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 date. No exceptions.

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.

Overview:
Brick and mortar stores are facing a crisis with the new online shopping and fast delivery alternatives. Stores like Toys R Us are soon to be gone after failing to reinvent themselves. Amazon and others have revolutionized how we can receive items with minimal delivery costs and within a very short period of time. Delivery used to be limited to UPS or Fedx, but now is wide open with personal (e.g., Uber) delivery options.

Program #1
For Program #1, you will be creating an object oriented program that will provide three different types of delivery service for local items: Standard, Express, and by Drone. Here is a quick summary of each of the types of delivery – you may add additional characteristics to each of these:

1. Standard – Driver receives a flat rate of pay per package. They can fit as many packages as possible into their vehicle. Each vehicle will have a set number of packages that they can hold (e.g., a van can hold more packages than a smart car!). A driver for a standard delivery cannot pack-in more packages than they have room. For standard dellivery, the driver will find the best route to take to deliver all of the packages, but time is not of the essence. They should be able to provide the company with an estimated delivery time to be finished for delivering all of their packages.
2. Express – Driver receives pay based on time. The least amount of time to deliver the package will receive the most pay (inversely proportional). The pay calculation should include information about the distance and the weight of the package as well. Express delivery must find the fastest route to deliver the package. An express delivery service can have up to 3 packages being delivered at any one trip.

3. By Drone – Delivery by drone means that only one package can be delivered at a time. The package must be under 10 pounds and under 1 sq ft. They do not need to find a route (since roads are not used).

To find the best route for Standard and Express delivery, you will be implementing a weighted graph. It should include some basic road choices (freeway, main roads, some short cuts) for at least 20 locations for the purposes of our program. You are welcome to do more than this. The data for each edge needs to include the speed limit for that road and the number of miles for that edge (segment of the road). This is important because drivers may never exceed the speed limit!

To make this more interesting, the Express driver may encounter "interruptions". An interruption could be (a) traffic slow down, (b) road closure, or (c) equipment failure. Each of these will delay the driver for a different period of time and may require a new route to be determined. Use a random number generator to control if an Express driver encounters an interruption and the severity of that interruption (have some fun with this part!!).

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. Here are some suggestions to start with. With hierarchies always push the common elements up to the base class.

Remember to avoid classes with only setters and getters!

1. Delivery – since all three different delivery services have common elements about them, there could be a common base class. It could hold information such as (a) point of origination, and (b) list of packages.
2. Standard, Express and by Drone are all forms of delivery and may be derived from a Delivery class
3. Think about what a package is? It has a weight and a size. It has (or is) a specific destination.
4. Think about what an interruption is? There are traffic jams, road closures and equipment failures. All three will be similar in that they will delay the Express driver. But, all three are different in that a traffic jam will just slow us down but a road closure will require a different path to be taken through the graph. An equipment failure may mean we have to wait for a tow truck or a part to be delivered! This in itself could be a hierarchy!
5. And more! You need a total of at least 5 classes!
6. And Nodes will need to be classes instead of structs….

Anything that is similar between these or other classes should be pushed up to be part of a base class. 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
The following are the data structure requirements:

1. An adjacency list (an array of vertices where each vertex has an edge list which is implemented as a linear linked list) for the routes.
2. A circular linked list of the route chosen to take for deliver for Standard and Express service
3. A linear linked list of packages to deliver for the Standard and Express delivery choices. These should be re-ordered to be in the order that they will be delivered.
4. All arrays must be dynamically allocated.
5. Use recursion rather than iteration for traversal!