Programming Assignment #6

Bookstore Inventory using a Linked List

CS 2308.003, Fall 2011
Instructor: Jill Seaman

Due: in class Thursday, 11/17/2011 (upload electronic copy by 10am)

Problem:

You will implement the BookInventory class from Programming Assignment #5 using a linked list to represent the book inventory. You should use the Book class (book.h and book.cpp) from the solution on the TRACS site. You will also be provided with a partial implementation of the BookInventory class. The BookInventory.h class will be complete. The BookInventory.cpp file will provide complete implementations for the following functions:

```
BookInventory();
bool addBook(Book);
int removeBook(Book);
bool removeOne(Book);   (private, used by removeBook)
  Note: this functions tries to remove exactly one instance of the book argument
        from the list.  It returns true if it succeeded.
```

You will be responsible for implementing the following functions (described below):

```
void showInventory();
int getTotalInventory();
float getCostOfTotalInventory();

Book findMinimum();   //should be private, but public for testing
void sortInventory();
```

The BookInventory.h and BookInventory.cpp files will be provided via links on the webpage, next to the link for this assignment. You should download these files, and make all your changes/additions to the BookInventory.cpp file.

showInventory: displays a listing of the book inventory to the screen, nicely formatted, one book entry per line. Output sku, then quantity, then cost, then title.

getTotalInventory: returns the total quantity of (copies of) books in the inventory.

getCostOfTotalInventory: returns the total cost of ALL of the (copies of) books in the inventory.

findMinimum: returns the minimum book in the list, using the < operator over the books (the definition of < over books is provided in the Book class).
NOTE: if the list is empty when this function is called, it should return a book made using the default constructor. You won't need to call it on an empty list.

sortInventory: reorders the books in the list, using findMinimum.

Here is the algorithm I recommend for implementing the sort function. It is a form of a selection sort. Basically, it selects the next (minimum) element from the current list, appends it to a new list, and removes it from the current list. When finished, it makes the head pointer point to the new list. "bookList" below is the head pointer.

```
while bookList is not empty
   find the minimum book in bookList
   make a new node using the minimum book values,
   append the new node to end of the new list*
   remove the minimum book from bookList (use removeOne(book))
end while
make bookList point to the new list
```

*maintain a pointer pointing to the last node inserted into the new list.
note: adding the first node is a special case (as in the appendNode function)

I will go over this algorithm in more detail in class.

Input/Output:

You should use the BookDriver.cpp file from the Assignment 5 solution on TRACS. You can modify it if you want, but your solution should give the same results as running it with the Assignment 5 solution. And you don't need to submit it this time.

NOTES:

This program DOES need to be done in a Linux/Unix environment. You should use the makefile from Assignment 5.

DO NOT change the names of the functions or files.

Follow the rest of the style guidelines described here:
http://www.cs.txstate.edu/~js236/styleguidelines.txt
(there is a link to these on the course webpage as well).

Logistics:

For this assignment you will be making changes to ONLY the provided BookInventory.cpp file. You should submit your modified version of that file (don't change the name).

There are **two** steps to the turn-in process (also see the Assignment Turn-in Policy):

1. Submit an **electronic copy** using the following upload link:

   http://www.cs.txstate.edu/~js236/homework
   (There is a link directly to this page on the course website).

   Click on your course number, and log in with your Net ID and follow the directions to upload your file.

2. Submit a **printout** of the file at the beginning of class on the day it is due. Please print your name on the front page, and staple if there is more than one page.