CS1003 – Programming with Data

W08 Exercise

The aim of this week's exercise is getting familiar with stream programming in Java. We will use a large CSV dataset as the data source.

Data

We will use a (large) CSV dataset for this exercise: the Price Paid Data (PPD) which is released under the Open Government Licence v3.0.

Information about the dataset:
https://www.gov.uk/government/statistical-data-sets/price-paid-data-downloads

Please use the CSV version of either one of the following datasets:

(1) Yearly file for 2020 (23.2MB)
(2) Single file from 1 January 1995 to the most current monthly data (3.7GB)

You might want to start working with (1) and test your program with (2) once you are happy with its behaviour. Working with fairly large datasets like these will highlight some of the benefits of a stream-based computation.

There are 16 columns in these CSV files, but they do not come with a header line. Here is a list of the columns.

- ID
- Price
- Date
- Postcode
- Property type
- New build
- Estate type
- PAON Primary Addressable Object Name. Typically the house number or name.
- SAON Secondary Addressable Object Name. If there is a sub-building, for example, the building is divided into flats, there will be a SAON.
- Street
- Locality
- Town/City
- District
- County
- Ignore
- Ignore

Suggested tasks

We suggest a few operations to implement on this dataset. Feel free to get creative!

- Find the cheapest property within the first 1000 entries. Remember, since streams are executed lazily this means you won't even have to parse the entire dataset. So a correct implementation will run very quickly using either data file as the source.
- Find the average property price in a given area (choose one of town/city/district/county)
- List the top 10 areas with the highest/lowest average property price
- Add filtering/grouping by year of transaction to the above

Parsing CSVs

For all of these tags you will be required to parse the CSV file. Use a stream to read the lines of text from the file. For each line, you will have to split the data into columns. Be careful though, the columns themselves are likely to contain commas as well. So just splitting by the comma character will not be enough. Consider using the Apache Commons CSV library for parsing the file.

The JAR file is provided in the exercise directory. See file W08Exercise.java as a starting point.