Programming Assignment #6

Small Store Inventory using a Linked List

CS 2308.251 and 252, Spring 2015
Instructor: Jill Seaman

Due: before class **Wednesday, 4/22/2015** (upload electronic copy by 9:30am)

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 (Resources tab). 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);
bool removeProduct(Product);
```

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

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

ProductNode *removeMaximum();  //private function
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 (only!!)

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.

removeMaximum: finds the maximum product in the list, using the > operator over the products (the definition of > over products is provided in the Product class). It removes the node of the maximum from the list and returns a pointer to this node.

NOTE: if the list is empty when this function is called, it should return NULL.

Step 1: Do a linked list traversal to find the value of the maximum (greatest) Product element in the list. Store it in a temporary variable like maxProduct. This is exactly like finding the maximum value in an array, but you are traversing a linked list instead.

Step 2: Re-use the code for deleteNode from class (or removeProduct in the ProductInventory class) to remove the node containing the maxProduct value from the original list. Do NOT **call** removeProduct, it will deallocate the node! Copy the code and revise it so that instead of deallocating the node it will return the pointer to the node (the address) as the result of the function.

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

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

Here is the algorithm you must use for implementing the sort function. It is a form of the selection sort. It selects the next (maximum) element from the current list and removes its node from the current list, and then inserts the node at the **front** of a new list (pointed to by a temporary pointer). When finished, it makes the old head pointer point to the new list. "productList" below is the head pointer.

```
while productList is not empty
   find the maximum product in productList and remove it (i.e. call removeMaximum)
   add the maximum node to the front of the new list
end while
make productList point to the new list
```

The new list will contain the elements in ascending (increasing) order.

Input/Output:

I will put a 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.

NOTES:

- This program should be done in a Linux/Unix environment. Use the PA5 makefile.
- DO NOT change the names of the classes, functions or files.

- For sortInventory and removeMaximum, you must NOT dynamically allocate any new nodes or delete/deallocate any existing nodes. Do NOT copy a ProductNode's value to another ProductNode. Reuse the existing nodes!!

- **DO NOT create another ProductInventory** local to the sortInventory or removeMaximum functions. I want you to use ProductNode pointers that are local to these member functions instead.

- **DO NOT call other member functions** from the sortInventory or removeMaximum functions (except to call removeMaximum from sortInventory).

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 on the TRACS website for this class.

2. Submit a printout of the **ProductInventory.cpp** file 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 (Comal 211). They will stamp it and put it in my mailbox. DO NOT slide it under my office door.