Homework 4 - Trades

PROG 1403 - Java I

Real world example.

Summary

For this assignment we will use the Collections Framework to solve a problem. Consider the following scenario: A company / individual buys 100 shares of a stock at $12 per share, then another 100 of the same stock at $10 per share, and then sell 150 shares at $15. After the sale, taxes must be paid (capital gains tax). But what is the gain since two different prices were paid for the stock. Well, the IRS has a rule for this situation. It is an asset-management and valuation method in which the assets (in this case stocks) acquired first are sold first. So, we must sell all shares of the first batch for a sale amount of $300 ($3 x 100), then 50 of the shares from the second batch, for a sale amount of $250 ($5 * 50), yielding a total sale amount of $550. This assumes all the stocks are from the same company. We made 550 but we must pay tax at a rate of 15%. So the tax is $82.50 and the take-home is $467.50.

We will write code to solve this problem. The best solution is one built on classes, but for now we will use the Collection Framework.

For our solution, we will make these calculations for arbitrary purchases and sales of shares of multiple companies

Specifications

1. Create a new project and add a class Stocks.
    a. Present the user with a menu of three options:
    b. Buy (B or b)
    c. Sale (S or s)
    d. Print (P or p)
    e. Exit (E or e)
2. Do not assume the user will enter valid data.
    a. This means invalid entry for the menu, prices, stock quantity.
    b. Note – When selling, you cannot sell more stock than you have! Keep that in mind.
3. If the user selects Buy, then prompt the user for the three-letter symbol, the quantity, and the price.
4. If the user selects Sale, then prompt the user for the three-letter symbol, if the symbol exists in those bought, then prompt for the quantity sold and price. Make sure you do not sell more than the number of stocks available.
    a. Display the number of units sold at each purchase price and the total amount for the sale, the tax, and the total take-home. Use the example above as a test case. Your output should be formatted to two decimal places an aligned in a tabular format. Sample output given below.

5. If the user selects Print, then display the name of the stock along with all number of shares and the purchase price. Remember, you may have multiple copies of the same share.
6. If the user selects Exit, then close the application.
7. You may use any number / combinations of collections you wish to use. Remember you can use a collection within a collection. For example, an ArrayList of Maps, or Maps which contain Lists, etc…
8. Use user-defined methods for user input, validation, calculations, etc…
9. Do not forget the JavaDoc comments.

Sample output

```
// Assuming the user input and sold the stocks listed above.

// ### is the stock being sold.

Selling stock (###)

Selling 100 shares at      $15.00 ($3.00 profit) :          $  300.00

Selling 50 shares at       $15.00 ($5.00 profit) :          $  250.00

Total sale amount                                           $  550.00

Tax (15%)                                                   $   82.50

Take Home                                                   $  467.50

50 Share of ### remaining.
```

Documentation

A text document (.docx, .rtf, .pdf) which contains the following:

- Your name and assignment.
    - A screenshot of your code output for each of the menu options. Make sure you demonstrate your validation is working as expected.
- You also need to explain in detail the following:
    - What data structures did you use? Why?
    - What benefits do you feel you gained using the data structure? What drawbacks?
    - Do you feel you choose correctly? Why or Why not?
- Remember to be specific in your responses.

What to Submit

You need to submit your document and your .java file. DO NOT Zip your files.