Programming Assignment #1
CS 202 Programming Systems

***** Make sure to read the Background Information first!**
It applies to all programming assignments this term***

We do not accept late work beyond the late due date. There are no exceptions.

Make sure to back-up your files PRIOR to using tar
Double check that the arguments specified with for tar are correct!
Late penalties will apply to all submissions incorrectly archived/uploaded

Background:
When beginning with this project, the first thing to keep in mind is that we are no longer working on CS163 programs! In CS163 we were concerned about creating Abstract Data Types and the class construct facilitated this. Instead, this term we will be focusing on how to create Object Oriented Solutions. An ADT may be part of that solution – but it certainly shouldn't be the primary focus. Instead you want to strive for classes to have specific "jobs" and have classes derived from more general classes, whenever appropriate. We will be working in situations where there are multiple classes, so you will want to focus on dividing the design into smaller components that have specific jobs working together to solve the problem.

Every assignment this term needs to have at least 5 classes. With these, think about how to design the classes such that they reduce the amount of work another class needs to do. The idea is if we have "robot" like classes doing the smaller tasks or "jobs", that by the time we get to a larger class that has more to manage – it will have little left to do! We can achieve this by delegating. Often the over-use of "getters" can cause the opposite to happen – and instead of delegating the managing class has to fundamentally do all of the work itself.

Overview:
We just had an incredibly long break between Fall and Winter terms. During this time we had some unusually bad weather events that brought our city to a screeching halt. I found the MAX and even the streetcar invaluable resources to get around when other methods were not practical. As I sat on the street car, leaving my vehicle at PSU, I thought about the programming that exists behind the scenes, and the power that OOP could bring to software managing streetcar schedules.

Program #1

For Program #1, you will be creating an object oriented program that will create a streetcar scheduling program that will allow a controller to observe where all of the streetcars are located and determine when it is time to take a streetcar out of service. The following represents some ideas on how a streetcar scheduling system could work; you have the liberty to add other ideas and classes to this.

Some of the basics that are part of an OO program managing streetcars **could** be:

1. Line – The information about a particular streetcar line. In Portland, we have three streetcar lines. Each has a name, a number of stops, and then a type (e.g., some are loops and others are lines (that go back and forth)).
2. Streetcar – The particular information about a street car. Streetcars could have an identification number (e.g., Vehicle #S002) along with its current location and status (in service or not).
3. Location – The location can only be on a route and going a particular direction (e.g., Route: A loop, Direction: A loop to Lloyd via Pearl, Closest Stop)
4. Stop – A place where the Streetcar comes to a stop and lets in/off passengers
5. Pace – At any Stop, a streetcar should pause and not continue if it is within N (e.g., 4) stops of another streetcar. The controller should be able to specify the spacing. For example, at rush hour, there may be more streetcars in service with a smaller/closer pace.

Once you have the basics, develop a Map that includes two lines, with at least four streetcars per line, and 20 stops. A Controller should be able to Start a streetcar on a line, set the pace of the streetcar (how frequently it moves from stop to stop based on the location of the other streetcars), find out where all of the streetcars are located, trace a particular streetcar, and take a streetcar out of service.

To make this Object Oriented:

You will want to first think about breaking this down into a series of classes and create them independent of the entire problem. Some relationships should be hierarchical, others can be containment. With hierarchies always push the common elements up to the base class. Remember to avoid classes with only setters and getters! And Nodes will need to be classes instead of structs. Here are some suggestions to start with:

1. Streetcar IS its location plus more
2. A Line IS its type
3. A Line can have multiple streetcars on it and at different locations
4. A direction is one of the different directions available based on its line
5. The Map has multiple lines
6. And more! Can you do something with Route or Stop? Remember, you need a total of at least 5 classes!

Anything that is similar between these or other classes that you write should be pulled up to be part of a base class. For example, classes that manage collections of items may be derived from a common base class that manages the collection. Keep classes small and functions small. A large class or function means that the problem has not yet been broken down into its basic components (objects).

Data structures

This program should implement the following data structures:

1. A Graph to handle the map of directions, stops and lines should be implemented as an Adjacency List. All traversal functions should be implemented recursively.
    a. Any Line that is a loop should be implemented using a circular linked list of Stops
    b. Any Line that is a line (back/forth) should be implemented using a linear linked list of Stops