CS520 Project

General Rules for Project and Homework Assignments

- You are strongly encouraged to add comments throughout the program. Doing so will help your facilitator to understand your programming logic and grade you more accurately.
- You must work on your assignments individually. You are not allowed to copy the answers from the others. *However*, you are encouraged to discuss the approaches to the homework assignments with your section mates and the facilitator in your section via the discussion board.
- Each assignment has a strict deadline. However, you are still allowed to submit your assignment within 2 days after the deadline with a penalty. 15% of the credit will be deducted unless you made previous arrangements with your facilitator and professor. Assignments submitted 2 days after the deadline will not be graded.
- When the term *lastName* is referenced in an assignment, please replace it with your last name.

You are strongly encouraged to add comments into your program!

Create a new Java Project in Eclipse named PROJ_*lastName* and complete the following two parts.

Project

This project will demonstrate a simulation of a dockyard. The dockyard regularly has ships arriving with many containers that are offloaded into the dockyard and stored in an area of the dockyard for later pickup.

As each ship arrives, it has a manifest which lists all the containers and the destination city for each container. You need to offload the containers and store them in a staging location within the dockyard by destination city. You should notice that as ships arrive these staging areas start getting filled up with containers.

At any point, trucks can arrive to pick up a single container. The dockyard will load a container onto a truck based on the destination city of the truck. The dockyard will use the truck's destination city to get the next container in the queue for that destination city and place it on the truck for deliver. As more and more trucks arrive, you should notice that the staging areas for the truck destination cities become freed up since the containers are leaving the dockyard.

The starting point for the project is uploaded to the class discussion area in blackboard.

You will need to do the following:

1. There are interfaces defined for a container, a ship, and a truck in the shipping package that you will need to write implementation classes for. The implementation details are up to you as long as you implement the methods to satisfy the expected inputs and outputs.
2. There is an interface defined for the dockyard that you will need to write a class for. The dockyard needs to be able to store containers in different staging areas by destination city.
3. You will need to complete the implementation for the ShippingProcessor class. There is a base class the provides default implementations the ShippingProcessor should leverage.
4. Implement any custom exceptions you think are appropriate.
5. **All of the code you need to implement should be created in a package named solution.**

We will discuss approaches in class.

NOTE: You are not to change any interfaces or code within the shipping package.

Sample output based on the input files provided:

```
The dockyard contains 0 containers.

A new ship has arrived. Processing...
Before unloading: Ship S123 has 6 containers.
```

```
After unloading:  Ship S123 has 0 containers.

The dockyard contains 6 containers.
        LA: 1
       BOS: 1
       NYC: 2
       ATL: 2

A new ship has arrived. Processing...
Before unloading: Ship S124 has 8 containers.
After unloading:  Ship S124 has 0 containers.

The dockyard contains 14 containers.
        LA: 2
       BOS: 4
       NYC: 4
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T223 is headed to BOS with no container.
After loading:  Truck T223 is headed to BOS with container C123
The dockyard contains 13 containers.
        LA: 2
       BOS: 3
       NYC: 4
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T224 is headed to NYC with no container.
After loading:  Truck T224 is headed to NYC with container C124
The dockyard contains 12 containers.
        LA: 2
       BOS: 3
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T225 is headed to BOS with no container.
After loading:  Truck T225 is headed to BOS with container C523
The dockyard contains 11 containers.
        LA: 2
       BOS: 2
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T226 is headed to BOS with no container.
After loading:  Truck T226 is headed to BOS with container C529
The dockyard contains 10 containers.
        LA: 2
       BOS: 1
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T227 is headed to LA with no container.
After loading:  Truck T227 is headed to LA with container C125
The dockyard contains 9 containers.
        LA: 1
       BOS: 1
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T228 is headed to BOS with no container.
After loading:  Truck T228 is headed to BOS with container C530
The dockyard contains 8 containers.
        LA: 1
       BOS: 0
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T229 is headed to LA with no container.
After loading:  Truck T229 is headed to LA with container C525
The dockyard contains 7 containers.
        LA: 0
       BOS: 0
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T230 is headed to BOS with no container.
After loading:  Truck T230 is headed to BOS with no container.
The dockyard contains 7 containers.
        LA: 0
       BOS: 0
       NYC: 3
       ATL: 4

A new truck has arrived. Processing...

Before loading: Truck T231 is headed to LA with no container.
After loading:  Truck T231 is headed to LA with no container.
The dockyard contains 7 containers.
        LA: 0
       BOS: 0
       NYC: 3
       ATL: 4

The dockyard contains 7 containers.
        LA: 0
```

```
BOS: 0
NYC: 3
ATL: 4
```

Submission:

Create an archive of your Eclipse project using the following steps. Select the PROJ_*lastName* project in the Eclipse IDE's *Package Explorer* or the *Navigator* window.

Click *File->Export*. Select the *General->Archive File* option. Click *Next*.

Specify the "*To archive file:*" entry as say, C:\Temp\PROJ_*lastName*.zip.

The zip file will be created and stored in the C:\Temp folder.

Submit this zip file as an attachment in the Assignment Section.