# Problem Specification

**What is the problem?**

The problem given was to build a traffic simulator for a city. The program will need to be able to have a customizable map, of varying sizes, and allow the user to build the city during the ‘map editing’ section of the simulation. When running the vehicles must be able to automatically enter the map, and travel through the city with random destinations without hitting another vehicle and maintaining all Australian road rules.

**How will the program be used?**

The program will be used to help design and plan out a city. This will be by displaying useful information to the user whiles the simulation is running, such as the total number of cars on the simulation, and the average speed of each car on the simulation

**How will the program behave?**

When the simulation is running the program will update incrementally, with all vehicles updating appropriately (moving / staying in the same location) through each update of the simulation. Whiles the is happening cars can only enter from sides of the maps with road entry points on them, and most obey all Australian speed limits, and give way / brake appropriately when needed.

**Map Design**

When building the traffic simulator there was two main problems to consider, how the city would be designed, and how cars would move through the city and road pieces.

For the first class problem it was decided to use a grid like fashion (e.g. 4x4 / 5x5 / 10x10) to facilitate areas to place road maps, and to have a simple number system which is scalable based on the size of the map that the user enters.

1, 2, 3, 4

5, 6, 7, 8,

9,10,11,12

By using this design cars can move around according based on (right - +1, left -1, up – map size, down + map size).

**Movement design**

For vehicles to move around the simulation vehicles were given a road location, map location and direction. This allows the simulation to move pieces accordingly, based on where it is and what direction the vehicle is going

**Data design (types of lists & saving)**

To run the simulation all data during the map editing stage (Road pieces and traffic lights) are saved into an Array list (which are saved to a txt file if chosen to be saved). These lists are then passed to the simulation, with vehicle list changing throughout the simulation (location, location on road and speed) and the traffic light list and road list effecting how the vehicle moves throughout simulation.

## Class Design

**Road**

The road class was decided to consist of 3 constructors, name, orientation and location which allows the simulation class to see what type and where the road piece, as well as what way the piece is facing (e.g. for 3-way intersections and where the exists are). A print method was added to display what road piece and where it was added once the user has done so

**TrafficLight**

The traffic light class was built with multiple constructors to facilitate a variety of types of light cycles and road types. The location and road location determines where the traffic light is on the map and road piece. The roadName attribute is used to determine how the traffic light responds (e,g, 3-way and 4-way intersections had different light cycles) with the traffic light cycle and number used to track where the lights are on the intersections, and what part of the rotations they are through. Methods for both 2-way and 4-way intersections have been added to cycle through the different combinations turning each light green in red from the simulation using the changeColour method, which changes the lights colour. For ease of coming a 2 color light system has been utilized until a prototype of the program is running, where amber will be added to the traffic light colors. To stop traffic lights changing colour straight away, a colour timer was added with a method that ensures that the TrafficLight waits at least 5 seconds.

**Vehicle**

The vehicle class was built using the id constructor to allow references to a specific vehicle (to add it to the map), and the name to references what type of vehicle it is, in combination with length based on the size of the vehicle. To determine where the vehicle class will be located, 3 constructors where used, location and road Location like the previous classes, and road Direction to determine which way the vehicle is going, and what side of the road, and where on the road for intersections. Whiles the road Side was initially used in code, it was released that it was redundant, and road direction could be used in the same way, and will need to be removed from the class. The chosen direction constructor is used when the vehicle needs to choose which direction to go (such as 2-way and 4-2ay intersections) and will correspond with which exit it should take on the road. For now the Speed constructor has been simplified to a 1 to 5 scale for speed (multiplied by 10 for output) to allow the cars to move throughout the map, however will need to be redesigned to something more precise later on. To change the speed of the vehicle to methods were made, accelerateVehicle and decelerateVehicle, with the methods increasing the speed (up to 5) or decreasing the speed (down to 1) respectively. A stop vehicle method was use to set the speed to 0, which will stop the vehicle moving. To move the vehicle a basic method was added which adds the speed of the vehicle to the road location of the vehicle, which moves it across each road piece.

**Simulation**

The simulation class is what is used to run the simulation by extending the timer class, updating the simulation each second. The simulation class needs to be cleaned up, and has a lot of over excessive coding and has become fairly messy, however currently runs through a serious of conditions for the vehicle to accelerate (until reaching the speed limit), and if not will deaccelerate or stop.

Before the simulation begins moving vehicles the code for the traffic light rotation is run, and then checks for how many vehicles then user entered, then allows them enter through each section of the road by running 4 methods, enterTopMap, enterBottomMap, enterLeftMap, enterRightMap which checks for road pieces that are valid entry points, then checking to make sure there is no cars at the entry points using checkVehicleList, then adding the vehicle if no vehicles are there to the road using addVehicle.

Once vehicles are running the simulation moves around the vehicles based on various conditions , moving the vehicle based on what direction it is going and whether or not there is any vehicles in front of it, where the vehicle wants to go (based on a random number generator) and having to give way, and traffic lights. Once the simulation has run through and all cars have left the map, the simulator then exits and returns the user to the initial menu.