James Sandoval 01/31/2015

Program #2: Shopping List::Design Document

Design Description:

The goal of this program will be to allow a user to create a shopping list. They will be able to modify (add/remove) items in the list, with name, cost, quantity, and unit type attributes for each item. The user will also be able to print the entire list and trip cost via the use of a menu in the main function. The program should use an overloaded operator to prevent the user from adding duplicate items with the same attributes. Refer to psuedocode for additional information:

Psuedocode:

Items.hpp/Items.cpp

Create Items class

Private:

Name
Quantity
Price
Unit

Public:

Items() // default constructor to set defaults
Setter()/Getter() // functions for all data members
printItem() // function to print an item
extendedPrice() // function to print the extended price

List.hpp/List.cpp

Create List class:

Private:

Items * item // dynamic array of pointers to Item class
Sizeofarray
Numofitems // size of items in list

Public:

List() // Default Constructor to initialize array and listsize
~List() // Deconstructor to deallocate memory after no longer needed.

Void addItem() // add item to the list
Void removeItem() // remove item from the list
Void Printlist() // Print the whole list as opposed to just one item. (I had originally intended on just using the print item function for this purpose, but

because of the added requirements (ie, Trip cost, I chose to implement this function.)

Main.cpp

Create a list object

Greet the user.

Initiate a menu (using switch)
Print the menu

Switch (choice):

1. Create a list // call the addItem Function
2. Add to list // call the addItem Function
3. Remove from list // call the removeItem Function
4. Print the list // Call the print function
5. Quit the program // end the loop

Reflection:

The initial design process for this assignment was fairly consistent with my final product. I have, however, highlighted the changes in design implementation following the final product. This program felt a lot like some of the projects we've worked on in the past and it was a relief when I found that this one came a little bit easier than previous assignments. One of the biggest issues I ran into was in overloading the operator, which I thought would come easily, but I simply ran out of time and have decided to turn in the program without it. I dare say I'm learning to code and refine my design process, although it is still somewhat difficult to implement an idea sometimes.

Test:

Initial Menu:

Input:	**Expected:**	**Actual Result:**
Initial Menu: (1-5)	Appropriate selection is made and user is returned to menu.	Appropriate selection is made and user is returned to menu.
Choose a number <1 or <5	User is asked to input a correct value	User is asked to input a correct value

Add Item:

Input:	**Expected:**	**Actual Result:**
Name: string	Added to list	Added to list
Name: String with a space	Added to list	Program does an infinite loop and fails
Cost: integer/double	Added to list with appropriate formatting	Added to list with appropriate formatting

Quantity: Integer/double	Added to list as integer	Added to list as integer
Unit type: 1-5	Added to list and correct unit is chosen	Added to list and correct unit is chosen
Unit type: >1 or <5	Default is chosen - case	Default is chosen - case

Remove Item:

Input:	**Expected:**	**Actual Result:**
User inputs associated number with item list	Item is removed	Item is removed