CSC 101 Programming Assignment #5 11/16/17

Due date – **Thurs Dec 14th @ 11:59pm**

**Program:**

Take another look at the **Item** class we created in lab 11. We will add some additional stuff to this.

* Start with the items created in lab 11 including three instance variables, an accessor and mutator/modifier for each, and two constructors.
* Add a class constant which is an integer and represents the minimum quantity all items should have in stock( Use the value 12).
* Add a Boolean method called needToOrder( ) which returns true if quantity of item is less than this minimum value.
* Add toString( ) which will output this way: “Item: Fork Qty: 12 Cost: $1.75”
* Add method called itemWorth( ) which returns current total price of the object (single object) based on the quantity on hand (qty \* price)
* Add method called sellSome(int) which reduces quantity of the object by the amount inputted.
* Add an instance method called greaterThan( Item ) which returns a boolean based on whether the itemWorth of the calling object is worth more than the passed in object.
* Add Javadoc comments to class – comments before each method

Now write a better driver program to test this class.

Create an Array of 10 Items (the Inventory). Do not use a premade class such as ArrayList. We will be adding and removing Items from this Inventory. Start with an empty array. We will assume that there will always be 10 or less. When we have less than 10 we will always use only the beginning of the array, the remaining values will be null. If we delete any items in the middle of the list, you must shift them in a way to keep all the items in the beginning.

For the main program you will create a text menu that comes up on the output screen giving the user a bunch of choices. As they pick a choice, that action will be done and then the menu will repeat to the screen again. A switch statement and/or having a link to additional static methods could be useful. Your menu will do the following:

1. list all items

-use toString on each filled array location

1. list the items that need to be ordered

-process through the array until reach null

-call needToOrder on each and show the true results

-Report if no items need to be ordered

1. add an item to the inventory

-add it at the first null location in the array

-ask the user to give the 3 inputs and then use the constructor to create one

-Report if there is not room to add another item

1. Report total inventory cost

-search through entire array, and sum up the itemWorths

-report the total.

1. Report item that is worth the most (Use greaterThan method)
2. Sell some of an item

-Ask user for description, find array element and ask user to say how much is sold and update quantity.

-Report if not found – should not crash

1. Remove an item from list

-Ask user for description, find array element

-Report if not found – should not crash

-set array location to null (make sure to shift array to fix holes)

1. Clear list. – set all array locations to null
2. End program

Test all of your menu choices, make sure to remove items in the middle of the list and check if the rest of the choices still work. I will test very extensively, so be prepared to do so also.

Each of these programs must be done in a separate file. Name the class **Item.java,** and the driver program **Inventory.java**. Name the class in each file these names as well. If you do not name these files correctly you will lose points.

Hand in electronically – (NOT E-mail!!!)

In S-drive CSC 101 folder:

1. Create folder called **projectfive\_firstname\_lastname**
2. Place two files named above in folder. (NOTHING ELSE)

Cases Status:

1. List all items: Completed
2. List the items that need to be ordered: Completed
3. Add an item to the inventory: Completed
4. Report total inventory cost: Completed
5. Report item that is worth the most: Completed
6. Selling some of an item: Completed.
7. Removing an item from the list: Completed
8. Clear list – set all items to null: Completed.
9. End program: Completed