**Project 3 – Traffic Light Simulator GUI Developer’s Guide**

Thomas, Lindely

Computer Science Department

CMSC 335: Object-Oriented and Concurrent Programming

Professor Morad, Osama

12 OCT 2021

**Compile and Run Shapes GUI Program**

**Note:** Program was developed using Java 17

**COMPILING:** unzip Lindley\_Final\_CMSC\_335,zip, open your preferred terminal and change directory (cd) to the final\_src folder. Once in folder, enter the command: javac main\_FINAL.java.

**Running:** Once program is compiled, remain in the same folder and enter the command java main\_FINAL.java, then GUI will open to allow interaction with program.

**UML:** UML Diagram can be viewed from image included named “UML” for larger picture.

**Program Functionality and Design**

I have designed the GUI to allow the user to select how many cars and how many lights they would like to be in the simulation. When the simulation starts, threads for the lights and cars start at random times. This ensures that all cars and all lights act more like an actual traffic pattern, where cars are not all coming into the simulation all at once, and all being at the same position. Also, this makes the traffic lights act more natural, because they will not all be the same color at the same time. The start button starts the simulation after the user has selected number of cars and lights, then presses enter. Pause and resume allow the user to control the current simulation. Stop / clear, clears the simulation and removes all lights and cars.

Limitations

The program is limited to 5 cars and 5 lights. I also think my code is poorly optimized, because it can slow down quite a bit when using 5 cars and 5 lights, but only for a few moments.

Test Plan

* Test simulations with varying cars and lights work, and test stop, start, pause, resume.
* Ensure cars are stopping at red lights and starting when lights turn green.

Test 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Run# | Input | Expected Behavior | Actual Behavior | Pass / Fail |
| 1 | 1 car, 1 light, enter, start, pause resume, stop | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Pass |
| 2 | 2 cars, 2 lights, enter, start, pause resume, stop | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Pass |
| 3 | 3 cars, 3 lights, enter, start, pause resume, stop | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Pass |
| 4 | 4 cars, 4 lights, enter, start, pause resume, stop | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Pass |
| 5 | 5 cars, 5 lights, enter, start, pause resume, stop | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Cars stop at stop lights and start at green lights, pause stops the time and the cars/lights from changing, resume starts everything back up, stop clears the simulation and allows user to do another | Pass |

Run 1:

After starting

Graphical user interface, text, application

Description automatically generated

Pause

Graphical user interface, application

Description automatically generated

Resume

Graphical user interface, application

Description automatically generated

After clicking stop

Graphical user interface, text, application

Description automatically generated

Run 2:

start

Graphical user interface, application

Description automatically generated

pause

Graphical user interface, application

Description automatically generated

Resume

Graphical user interface, application

Description automatically generated

Stop

Graphical user interface, text, application

Description automatically generated

Run 3:

Start

Graphical user interface, application

Description automatically generated

Caught some cars at the first light

Graphical user interface, application

Description automatically generated

Pause

Graphical user interface, application

Description automatically generated

Resume

Graphical user interface, application

Description automatically generated

Stop

Graphical user interface, application

Description automatically generated

Run 4:

Start

Graphical user interface, application

Description automatically generated

Pause

Graphical user interface, application

Description automatically generated

Resume

A screenshot of a computer

Description automatically generated with medium confidence

Stop

Graphical user interface, text, application, Word

Description automatically generated

Run 5:

Start

Graphical user interface, application

Description automatically generated

Pause

Graphical user interface

Description automatically generated with medium confidence

Resume

Graphical user interface

Description automatically generated

Stop

Graphical user interface, text, application, Word

Description automatically generated

Test 2

In this test I will be monitoring 5 cars to see if they stop at lights when they are supposed to. Due to the staggering of time when cars start it is hit or miss if a car will ever have to stop at any light, just like in real life, sometimes all lights are green when travelling down the road. I will try to capture screen shots of the cars stopped

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Run# | Input | Expected behavior | Actual behavior | Pass / Fail |
| 1 | 5 cars, 5 lights | Cars stop at lights | Cars stop at lights | Pass |
| 2 | 5 cars, 5 lights | Cars stop at lights | Cars stop at lights | Pass |
| 3 | 5 cars, 5 lights | Cars stop at lights | Cars stop at lights | Pass |

Run 1

All cars, except 5, got stopped at the light.

Graphical user interface, application

Description automatically generatedCar 5 stopped at light 2

Graphical user interface, application

Description automatically generated

None of the cars will get stopped at the last light at this point, since it will be green by the time they get there

Graphical user interface

Description automatically generated

Run 2

Realized here I forgot to add the y position to the text field, so I just added that to the set text for the fields. Does not change anything besides what prints.

Car 3 stopped at light 1

A screenshot of a computer

Description automatically generated with medium confidence

Cars 1, 2, and 4 stopped at light 2

Graphical user interface, application

Description automatically generated

Car 5 stopped at light 3

Graphical user interface, application

Description automatically generated

All cars out of range of the lights

A screenshot of a computer

Description automatically generated with medium confidence

Run 3

Car 1 and 2 stopped

A screenshot of a computer

Description automatically generated with medium confidence

Car 4 stopped

Graphical user interface, application

Description automatically generated

Car 5 stopped

A screenshot of a computer

Description automatically generated with medium confidence

Car 3 stopped

Graphical user interface, application

Description automatically generated

UML

Diagram

Description automatically generated

Lessons Learned

The main take away I got from this project is that multithreading can make programming much more difficult. Preventing errors like deadlock is very important and seems difficult to detect from standard terminal output. I believe my program was deadlocked a time or two, but the issue is that nothing happens besides the program locking up. This is much different from what I am used to where exceptions occur, and that information is outputted to terminal. Also, with multithreading unexpected behavior seems to be able to happen just from the way the order of the code is setup, originally my code was working, but it did not look like it was working because the traffic lights were changing long after they were supposed to. This was fixed by calling the change color method in the traffic light task, before the thread sleeps.