CSCI 145 Week 10
Assignment 3

Assignment Description

A book store stocks a number of book titles. The provided file `Books.txt` gives the details of the inventory, one book per line, giving the ISBN, price and current stock level. Transactions for the inventory system are provided in the file `Transactions.txt`. Two types of transactions have to be handled by the inventory system:

- STOCK transactions, which specifies the ISBN of a book, and the number of copies of that book to be added to the stock level.
- ORDER transactions, each of which specifies the ISBN of the book being ordered, the number of copies of the book required by the customer, and a 6-digit customer number.

If an order cannot be filled completely because of insufficient stock, the unfilled part of the order (the entire order if there is zero stock) must be placed on a queue of back orders for that book.

When new stock is received for a book, the inventory system must work through the back order queue for that book, filling as many back orders as it can. If a back order can only be partly filled before the stock level of the book drops to zero, the unfilled remainder of that back order must remain at the head of the back order queue for that book.

You must use the provided queue type (`Queue.java`) for your back order queues.

For full credit, you should submit your program by 11:59 pm on Sunday, March 14th, 2016. This assignment will be worth 15% of your grade for the course.

Please read the entire assignment before beginning work.

Requirements

1. The book file will be formatted as follows (fields are separated by spaces):
    - An ISBN (up to 10 Characters)
    - The book price with two digits after the decimal point
    - The current inventory amount (an int).

    The file `Books.txt` provides a sample of the input.
2. The transactions file will be formatted as follows:
    - The string “STOCK” or “ORDER”
    - An ISBN
    - The number of books to add to stock (STOCK transaction) or the number of books to being purchased (ORDER transaction)
    - A customer number (an int) (ORDER transactions only).

The file `Transactions.txt` provides a sample of the input. All ISBNs will refer to a book that is in the inventory (possibly with zero stock).

3. Your program must get the names of the book file and the transaction file from the command line. If fewer than two command line arguments are provided, the program must display a message and terminate.
4. If either of the input files cannot be opened, the program must display an appropriate message and terminate.
5. Your program must display one or more report lines for each inventory event. Your program must report when stock is added to inventory and when an order, including a back order is (partially) filled.
6. Your program must keep track of the value of all books shipped. When the program ends, the total value of all books shipped must be reported by the program. The value of any back orders that have not been shipped is not included in the total.
7. You must name the program which contains your `main` method `BookInventory.java`. You are free to have any additional Java files (classes, modules) that you need to complete the assignment.

Notes

1. The file `Output.txt` in `assign3.zip` shows the results from running the program on the provided `Books.txt` and `Transactions.txt` files.
2. You should feel free to copy and modify logic from the previous labs and assignments. Warning: the logic for `getBooks` in `BookStuff.java` (Lab 5) contain a call to `lineData.useDelimiter`. If you copy this logic, leave this line out. (This call changes the delimiter between fields to a vertical bar (`'|'`) character.)
3. The coding will go faster if you break the logic up into multiple methods. That way you can keep the individual methods (relatively) small.
4. The STOCK transaction is the hardest to handle due to the back order processing that may have to occur when stock is added.
5. Use must use the provided `Queue` type for keeping track of back orders. You will want to keep a back order queue for each book, rather than a single queue for all back orders.

   Watch out for the requirement that partial back orders have to stay at the head of the queue. You can peek at the order at the front of the queue, but don't remove it from the queue until you know that the entire order has been filled.
6. Your program will have to maintain a collection of all the books in the inventory. You must use your `LinkedList` class from lab 6 (with modifications if needed), to maintain the inventory. If you make modifications, make sure that you don't impact the workings of the `Queue` type.
7. A utility method to print out the current Inventory and/or the current back orders might help with debugging. If you write these, you can leave them in
8. Here is a small sample of possible input and output (these files are included in `assign3.zip`):

File: `Books1.txt`

```
0061781231 17.99 20
159562015X 24.95 0
```

File: `Transactions1.txt`

```
ORDER 0061781231 5 599930
ORDER 159562015X 12 422099
STOCK 159562015X 6
ORDER 159562015X 8 123456
STOCK 159562015X 12
```

Output:

```
Order filled for customer 599930 for 5 copies of book 0061781231
Back order for customer 422099 for 12 copies of book 159562015X
Stock for book 159562015X increased from 0 to 6
Back order filled for customer 422099 for 6 copies of book 159562015X
Back order for customer 123456 for 8 copies of book 159562015X
Stock for book 159562015X increased from 0 to 12
Back order filled for customer 422099 for 6 copies of book 159562015X
Back order filled for customer 123456 for 6 copies of book 159562015X

Total value of orders filled is $539.05
```

Coding Standards

Your program should follow the standards described as part of Lab 1.

Your program will be graded on conformance to the coding standards as well as correct functionality.

Turn in your program on Canvas

You should make a zip file of all your java files and turn it in on Canvas.