publicis
sapient

Publicis Sapient Hiring Exercise – Senior Associate

Name: _____
Date: _____
Start Time: _____
City: _____
End Time: _____

Terms and Conditions
Prior to taking the coding exercise for assessment purposes, kindly read the terms and conditions stated below.

1. This document is meant to assess your technical skills and is classified as "confidential". This document by any means shall not be used, copied, pasted, circulated, published, modified or disclosed without prior written permission of Sapient Consulting Private Limited ("Sapient").
2. Your submitted coding solution will be thoroughly checked. In case, it is found that the code solution proposed by you is copied by you from any source(s) including but not limited to the internet or any kind of assistance is sought prior to providing the completed exercise, Sapient shall reserve the right to blacklist your candidature for current and future employment opportunities with the organization.
3. In case you have seen this assessment exercise previously, please immediately inform your point of contact in Sapient and request for a different exercise for fair evaluation.
4. You are not entitled to misrepresent your identity or conceal any material fact which may have a bearing on your employment prospects or employment with Sapient.
5. In the event of non-compliance with the requirements in clause numbers 1, 2, 3 & 4 above, Sapient shall reserve the right, in its sole discretion, to stop processing your candidature and further Sapient will reserve the right to revoke the offer of employment extended to you and/or accepted by you and/or terminate your employment with Sapient, as the case may be, without incurring any liability to you.
6. By signing below, you confirm your acceptance to the terms and conditions stated above.

Accepted by:
Complete name:
Signature:
Date:

publicis
sapient

Instructions

- Duration of this exercise is **135 minutes**. Please manage your time accordingly.
- Make any necessary assumption, and clearly state the assumptions made.
- Do not seek any help – online or through any other source

Expectations

- The code should run and solve the requirement.
- Write the Unit Test Cases using Junit or TestNG.
- You may optionally list down the test cases in text or excel file in case you are not comfortable with any unit testing framework like Junit or TestNG.
- Sample Input and Sample Output files are attached in Section 3 of this document. Please seek help from hiring team in case the files are missing or any clarification.

Evaluation criteria

- Code Completeness / Correctness
- Code Structure and quality:
    - Modularity
    - Usage of OO Design Principles and Design Patterns
    - Size of classes/functions
    - Naming convention; Class/function/variable names
    - package/class structure
    - Choice of data structures
- Unit Test cases.

The better you perform on these criteria; higher you will score.

Submission guidelines

- Please follow **Section 4** for project structure and packaging.
- Please follow **Section 5** for submission guidelines. You may demand extra time for this step.

publicis
sapient

1. Problem Statement

Sapient has won the contract to implement processing fee calculator for a major investment bank (henceforth referred as client). Client receives transactions from various external sources.

These transactions are received in a pre-configured format, for example, CSV, EXCEL, XML or a simple pipe delimited format text file placed at a file location.

The client needs to calculate the processing fees for the transaction and generate a report which can be sent for invoicing.

1.1. Objective

- Read the transactions into the system. The various transaction attributes are listed below.
- Execute the processing rules (mentioned below) over input transactions
- Provide API to get the summary report in a particular format (format mentioned below)

  Note: The code should handle CSV format input however the design should be extensible to support other input formats as well in future.

1.2. Transaction Attributes

Attribute Name	Attribute Description
External Transaction Id	Unique Identifier
Client Id	Unique Id for each client
Security Id	Unique Id for the security for example RELIND for reliance industries
Transaction Type	Buy, Sell, Deposit, Withdraw
Transaction Date	Date in MM/dd/yyyy
Market Value	The current market value of this transaction
Priority Flag	Value Y,N

2. Processing Rules

2.1 Intraday transactions

Intra-day transactions are the ones where security (like IBM Equity) is bought & sold on the same day.

- Intra-day transactions will have two transactions having same Client Id, Security Id, & Transaction Date but opposite Transaction Type i.e. one transaction would be 'Sell' '& other would be 'Buy'.
- Each 'intra-day transaction should be charged **$10** for both the Buy & Sell legs.

2.2 Normal transactions

publicis
sapient

A nominal fee is charged to process each transaction. Fee calculation is based on following rules:

- **$500** for a transaction with **high priority** (denoted by the priority field in the transaction)
- **$100** for a transaction with **normal priority** and Transaction Type is **Sell** and **Withdraw**
- **$50** for a transaction with **normal priority** and Transaction Type Code is **Buy** and **Deposit**

3. Summary Report Definition

The summary report should contain the processing fee to be charged from each client, for a particular transaction type, transaction date and priority.

- The processing fee should be grouped by client Id, transaction type, Transaction date, & priority flag.
- Report should further be sorted by the Client Id, Transaction Type, Transaction Date and Priority.
-

Client Id	Transaction Type	Transaction Date	Priority	Processing Fee

Sample input Data is attached in the below CSV file.

Input data file.csv Sample_Output.csv

3.1 Saving attached sample input/output files

1. Double click on the attached file, e.g. **Input data file.csv**
   It will open the file in Excel in a new window.

2. Click on **File** -> **Save As** -> **This PC** -> **Desktop** (any desired location)
   It will open **Save As** dialog for saving the file.

3. Change **Save as type** to **CSV (MS-DOS) (*.csv)**, in **Save As** dialog.

4. Change **File Name** to Sample_Input.csv, in **Save As** dialog.

5. **Click** on **Save** button to save the file.

6. After saving the file, in case these is a dialog saying:
   "Some features in your workbook might be lost if you save it as CSV (MS-DOS).
   Do you want to keep using that format?"
   Click **Yes** on the dialog.

publicis
sapient

7. Follow the same steps to save **Sample_Output.csv**.

4. Project Setup

- Create Eclipse/IntelliJ project with name **<SapeFeeCalc<Your First Name>>**
  e.g. – SapeFeeCalcJoe

- Usage of Spring or any other framework is optional.

- Please prefer to create Maven/Gradle project, in case you are using any third party libraries (jar files) or using any framework like Spring.

- If you are not comfortable with maven configuration, place the required jars in lib folder and include jar from lib folder into build path.

5. Project Submission

- We expect a complete Eclipse/IntelliJ project. It is not recommended to submit java files without any project structure.

- Copy project e.g. SapeFeeCalcJoe from IDE (Eclipse/IntelliJ) workspace to some other location e.g. C:\Users\Public\SapeFeeCalcJoe

- Navigate to copied project e.g.: C:\users\Public\SapeFeeCalcJoe

- Delete these folders, if any of these folders exist. These folders are very heavy and exponentially increase the size of ZIP file.
  .git, .svn, .metadata, RemoteSystemTempFiles

- Delete everything inside **bin**, **target** or **build** folders, if any of those folders exist.
  These files are not required and gets automatically generated when the project is imported into any IDE (Eclipse/IntelliJ).

- Zip the folder e.g. C:\Users\Public\SapeFeeCalcJoe and save the zip with same name, e.g.: SapePerCapitaJoe.zip

- In case you are unable to zip the third part jar files, create a file named **dependencies.txt** and mention the names of all of the dependencies, place **dependencies.txt** in the project folder.
- Send the zip file, e.g.: SapeFeeCalcJoe.zip