Self-driving Car Project

Software Development 1 – Matthew Meyer

The first thing I want to mention is that I have never used C# before this project, but I wanted to dabble a little bit and I figured this skeleton code project was a great opportunity. I ended up with the Program class where the main method lives and 3 other classes. The other classes are Car, Routes, and Navigation. I tried to follow the Single-responsibility principle. In my high level design assignment I was looking at it from a “ride-sharing” perspective where as this one I viewed it from a personal vehicle perspective.

Car class contains all the methods the car would perform such as starting, driving, etc. Pretty basic.

The Routes class was designed with the aggregation of new and old routes in mind. It essentially just gets the name of the route and stores it in a list. I had originally thought the have it store the specific route steps (left, straight, right) as well, but technically a GPS route is dynamic depending on traffic, road closures, etc. Thus, we want the route to be checked and re-calculated each time. It could be reasonable to have the routes list store objects with a name and associated address which is passed to the GPS (Navigation class) where the route is checked for changes, but I did not attempt to implement that here.

The Navigation class was sort of designed as a stand-in for the GPS system our self-driving car would be connected to. In this case, it has a method to calculate the navigation which is passed the route name. I then hardcoded a one-way trip that mirrors my actual route to work to my old job by doing a series of method calls on a new car object.   
I would like to note here that I did this entire project incorrectly at first. I did not carefully read the directions. Rather I took in what the result was supposed to be but ended up forgetting the instructions about having a method for each input/direction. I initially wrote the output strings as variables containing each string and chained them together inside the calculateNavigation method. This chain of string variables was assigned to a variable I called path which the method then returned and the main printed to the screen. I do not know why I chose to do it this way at all, but when I reread the directions before submitting, I changed all the string variables in the Navigation class to methods in the Car class.

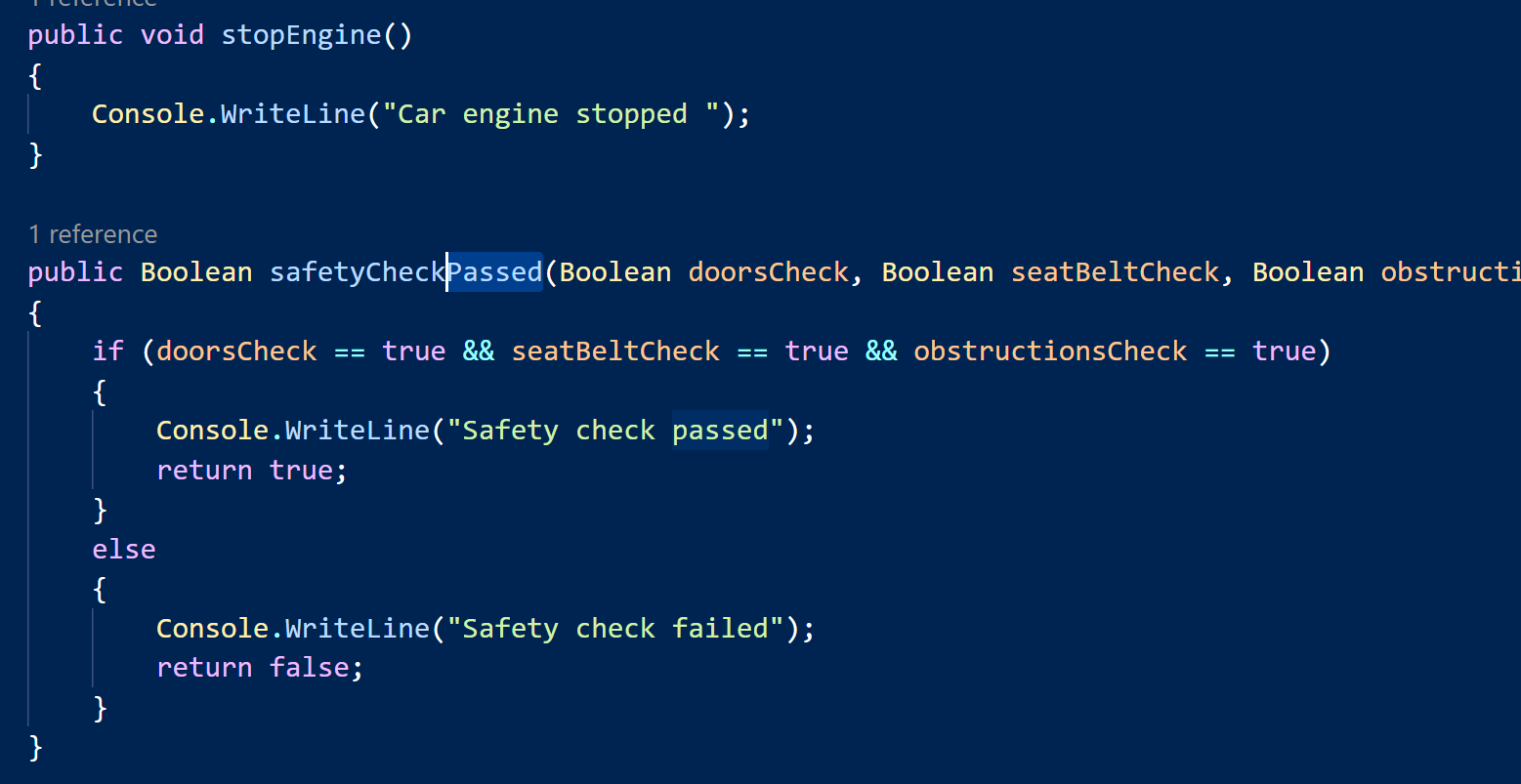
See other pages for screenshots from the files and the console.

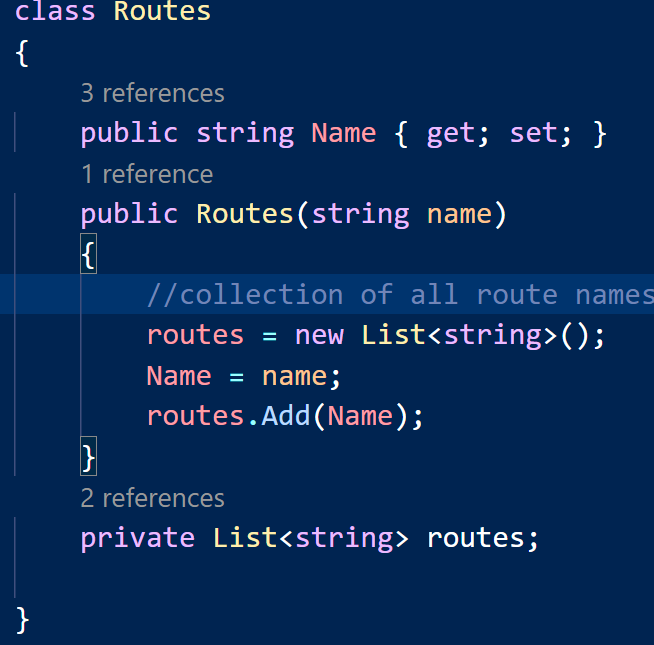
The created this project as a .Net console application, so here is a sample output using dotnet run in cmd.  
A screenshot of a computer

Description automatically generated with medium confidence

This is Program.cs and the main method. Realistically, the name passed to the new route object would likely be a variable that is assigned via some sort of user input such as text, selecting from a list or a voice command.   
Graphical user interface, text

Description automatically generated

An example of a couple methods in the Car class.  


The Routes class.   


This is originally how I had setup the calcNavigation method inside of the Navigation class. Graphical user interface, text

Description automatically generated