**Program 1**

**CIS357, Summer 2022**

**Due 11:59 pm, July 5**

In this program, you are to write a Java Program that emulates cash register at a Grocery Store.

To make the problem simple, we assume that there are ten different items for sale and each item with its unique code 1 through 10. The following table shows the item code, item name and unit price:

|  |  |  |
| --- | --- | --- |
| **item code** | **item name** | **unit price** |
| 1 | bottled water | 1.50 |
| 2 | candy | 1.00 |
| 3 | chocolate | 2.50 |
| 4 | gum | 1.00 |
| 5 | soda | 2.50 |
| 6 | juice | 3.00 |
| 7 | popcorn | 2.50 |
| 8 | donut | 1.50 |
| 9 | pretzel | 2.00 |
| 10 | caramel | 1.50 |

Provide the item data in a file and read them into your program.

Minimally, your program must have at least two classes: Item, CashRegister. The Item class is for the item data with three attributes: itemCode, itemName, and unitPrice. The CashRegister class manages the sale. You are encouraged to include more classes. For example, you can have the Sale class that keeps the sale records.

Assume that a clerk in a grocery store uses your program to get the total of the sales items. Make the main class name as CashRegister.java. The following shows a sample run of the cash register system:

$ javac CashRegister.java  
$ java CashRegister  
**Welcome to MyLastName cash register system!**

**Beginning a new sale (Y/N)***y* ;both the upper and lower case should work **--------------------**  
**Enter product code:** *2* ;the user input stays right to the message  
         **item name: candy**  
**Enter quantity:**     *2*  
        **item total: $   2.00**;the output should be aligned properly  
**Enter Product code:***4*  
         **item name: gum  
Enter quantity:**     *1*  
        **item total: $   1.00  
  
Enter product code:** *-1*  
**----------------------------  
Items list:   
    2 candy         $   2.00  
    1 gum           $   1.00  
Subtotal            $   3.00  
Total with Tax (6%) $   3.18  
Tendered amount     $***5* ; the output should be properly aligned **Change              $   1.82** ; if less amount is entered, ask again  
----------------------------

**Beginning a new sale (Y/N)***Y* **--------------------**  
**Enter product code:** *3*  
         **item name: chocolate**  
**Enter quantity:**     *4*  
        **item total: $  10.00**  
**Enter Product code:** *8*  
         **item name: donut  
Enter quantity:**     *2*  
        **item total: $   3.00**

**Enter product code:** *3*  
         **item name: chocolate**  
**Enter quantity:**     *1*  
        **item total: $   2.50**

**Enter product code:** *1*  
         **item name: bottled water**  
**Enter quantity:**     *1*  
        **item total: $   1.50**

**Enter product code:** *11*  
**!!! Invalid product code**

**Enter product code:** *A* ; your program should not throw an exception  
**!!! Invalid product code**

**Enter Product code:** *8*  
         **item name: donut  
Enter quantity:**     *a*  
**!!! Invalid quantity**

**Enter quantity:**     *2*  
        **item total: $   3.00**

**Enter product code:** *-1*  
**----------------------------  
Items list:  
    1 bottled water $   1.50  
    5 chocolate     $  12.50** ; note that multiple items are listed once **4 donut       $   6.00** ; note that the listed items are sorted by the name **Subtotal            $  20.00  
Total with Tax (6%) $  21.20**   **Tendered amount     $***40* **Change              $  18.80**

**Beginning a new sale (Y/N)***N* **The total sale for the day is  $  23.00**

**Thanks for using POST system. Goodbye.**

In the sample run, the bold face letters are output from the cash register system and non-bold italicized letters are input from the user. The user enters ‘-1’ to terminate each current sale. The system is terminated if the user enters 'N' on **Beginning a new sale (Y/N)**.  Then, the POST system prints out the total sale amount of the day.

Your program should compile and run in the command line environment. All user input needs to be checked properly to avoid exceptions.

Suggestions: Provide methods (or functions) that perform the following tasks:

* Accepts the item code for each item for sale
* Get the item name and print the name
* Accepts the quantity of the item
* Computes the item total for each item
* Lists the quantity, name, and price for all sale items in sorted order, and get  the subtotal of all items purchased
* Computes the total sales amount with tax
* Initialize the item purchase list (for each new sale)

Use arrays for the item data. That is, you are not allowed to use other data types like ArrayList, Vector, Set, etc.

Provide two sample runs: one for your own that has three sales and one that is shown above – follow the exact same input sequence: if not, points will be deducted.

Provide screen images of the output: “hw1-out1.jpg” and “hw1-out2.jpg” for the two sample runs. Make sure that the screen image is easily readable with proper size. (You can also copy & paste the output images to a Word file).

**To get a full point**

* Properly document/comment your program. Your program header comment for CashRegister.java should be like

// Homework 1: Sales Register Program

// Course: CIS357

// Due date: xxx

// Name: Jon Doe

// GitHub: xxx

// Instructor: Il-Hyung Cho

// Program description: ...

/\*\* Javadoc comments for CashRegister.java \*/

public class CashRegister

{

...

}

...

* As noted, do not provide program comments with the Javadoc format. JavaDoc comments should be made for the description of classes, attributes, and methods. For methods, describe the return value and parameters passed, if any.
* Properly layout and indent your code with Javadoc comment format. For example,

public class Post

{

/\*\* member data \*/

...

/\*\* Constructor \*/

Post() {

...

}

/\*\* member function \*/

void blah() {

...;

}

}

* Format your code properly: make it sure that your code conforms to the standard coding style.
* Format your output properly. (You can use the printf function).
* Avoid exceptions due to wrong input type.
* Run Javadoc on your code
* Create a GitHub account and create a repository for this homework. Make the repository name to **cis357-hw1-lastName**. Do not expose your GitHub account to other students. Specify the URL to the repository in your program header comment.
* Zip your program (all Java files), the Javadoc generated files, the input file (that contains the item data), and the output files with two sample runs (hw1-out1.jpg and hw1-out2.jpg or hw1-out.docx and hw2-out2.docx) to as lastName-hw1.zip and submit to Canvas.

Note: It may be easy to process the input if the item data is provided like the following:

1

bottled water

1.50

2

candy

1.00

Or, it can be like the following. It is up to you.

1, bottled water, 1.50

2, candy, 1.00