CS1083 Assignment #1 – Fall 2021

Due: Wednesday, September 22nd before 11:00pm (Atlantic), submitted in the Assignment 1 dropbox in Desire2Learn. (Read the submission instructions at the end of this document carefully).

The purpose of this assignment is to:

- Review classes and objects and partially filled arrays.

Important habits to develop for all assignments:

- Start each assignment early so you have time to ask questions if / when you encounter problems (remember you should expect at least a 24-hour response time for email/chat messages).
- When working on the computer, save your work early & often.
- Include **Javadoc** for each program that you write (for drivers only your name and student number beside the @author tag in a Javadoc comment block at the top of the driver class is required). Run the Javadoc utility on your classes to ensure there are no errors in your comments (the documentation files created from running the Javadoc utility only need to be submitted with the assignment when specified): `javadoc -author -private FileName.java`
- Don't forget to make a back-up copy of your work somewhere other than on your computer (e.g.: on a USB key, external hard drive, or cloud drive).

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during a help session in the "Faculty of Computer Science Student Success Centre" team or to your course instructor.

Inventory System

You are working on part of a new product inventory system for the CSA (Computer Science Association) at UNB. The CSA has decided to sell a small number of products to help with fundraising. Your responsibility is to write two classes, one called Stock and one called Product. The Product class represents an item that is for sale in the CSA store. The Stock class keeps track of a number of products that are available for sale.

The Product class contains instance data for:

1. The product name;
2. A unique id for the product (which is automatically assigned starting at 10001 and cannot be changed);
3. The quantity of the product currently available for sale;
4. The quantity of the product currently ordered from the supplier; and,
5. The price of the product.

Note: the constructor is passed the name of the product, the price of the product, and the initial quantity of the product currently available for sale. Accessor methods are required for the id and the quantity of the product currently available for sale. Mutator methods are needed to set the quantity of the product currently ordered from the supplier and change the price of the product.

In addition, the Product class has the following methods:

1. orderReceived: updates the quantity of the product that is available for sale, and sets the value of product ordered to zero.
2. getTotalValue: returns a number value representing the total value of the product (i.e. the quantity available X the price).
3. sellProduct: returns true or false, representing whether or not the product can be sold; if true, the total number available is reduced by 1.

The maximum number of products that can be in the stored inventory is passed in to the constructor (via a parameter). The Stock class needs methods to accomplish the following (consider how you can make use of methods from the Product class to accomplish some of these tasks):

1. Add a product (given a Product object), returning true or false to indicate if the addition succeeded (make sure duplicate products are not added);
2. Sell a product (given a Product object), returns true or false to indicate if the object can be sold, and if it can be sold then the sale is completed;
3. Remove a product from the list of available products, returning true or false to indicate if the removal succeeded;
4. Determine whether or not more of any individual product should be ordered to replenish their stock; 10 of any given product should be ordered if there is less than 5 currently available in stock;
5. Calculate the total value of all items currently available for sale (i.e., add together the total value of all products).
6. Retrieve a textual list of all products, formatted in tab-separated columns as shown in the example on the next page including the product name, id, quantity available to sell, quantity ordered, and price:

```
Scissors  (id: 10001)    qty: 20   0 ordered      $5.25
Pencil    (id: 10002)    qty: 40   0 ordered      $0.25
Eraser    (id: 10003)    qty: 30   0 ordered      $0.65
Pen       (id: 10004)    qty: 4    10 ordered     $0.65
```

***Notes:**

- Dollar values can be formatted correctly from a double (as shown above) using the NumberFormat class, as shown in section 3.6 of your textbook.
- You must use arrays (and not ArrayLists) to store Product objects in the Stock classes.
- The order of the products in the inventory is not important (they do not need to be ordered by their id number).
- Products are identified by their id when searching through a list of Products.
- You may find it helpful to create a helper method in the Stock class to return the position of a product in the list (this is optional but preferred).

Driver program for testing:

Write a driver program to test the **`Stock`** and **`Product`** classes, covering the following:

- Create some products,
- Add some products to the stock,
- Try to add one too many products,
- Remove a product successfully,
- Try adding another product,
- Remove a product that doesn't exist,
- Sell a product,
- Try to sell a product that has no more items available,
- Order new products that are running low, and
- Display the total value of all products (properly formatted).

In your assignment you must describe each test case, and show the effects on the stock after each of the operations described above (this can be done by including print statements in your driver describing what is being tested followed by the result).

Submission instructions on next page.

Your electronic submission (submitted via Desire2Learn) will consist of two files:

1. a written report. This should begin with a title page that includes: the course (CS 1083), the assignment number, your full name, and your UNB student number. That should be followed by two sections, with each part clearly identified with a section heading. Include:
   a. all the source code for your solution (with Javadoc comments included),
   b. the sample output from running your application.

   This written report should be prepared using a word processor; we recommend using Microsoft Word (i.e. create a .docx file for your report). Copy & paste your java source code & required output into the report document. Add appropriate headings for each part. Fix up the formatting where necessary, adjusting line breaks & page breaks to ensure that your document is easy to read. Use a monospaced font for your code to maintain proper indentation.) Once the report is complete and you've checked it all over, save the .docx file for your own records, and then **save a second copy in pdf format for submission**. (Note: Be sure to open that file in a pdf viewer to verify that the pdf was generated correctly.) **The SINGLE pdf file containing your report will be submitted to the appropriate assignment drop box on Desire2Learn**. (It is important that you submit a pdf file and NOT the original Word document. This pdf will allow the marker to write comments directly on your work to give you better feedback.)

   Note: name this report as follows: **YourName_CS1083_As1_Report.pdf**

2. an archive file (**.zip**) that contains your Java source code for this assignment. Make sure that your archive includes all .java files (in case the marker wishes to compile & run your code to test it). You should not include the report document or the .class files in your archive. This archive should be submitted as a **single file** to the appropriate drop box on Desire2Learn.

   Note: name this archive file as follows: **YourName_CS1083_As1_Archive.zip**