Some challenges I encountered occurred before I even began programming my project. Initially, I planned on including a lot more components than I was able to include in the end. The first thing I hoped to include was a way to import real-world map data to generate in the traffic simulation. After spending a couple weeks researching this possibility, I concluded that the amount of work involved in making this a reality would be too much for the amount of time I had. While it was possible to download real-world map data, translating it into the simulation would require creating a variety of roads in the simulation itself; this includes multi-lane roads, one-way roads, as well as irregular shaped roads such as curved roads and roads with sharp bends. This would require various methods to display all of these types of roads, create the correct pathfinding for cars to be able to drive along the roads, be able to use multiple lanes of the road, as well as know what lane to be in at various points of the drive. Although this wasn’t implemented, it is something that could be implemented given more time.

Another component that I would’ve hoped to add was a randomly generated map. I initially, thought of doing this when I first programmed the road layout of my simulation, as it would’ve simplified adjusting the map size with a simple change of an input or two. I didn’t immediately pursue this because I wanted to have a working simulation before I started to change the map design aspect of the simulation; however, I ran into more issues that ended up delaying, and eventually, cutting this component out of the final project completely.

Since I didn’t really do any UI programming, aside from a week or two in an undergraduate class, I spent a lot of time reading and looking at other coding examples of displaying objects on a screen. I first designed and programmed a two-way road in the north-south directions. I had several of these scattered across the screen and displaying properly. Then, I created the same road, but in the east-west directions, as well as the junctions that would connect two or more roads. I then found that nothing was displaying on the screen properly. I spent about a week trying to figure out why the objects are no longer being displayed on the screen. After about a week, I decided to begin implementing other aspects of the project that didn’t directly relate to the roads so that I wouldn’t be behind on the entire project. Then, after asking for help, I created a completely new project on the side to try to figure out at what point objects would no longer be displayed on the screen. With this test, I saw that somethings were being displayed while others were not. I soon realized that everything that needed to be displayed had to have been in the same paintComponent() method, or at least that’s my understanding of it. After reconfiguring my original code to accommodate this, all of the roads and traffic lights were properly being displayed and the traffic lights were cycling every 5 seconds.

At that point when everything was being displayed properly, I had already implemented roughly half of the necessary code of the car object. Once I finished that, I began testing the simulation as a whole and I immediately saw some issues: cars were driving off of the roads, cars were getting stuck in intersections as well as on the roads, and cars were jumping around from one intersection to another. Although graphical bugs are easier to spot on the screen, it took some time to figure out the causes behind them. An issue with programming logic was the reason for cars driving off of the road, which was the first fix. The cars jumping around from one intersection to another was fixed next and that was due to the way the car object decides which way it will go at that intersection; as a car approaches an intersection, the car has a list of intersections that it cycles through until it has one that it is approaching. However, after the car decided which way it would turn, the for loop would still cycle through all the intersection, which is why cars would end up being placed at various other intersections, rather than the one it should be at. Currently, the simulation works well with only one car, as I was not able to figure out why cars would stop at various points of the map; however, with one car this doesn’t happen. As soon as another car is placed in the simulation, one of the cars would eventually get stuck somewhere; perhaps it is an issue with collision detection, I can’t say for sure. While it may be an easy fix, I am running out of time to fix the issue, so as of now, there is only one car in the simulation, though this can be changed in the Vehicles.java class by adjusting the amount of iterations of the for loop in the spawnVehicles() method.

I only managed to get three design patterns into the project. The first one was the factory pattern, which is implemented in RoadFactory.java. This helped with creating the different road types that I included in the simulation. The next two patterns are the command pattern as well as the state pattern, which I used together to help with cycling the traffic lights. These can be found in CommandCycle.java, TrafficLightState.java, NSGreenEWRedState.java, and NSRedEWGreen.java