Programming Assignment #2

General Description:
In this assignment, you are to work on a scenario where you are helping manage a person's stock transactions. In this simplified scenario, the user has the option to purchase a set number of shares at the current stock price, sell a set number of shares at the current stock price, and to view the price of the longest-held set number of shares. When a sale is made, the capital gain/loss for the transaction is to be displayed to the user. Additionally, upon exiting the program the total gain/loss for all transactions is to be displayed. In order to have a manageable assignment, some simplifications to the "real world" of stock purchasing are being made.

Specifics:

1) Use a linked list of nodes with each node representing a purchase transaction. Name the class **purchaseNode** and the instance variables are as follows:

```
private float purchasePrice;
private purchaseNode link;
```

This should be designed as an ***inner*** class.

2) Write a class that implements the **MyQueue** interface that was given in class. To start, download the **MyQueue.java** file from the course website and ensure that it compiles successfully. ***IMPORTANT***: You are **NOT** to copy this interface into your own file(s).

3) Create a class, name it **UserQueue**, to implement the **MyQueue** interface. This class will implement the queue ADT via a linked list of nodes. Make the **UserQueue** class an ***outer*** class that acts as a container for the individual nodes.

4) In class **UserQueue** the methods of **MyQueue** must be implemented. Additionally, the **front** and **back** pointers for the queue must be defined, and an appropriate constructor(s) must be written. No additional instance variables are permitted.

5) Next, create the class **QueueApp** that will house the **main** method.

6) In the **main** method there should be a menu of choices offered to the user which repeats until he/she decides to quit. The menu should match the following:

Please enter your selection:

0 Purchase 10 shares of stock.
1 Sell 10 shares of stock.
2 Check the purchase price of the oldest 10 owned shares of stock.
3 Quit

{continued on the next page}

The actual printing to the screen of the menu and the acceptance of the choice should be done in a ***separate method*** within the **QueueApp** class.

7) When the user wants to purchase or sell stock, he/she must be asked for the current stock price. For purchases, this value is inserted into the new purchase node that is created. For a sale, this value is used when computing the capital gain/loss. The formula for this value is:

gain = 10 * (sale price - purchase price) ***{if negative, this is a loss.}***

When the user chooses option 3 (quit), the **total** gain/loss for all sales should be output to the terminal screen. Note, if a sale is never done, no gain/loss is realized.

8) For each of these menu options, provide a robust implementation. This means, for example, that if the user tries to dequeue an item from an empty queue the program does not crash. Rather, it would output an appropriate message, print the menu, and accept a new selection. You can assume that if a floating point value is **asked for** the user will respond properly by entering a floating point value.

9) Completely test your program editing it, if necessary, so that it correctly performs all operations.

10) Make sure to properly document your program.

11) Submit a "Statement of Functionality" as a .doc/.docx file. Specifics of this will be discussed in class.

Submission Information:

The source code for this assignment is due by 11:59pm on Tuesday, March 4th. Put all three of your classes into one file name QueueApp.java and upload it to the appropriate folder in Canvas. Do **NOT** submit the MyQueue.java file as your files must work with the Instructor's interface. Additionally, do not define any of your own packages. Make sure that your code is properly formatted and well documented. Additionally, submit the "Statement of Functionality" doc/docx file to the same folder. On Wednesday, March 5th, submit printouts of your code **and** the "Statement of Functionality" at the start of the lecture period.