List Containers

Purpose:
The purpose of this lab is to introduce list container classes.

Setup:
If it does not already exist, create a subdirectory named Lab23 in your Java directory, and a subdirectory inventory in Lab23. Copy everything from ~labCourse/Labs/Lab23/inventory/ into your inventory directory.

Containers and lists:
A container is an object whose function is to contain a collection of other objects. A list is a container with the following properties:

• it can hold any (finite) number of objects;
• all objects on the list must be instances of the same class;
• each object on the list can be accessed by an integer index denoting its position on the list.

(We will see later that with sub-classing, the restriction that all objects on the list must be of the same class is not severe.)

The class RetailItemList:
In this lab, you will use the class RetailItemList. A RetailItemList can contain, surprise, only instances of the class RetailItem. These classes are specified here.

Note in particular the following methods of the class RetailItemList:

• size returns the number of objects on the list. The size of a list can be 0, in which case the list is empty: it contains no elements.
• get returns the object at the specified position (index). Indexing starts at 0. If a list contains 10 objects, the first object is at index 0 and the last object is at index 9.
• append adds an object to the end of the list.
• remove removes the object at the specified index position. This does not leave a “hole” in the list. For instance, if the object with index 3 is removed, the object that was at index 4 “moves up” to 3.

The class Inventory:
To see how a list can be used, we first examine a simple class that uses a RetailItemList.

• Open the file Inventory.java and read the definition of the class Inventory.
Open InventoryTest.java and read the test code.

Compile and run Lab23.inventory.InventoryTest.

The class InventoryTest manipulates an Inventory, which contains a list of references to RetailItem objects. The first step is to create an Inventory.

- What operation creates the Inventory?
- How many items are in a newly created Inventory?
- When an item is added to the inventory, what list operation is used?
- How many items can be added to the inventory?
- A client needs to know if an item with a given item code is in the inventory. How can the client achieve this?
- A client needs to know the net price of the cheapest item in the inventory. Do any of the features of a Inventory accomplish this?
- We want to add a feature to the class Inventory to answer the above question. Should it be a query or a command? State pre-conditions and post-conditions for this feature.
- Consider the logic for searching for an item with a given item code in the inventory and the logic for finding the cheapest item in the inventory. Both involve traversing a list of RetailItems, with one major difference. What is the difference?
- The client needs to know the total value of the inventory, based on gross price. Can the client accomplish this, or must we provide additional features in the class Inventory?

**Modifications to the class Inventory:**

Now you will add three methods to the class Inventory. Examining the methods already defined in the class, particularly the method numberOutOfStock, should give you some clues as to how to proceed.

- First, implement a method totalUnitsOnHand. This method should return to total number of units on hand for all retail items in the inventory.
- Second, implement a method outOfStock. This method should return a RetailItemList containing all of the items that are currently out of stock: that is, items for which there are no units on hand. Note that this method should create a new RetailItemList, and append RetailItems to it.
- Third, implement the method cheapest. This method should return a reference to the cheapest item (by net price) in the inventory.
- Modify the test code to test your methods.
- Compile and test.
Post-lab:
Submit the following, as directed by your lab instructor:

- answers to questions posed in this lab;
- a browser printout of the specifications of the class *Inventory*, including specifications of the three added methods;
- listings of the modified *Inventory* class and modified test code;
- a script listing showing the results of your test.