The working document:

To help you plan the software development process, your working document is expected to describe the following:

* Problem Specification: What exactly is the problem? How will the program be used? How will the program behave?

The problem is to design a car traffic simulator, at this stage the simulator only needs to include basic functionality utilising for the core classes that will be needed to represent the basic objects such as cars, roads and traffic lights. The final program will have functionality to create and modify maps that comprise of a 2 dimensional layout of roads and traffic lights. However, for this stage of the program, the simulator will just comprise of 2 roads, a car and a traffic light. The simulation will be contained inside the main class for the first part of the assignment and produce some statistics about the simulation as it is running.

Problem decomposition using UML class diagrams: What objects will be used and how will they interact?

* Divide the problem into objects
* The UML class diagrams should answer the following design questions:
* Class design:

1. What role(s) do objects of this class perform?
2. What member fields do objects of this class need? Should they be public or private?
3. What methods do objects of this class need? Should they be public or private?

* Method design:

1. What should its method signature be?
2. What task will it perform? What algorithm will it use?

What classes will need to be built and what functionality will they have?

Road:

* The road class will need to be able to track car objects and where traffic lights are positioned. This will be implemented by dividing the road into segments. Each traffic light object will have an associated static road and segment in that road where it is located. Each car (or vehicle) object will move from segment to segment which will allow its speed to be roughly calculated.
* Member fields (these fields will be represented by private variables, which will be accessed using a get method to return its value):
* Speed limit to represent the maximum speed that cars will be allowed to travel at.
* Number of segments, as the length of a road is set out by the project requirements of being between 6 to 15 (units) times the length of a car (2 to 5 times the bus length). Let 1 unit be the length of a road segment, thus roads will range from 6 to 15 segments long. This is an intrinsic characteristic
* Orientation, either horizontal or vertical. This isn’t an important variable for the initial part of the simulation but will be needed later to graphically position the road objects.
* Road ID, this will be a 2 dimensional value to represent the position of the road graphically. But, for initial functionality this isn’t important as the simulation will only consist of 2 roads so the position can be represented 1 dimensionally for now.
* Methods:
* Overloaded constructors to manage the assignment of variables to the object. Its method signature will consist of the method name (same as the class name, Road) and the multiple parameters that it takes.
* Methods to return the values of the roads speed limit, number of segments, orientation and position of the road. The method signature for these methods is just the name of the method, as they take no parameters.

Car:

* The car class will need to be able to represent a car object with its position (what road and segment of road is the car currently on), the car will also have some internal characteristics, the cars length and width. The length and width of the vehicle will be altered if the vehicle is a bus or a motorbike (this will be accomplished with an overloaded constructor that will be implemented later). The length of a car will be used to determine the values of length for all other objects.
* Member fields:
* Type, this field will allow for the implementation of motorbikes and busses later on and will identify what type of vehicle it is.
* Road number, to identify which road the car is currently on.
* Segment in road, which segment the car is currently located.
* Methods:
* Two overloaded constructors to allow a default vehicle (car) to be created and its attributes assigned and another which will allow a bus or motorbike to be created.
* Drive, which will move the car object up a segment if it is able to (ie if there is a traffic light at the next segment).
* Calculate and return the speed of the car
* Stop, which will prevent the car from advancing to the next segment.

Traffic light:

* The traffic light will need to switch between go and stop modes. It will be located on a road and a position on that road.
* Member fields:
* Status, a Boolean value to represent the red or green status of the traffic light. This may be changed later to also implement functionality for a yellow status.
* Road number, to identify where the traffic light is located.
* Segment in road, which segment of the road is the traffic light located.

City:

* The city will need to store a layout of roads and traffic lights. Possibly by storing and reading data about the layout of the city in an external file.
* Member fields:

The road, car, traffic light and city classes will also need to have associated unit test classes.

Main:

* The main class will act as the simulator for the first part of the assignment. Which will just create 2 roads, a traffic light and a car object. The program will intermittently call methods that return the position of the car on the road and its speed.