Test Assignments

Design principles

Assignment should fulfill below concepts:

1. Should be simple to understand
2. Doable in 4-8 hours by senior engineer
3. Covers important software development topics such as unit tests, API design, object-oriented design
4. Similar to what we are doing for the project
5. Use technologies that are approved in the Bank (based on https://axess.sc.net/scb-axess-cms-client/pages/catalogue)

Backend engineer

Data Reconciliation Engine

Your assignment is to write a simple reconciliation program. The program must read sets of records from two files (called X and Y further in the document), match these records and report about discrepancies (also called breaks).

Records in both files have identical structure consisting of the following properties:

- Transaction ID – string that uniquely identifies a transaction record
- Account ID – string identifier of the account to which the transaction is posted
- Posting Date – date when the transaction was posted in the time zone of the processing system
- Amount – amount of money posted, represented by number with 2 digit precision

Pair of records (x, y) is considered a match if the following criteria are met:

- Account ID is identical
- Posting Date absolute difference is no more than 1 working day (excluding weekends). For example, difference between date falling on Friday and the one falling on the following Saturday, Sunday or Monday is 1 day
- Amount absolute difference is no more than 1 “cent” (0.01)

If all of the three properties listed above are identical between records in a match, then such match qualifies as exact, otherwise the match is called weak. Records between two sets (xs, ys) are expected to match one to one. Records are ordered. If a record from xs matches more than one record from ys, the first match is selected. A record from any side that matches no record from the other side must be reported as break.

Acceptance Criteria

Your solution will be evaluated against the following acceptance criteria.

1. Must be written in Java 8 or higher or kotlin programming language
2. Must compile without errors and use only JDK classes, no external libraries are allowed except libraries for unit testing such as jUnit, Mockito, or PowerMock
3. Must take input from two files containing sets of input records and produce a report containing all matches and breaks. See example of input data sets and reconciliation report below.
4. Should have source code that is clean and covered by unit tests
5. Should have execution time of O(n) with regards to number of records in input files
6. Should have design that supports future extension of record structure and matching rules
7. Should have REST API and act as a web service
8. Should be containerized

"Must" requirements represent a minimum acceptable solution; failing to meet any of them will render the submission as a fail. "Should" items represent advanced requirements and will be used to rank your solution. Your aim is to meet as many of them as possible within the allowed time limit. In cases when certain requirement in this document is not clear to you or open to different interpretations, you should make reasonable assumptions, document them and provide the solution based on those assumptions.

Submission Requirements

Your solution should be submitted as a single zip/jar file containing only items listed below. The package should NOT contain any other files such as class files, IDE specific files, external libraries, etc.

- Java sources of the program available as GIT repository (**do not** put your code on Github/Gitlab/Bitbucket)
- README.txt file that describes how to run a program and any assumptions made

Data Samples

X set
x0; 000000001; 01-Jun-2016; 100.00
x1; 000000001; 01-Jun-2016; 100.00
x2; 000000002; 01-Jun-2016; 100.00
x3; 000000002; 01-Jun-2016; 100.00
x4; 000000003; 01-Jun-2016; 100.00
x5; 000000003; 02-Jun-2016; 100.00
x6; 000000004; 03-Jun-2016; 100.00
x7; 000000004; 06-Jun-2016; 100.00
Y set
y0; 000000001; 01-Jun-2016; 100.00

y1; 000000001; 01-Jun-2016; 100.01
y2; 000000002; 02-Jun-2016; 100.00
y3; 000000002; 02-Jun-2016; 100.01
y4; 000000003; 02-Jun-2016; 101.00
y5; 000000003; 04-Jun-2016; 100.00
y6; 000000004; 06-Jun-20s16; 100.01
y7; 000000005; 06-Jun-2016; 100.00
Report
XY exact matches
x0y0
XY weak matches
x1y1, x2y2, x3y3, x6y6
X breaks
x4, x5, x7
Y breaks
y4, y5, y7