Kyle Zalewski

10/08/2018

CS202

Lab Section: Mon 2pm-3:50pm

My goal with this assignment is to attempt to structure my hierarchy in a way that makes the use of “get” functions unnecessary. Hopefully I can eliminate them entirely by constructing methods that more specifically perform necessary tasks. The idea behind a “getter” is basically a hand-off to another task, so if I can eliminate the need for this “middleman” function, my code will be more concise, less confusing, and more efficient. This will make the design and implementation process much easier and less convoluted.

To maintain object-orientation, I’m going to ensure that all my classes have single inheritance. This should be relatively easy to do and will make the hierarchy more sensible and allow the location of various obstacles to the car to be more easily defined and worked with. Proper division of responsibility should ensure that multiple inheritance is not necessary.

I intend for the overall base class to be location. From here we can derive classes for space occupied by a vehicle and space that is available for the vehicle’s use (the class called “lane”). The base class will have the speed limit of the street as a private data member, as at this point, I don’t see that information needing to be directly accessed by any child function. This may change as the design takes shape. The lane class will contain and manage an array of linear linked lists to represent empty space in the street, as well as lanes other than car-dedicated travel lanes.

Child classes for lane will be “bus” and “bike” that will each have special characteristics and functions specific to those types of travel lanes. For instance, the bus lane will have a function that takes information on whether the lane is dedicated or not, and whether there happens to be a bus in the lane at the time, and decides if that lane is usable. The bike lane is similar, but will include extra functionality to determine if it is a lane that can be traversed by the car (like a shared lane) or whether it needs to be considered when switching between two car lanes (as if the bike lane were positioned in the middle).

The car class will be derived from location. This class will be responsible for the array of linear linked lists that represent the locations of other cars on the road. There will be functionality in this class that dictates whether the car is able to move at present, whether manual control is necessary, and the distance from other vehicles adjacent to it. Specifically, the vehicle will be able to see distance to the car in front, as well as if there is a car to either side of it.

The class “followingDistance” will be a child to “car” and will be responsible for most of the safety protocol between the self-driving car and the car in front of it. This class will be the one to contain information on vehicle speed and road conditions (specifically rain) that will allow calculations to be made for required stopping distance, and whether the actual amount of space requires manual control to be taken by the driver to evade to other lanes and avoid a collision as they see fit. I will most likely look to work out a solution where this scenario is impossible, and the car will keep proper distance in case the driver ahead is somehow able to immediately stop regardless of speed.

As the design sits in its present form, I believe I have the groundwork for a solid structure. I may attempt to bring lane and car together somehow, if managing two separate data structures for occupied and unoccupied locations becomes tricky to manage.