Homework 5

100 Points

Classes

Project A: Right Triangle Class
Project B: Inventory Class
Project C: Inventory Class Report

Grading

Create projects consisting of two or more files (source and header)

Program 5A – 15

Program 5B
- Create the Inventory class – 20
- Test the Inventory class – 10

Program 5C
- Read data from an input file (**inventory.txt**) into array – 10
- Output described in C.1 (out of stock items) – 5
- Output described in C.2 (cheaper items) – 5
- Input validation (prompt user to enter price) – 5
- Output described in C.3 (name match items) – 5
- Writes report to file (**inventoryReport.txt**) – 10

Create projects consisting of two or more files – 10
Self Assessment Report – 5

Run each program as required and save the output at the end of the source file as a comment. Compress the source and header files, input and output files (if any), and upload the compressed file: 22B_LastName_FirstName_H5.zip Note: please keep the two given folders (Program_5A and Program_5B_C) in your compressed file and nothing else.

Next Page

Project B: Inventory Class

Write a definition of a class that has the following properties.

a. The name of the class is **Inventory**.

b. The **Inventory** class has five member variables:
name – a `string` that holds the item's name
itemNo – an `int` that holds the item's unique identifier
quantity – an `int` for holding thw quantity of the items on hand
cost - a `double` for holding the wholesale per -unit cost of the item
totalCost – a `double` for holding the total inventory cost if the item (calculated as `quantity` times `cost`).

c. The class **Inventory** has the following member functions:
Default Constructor – Sets all member variables to 0
Overloaded Constructor – Accepts an item's number, cost, quantity, and name as arguments and copies them to the appropriate member variables and then calls the `setTotalCost` function
setItemName – Accepts a `string` argument that is copied to the `name` member variable.
setItemNo – Accepts an integer argument that is copied to the `itemNo` member variable.
setQuantity – Accepts an integer argument that is copied to the `quantity` member variable.
setCost – Accepts a `double` argument that is copied to the `cost` member variable.
setTotalCost – Calculates the total inventory cost for the item and stores it in `totalCost`.
getItemName – Returns the value in `name`.
getItemNo – Returns the value in `itemNo`.
getQuantity – Returns the value in `quantity`.
getCost – Returns the value in `cost`.
getTotalCost – Returns the value in `totalCost`.
printItem – Display the values in `name`, `itemNo`, `quantity`, `cost`, and `totalCost` with description as shown below:

ItemNo: 123244
Name: Slippers
Quantity: 25
Cost: $10.0
Total Cost: $250.00

d. Write definitions of all the member functions of class **Inventory**.

Next Page

Demonstrate the class in a driver program: once you have written the class, write a separate program that creates three **Inventory** objects. Input Validation: Do not accept negative values for item number, quantity, or cost.

For instance, create one object using the default constructor and display its member variables (they should all be zeros). Then create a second object using the overloaded constructor with data provided by the user and display its member variables. Finally, create a third object. This time use setters to copy data into the member variables and use getters to display its member variables. This way you have demonstrated and tested all member functions.

Project C: Inventory Class Report

This program will create an array of up to 100 **Inventory** objects and it will read data from an input file (**inventory.txt**) into this array. Then it will display on the screen the following:

1. (C.1) The names of the items that are out of stock (quantity 0)
2. (C.2) The names and costs of the available items with a cost less than or equal to a given cost. Prompt the user to enter a double (such as `8.99`), then show all items that are in stock and are cheaper than the user's input.
3. (C.3) The names and costs of the items with the same name as a given name. Prompt the user to enter a string (such as `Bag`), then show all items that are in stock and have the same name as the user's input.

   Optional: display all items that are either a perfect match or include the user's input. For instance both `Canvas Tote Bag` and `Bag` will be displayed.

Finally, it writes to another file (**inventoryReport.txt**) a table as shown below. At the end of the report the program writes the number of items and the store's total calculated by adding total costs for all items in the array.

```
Inventory Report
====== ==================== ======== ======== ===========
ItemNo          Name         Quantity   Cost    Total Cost
====== ==================== ======== ======== ===========
123244 Slippers                   25      10.00      250.00
610223 Belt                       15      10.00      150.00
====== ==================== ======== ======== ===========
Number of items: 2
Store Total: $400.00
```

Assume that a name has at most 20 characters. Design a solution to this project that consists of several small stand-alone functions (about 6). Each function should solve a specific part of the problem.

Next Page

On each line in the input file there are four items: item number, quantity, unit cost, and name. Test your program using the following data (input file: **inventory.txt**)

```
113244 50 10 Slippers
122339 0 5.99 Ceramic Pot
131299 25 10 Credit Card Holder
211298 25 8 Business Card Holder
221124 70 12.99 Slippers
230039 0 11.99 Belt
312378 43 9.99 Small Storage Basket
321123 97 4.25 Hand Soap
339234 33 10.99 Bag
411234 5 8.75 Scarf
422892 11 9.50 Makeup Brush Set
439123 0 9.99 Ceramic Pot
518934 20 8 Lip Balm
520223 2 10 Belt
532313 3 15.25 Slippers
610992 12 8.50 Cable Knit Beanie
621234 10 19.99 Bag
634411 0 9.75 Slippers
711099 0 6.25 Touch-Screen Gloves
722908 40 10.25 Canvas Tote Bag
812398 2 9.50 Pillow Cover
828824 0 7.25 Bag
911327 10 12.50 Slippers
921234 50 5.99 Bag
```

When the program prompts the user to enter a price, enter `8.99` or `9.99`. Input validation required. Assume that the price range of the items in this file is `0.25` to `99.99`. Therefore, if the user enters a value outside of this range or not a number, display an error message and ask for another input.

When the program prompts the user to enter a string, enter `Bag` or `Slippers`. Input validation is optional.