together, separated by a comma, and each new car is placed on a new line of the file. The points are as follows, with each car's points grouped inside parentheses: (1, 1, 1, 29), (1, 29, 1, 1), (17,29,1,1), (17,29,1,3), (5,29,11,8), (1,2, 9, 15), (17,29, 1,3), (1, 29, 18,29), (1, 22, 4,8), (1,7, 15, 24), (1, 1, 1, 29)
4. The user runs the simulation by pressing the start simulation button
5. The system displays the intersection types and the cars moving throughout the map
6. The system starts the timer and records time throughout the session
7. All cars should move at once

8. Check that car 1 and car 11 follow the same path to their destination
9. Upon all cars reaching their destination, the timer stops, signaling the simulation has finished

Test Case 3

Component under test

City Traffic Simulator

Features to be tested

1,2,3,4,5,6,7,8

Initial condition

The simulation is running and awaiting input from the user

Expected Behavior

1. The user selects the type of each intersection
    a. The selections of intersections are arbitrary for this test case, as they were all tested in the previous test case
2. The user adds cars by pressing the Add Cars button
3. The user sets the start and end points of the cars by uploading a file
4. The start and end points of each car are grouped together, separated by a comma, and each new car is placed on a new line of the file. The points are as follows, with each car's points grouped inside parentheses: (1, 29, 1, 1), (17,29,1,1), (5,29,11,8), (1,7, 15, 24)
5. The user runs the simulation by pressing the Start button
6. The system displays the intersection modes and the cars moving throughout the map
7. The system starts the timer and records time throughout the session
8. All cars should move at once
9. The user is unable to change the type of each intersection while the simulation is running
10. After an arbitrary amount of time, the user presses the Reset button to reset the simulation
11. All cars are returned to their start points
12. The user is able to run the simulation again, and it runs correctly until completion

Test case 4

Component under test

City Traffic Simulator

Features to be tested

2,3,4,6,7,8

Initial Conditions

The system is currently running and awaiting input from the user relating to the types of intersections, as well as the start and end points of cars.

Expected Behavior

1. The user selects the mode of the intersections
    a. The intersections, from left to right, top to bottom, are: Traffic Light, E-W Stop Signs, N-S Stop Signs, Sensor-Based Traffic Light, Traffic Light, Traffic Light, N Stop Sign, S Stop Sign, Traffic Light, W Stop Sign, Sensor-Based Traffic Light, Sensor-Based Traffic Light, All-Way Stop Signs, All-Way Stop Signs, E Stop Sign, S Stop Sign, Traffic Light, All-Way Stop Signs, Sensor-Based Traffic Light, W Stop Sign, Traffic Light, All-Way Stop Sign, Sensor-Based Traffic Light, N Stop Sign
2. The user adds cars to the traffics by pressing the Add Cars Button
3. The user sets the start and end points of the cars by uploading a file
    a. The start and end points of each car are grouped together, separated by a comma, and each new car is placed on a new line of the file. The points are as follows, with each car's points grouped inside parentheses: (1, 1, 1, 29), (1, 29, 1, 1), (17,29,1,1), (17,29,1,3), (5,29,11,8), (1,2, 9, 15), (17,29, 1,3), (1, 29, 18,29), (1, 22, 4,8), (1,7, 15, 24), (26,20,26,1)
4. The user runs the simulator by pressing the start button
    a. After pressing the start button, the drop-down box to change the type of an intersection disappears, so the type of an intersection can't be changed mid-simulation
5. The system displays bars representing the flow of traffic in each intersection, and displays the cars moving around the map
6. The system starts the timer and records time throughout the session, and the timer stops when all cars have reached their destination
7. All cars should move to their destinations
8. The simulation will stop when all cars have reached their endpoints.