Project

canvas.cityu.edu.hk/courses/46242/pages/project

The course project is to implement a simple stock trader. The detail description can be accessed by clicking here ↓ . This pdf file includes the description of requirements only. In the page below, we provide some design suggestions.

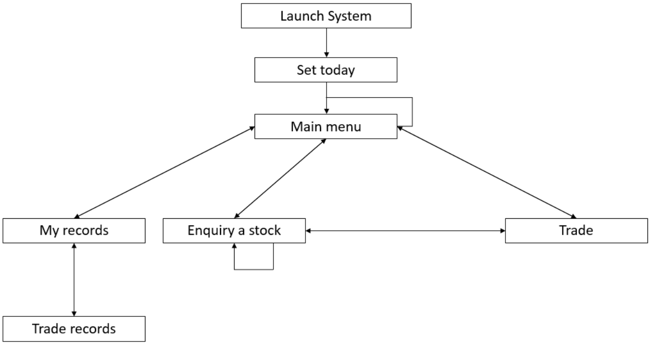
The project application requires to read the following csv files. Click them to download
stock-meta.csv ↓
performance.csv ↓
daily-price.csv ↓
The content of these files are just examples, your submission may be tested with files of different content (same file names). You may make some changes to the files for testing, but the changes should not violate the format definition. Be careful that do not modify by excel although they are csv files. More strictly speaking, do not even open the csv files by excel, open them by notepad++ or any simple text editor.

The file below provides the methods for all printings to screen as well as the Scanner object for reading keyboard input.
Screen.java ↓
The Screen.java file is assumed to be in the same folder as the application. Your submission should not contain any Java statements for printing and any objects for reading keyboard input. All the printings should be done by calling the methods provided by the following Screen.java file and all keyboard input should be read by the Scanner object, keyboard, provided in Screen.java file. You should not modify the Screen.java file and your submission should not include the Screen.java file.

The project must be completed with only the materials that are introduced in class. In particular, only those you can see in lecture slides, examples and lab pdf are allowed to be used. Something like "..." or "see Java documentation for more" are not allowed, they are told that Java provides some more, but their usage are not introduced in class.

Submission:
Your submission should be a zip file that contains all your necessary .java file, but not include Screen.java (even though you include it, I will not use it, but use my one instead). The Java class with the main method that to be automatically called when launching the program must be named **StockTrader**. The submission deadline is on Friday 22 Apr, 2022 23:59. Please be reminded that you should **download your submitted zip file from Canvas and extract it** under the lab environment (can be remoted via VM, see lab 01) as a double check to make sure the file is uploaded successfully and correctly.

Menu:
The menu flow is one of the challenge of this project. You may refer to the following flow chart.



You are suggested to create a class for controlling the flow of menus. The concept of inheritance may help (introduce in the coming week).

Sorting:
We will introduce a better sorting method with recursion in the lab part of recursion topic. But if you would like to complete it before that, you may use a simple sorting algorithm (but not good in performance). The simple sorting is described below:
1) You want to sort an array of n element, say list[0], list[1], ..., list[n-1] in ascending order (in project, we require descending).
2) Find the largest element from list[0] to list[n-1] (See lecture-05 page 16), say list[i] is the largest element.
3) Swap list[i] with list[n-1]. The largest element is now at the end of the array.
4) Repeat (1-4) but with considering the array has only n-1 elements, from list[0] to list[n-2].
5) At the end, largest element at list[n-1], second largest element at list[n-2], ..., and the smallest element at list[0], and thus sorted in ascending order.

Date comparsion:
Note that, the date format is DD/MM/YYYY. A simple string comparison cannot compare which one is larger, e.g. use string comparison, you got "03/06/2021" is smaller than "28/05/2021". You may need some conversion to the date in order to successfully compare two dates.