Programming Assignment #6

Small Store Inventory using a Linked List

CS 2308.001 and 003, Fall 2013
Instructor: Jill Seaman

Due:
section 001: in class **Wednesday, 11/20/2013** (upload electronic copy by 3:00pm)
section 003: in class **Thursday, 11/21/2013** (upload electronic copy by 10:30am)

Problem:

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

```
ProductInventory();
bool addProduct(Product);
int removeProduct(Product);
```

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

```
void showInventory();
int getTotalQuantity();

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

The ProductInventory.h and ProductInventory.cpp files will be provided via links on the webpage. You should download these files, and make all your changes/additions to the ProductInventory.cpp file.

showInventory: displays a listing of the product inventory to the screen, nicely formatted, one product entry per line. Output locator, then quantity, then price, then product name.

getTotalQuantity: returns the total number of units of all of the products in the inventory.

findMinimum: returns the minimum product in the list, using the < operator over the products (the definition of < over products is provided in the Product class).
NOTE: if the list is empty when this function is called, it should return a product made using the default constructor.

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

Here is the algorithm you must use for implementing the sort function. It is a form of the selection sort. It selects the next (minimum) element from the current list, appends it to a new list (pointed to by a temporary pointer), and removes it from the current list. When finished, it makes the head pointer point to the new list. "productList" below is the head pointer.

```
while productList is not empty
   find the minimum product in productList
   make a new node using the minimum product values,
   append the new node to end of the new list*
   remove the minimum product from productList
end while
make productList point to the new list
```

*suggestion: 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.

DO NOT create another ProductInventory local to the sortInventory function. I want you to use a pointer that is local to the sortInventory function instead.

Input/Output:

I will put a new ProductDriver.cpp file on the website (very similar to the previous one). Your solution should give the same results as running it with the Assignment 5 solution. You do not need to submit a driver file.

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 classes, functions or files.

- Follow the rest of the style guidelines from the class website

Logistics:

For this assignment you will be making changes to ONLY the provided **ProductInventory.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:

1. Submit an electronic copy using the Assignments tool in TRACS no later than one half hour before class the day the assignment is due (see top of page 1).

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

   If you are unable to turn a printout in during class, you have until 5pm on the day the assignment is due to turn it in to the computer science department office (Nueces 247). They will stamp it and put it in my mailbox. DO NOT slide it under my office door.