CSE 3302 Fall 2017

Lab 3b

This lab has one problem as described below.

For this problem you must perform the following:

1. Develop only the java Functional (Lambda Expression) program solution - this MUST use lambda expressions and .stream() operations ONLY.
2. You are to develop and get running in Eclipse.
   1. Code.
      1. Main program.
         1. You need to create the problem as a main program - the code is in the body of the program. You may create methods within this main class that does the computations called from the main method - please do not put methods in other classes.
         2. Name this class RealEstateMainFunctional.java
      2. Data declarations
         1. You will read the file *House Data.csv* which has the following elements in it. This data is read from the file and stored in the class RealEstateClass.java
            1. String Property\_Type
            2. String Address
            3. String City
            4. int Zip
            5. int Price
            6. double Beds
            7. double Baths
            8. String Location
            9. int Square\_Feet
            10. int Lot\_Size
            11. int Year\_Built
            12. int Days\_On\_Market
            13. int Dollar\_Per\_Sq\_Ft
            14. int HOA\_Per\_Month
            15. int Rating (see below)
         2. You will read the file *ZipRateTable.csv* which has the following elements in it. This data is read from the file and stored in the class ZipRateTable.java
            1. int ZipCode
            2. int Rating
      3. Algorithm
         1. Output.

File name. The output is written to 2 files: *HouseAveragesByZip.txt* and *HouseOrderByRating.txt*

* 1. *HouseOrderByRating.txt*

1. Header - the file writes the following header (names exactly as shown - tab delimited):
2. Type
3. Address
4. City
5. Zip
6. Price
7. Beds
8. Baths
9. Location
10. Sqft
11. Lot size
12. YrBlt
13. DOM
14. $/SqFt
15. HOA/mth
16. Rank Grp (see processing below)
17. Percnt SqFt (see processing below)
18. Data - each item is tab delimited
19. All the data above is written in the same format as in the input file *House Data.csv* with the following exceptions:
    1. Price, $/SqFt, HOA/mth - $, thousand separators, no decimal place
    2. Rank Grp - single digit integer, no decimal place
    3. Beds, Baths - double - single digit, single decimal place e.g, x.y
    4. Sqft, Lot size - integer, thousands separators, no decimal place
    5. Percnt SqFt - double, range is 0.0 to 200.0 inclusive, single decimal place
    6. YrBlt - 4 digit integer, no decimal place
    7. DOM - integer, no decimal place
    8. Rank Grp, Percnt SqFt - data - see processing requirements above.
    9. *HouseAveragesByZip.txt*
20. Header - the file writes the following header (names exactly as shown - tab delimited):
21. Zip Code
22. No. Homes
23. Ave Price
24. Ave Sqft
25. Ave Beds
26. Ave Baths
27. Ave $/sqft
28. Ave DOM
29. Ave YrBlt
30. Ave HOA
31. Data - each item is tab delimited
32. Zip Code is written in the same format as the input field Zip
33. No. Homes - single digit integer, no decimal place
34. Ave Price - $, thousand separators, no decimal place
35. Ave Sqft - integer, thousands separators, no decimal place
36. Ave Beds, Ave Baths, Ave DOM - double, single decimal place e.g, x.y or xx.y but - no leading zeroes
37. Ave $/sqft, Ave HOA - $, thousand separators, two decimal places
38. Ave YrBlt - integer, no decimal place
39. No. Homes, Ave Price, Ave Sqft, Ave Beds, Ave Baths, Ave $/sqft, Ave DOM, Ave YrBlt, Ave HOA - each value represents the average of that zip code.
    * + 1. Processing
40. *HouseOrderByRating.txt*
41. RealEstateClass.Rating (Rank Grp in the output) is set as follows:



1. When the RealEstateClass.Rating is between 1 and 4 inclusive and the corresponding ZipCode from the ZipRateTable.class has a rating >6 then those houses are printed by
   1. rating order (1 to 4)
   2. then by lowest Price first for each in the same rating
   3. then by the lowest Dollar\_Per\_Sq\_Ft in the same rating
2. Percnt SqFt is determined by 100\* Dollar\_Per\_Sq\_Ft/ average Dollar\_Per\_Sq\_Ft for that zip code from *House Data.csv*.
   * + 1. Code constructs
3. Operations.

You are ONLY allowed to use Lambda expressions and Stream operations

You are not allowed to use Collections.sort or basic for/while loops (other than to read the file data)

1. Data Structures. You may use whatever collection you so desire. TreeSet is a collection that implicitly has the operations .sort and .disctinct applied to its members.
2. Exception handling - surround necessary code with try and catch blocks or throw exceptions. Your code must NOT crash when properly run but you may assume the input and output files are there and correct.
   * + 1. Processing notes
          1. Notice that the data is not organized by county or city - your program must address that. You are not allowed to manipulate the input file.
          2. Your code must not hard-code county or city names - it will need to build Streams from the data file as needed.
       2. Verification of output
          1. How do you know what is correct? View this as a work assignment where you are given a large database and have to produce a report that can be run via your program. There is no solution to compare it with. You need to analyze the input data and determine if your output is correct.
       3. Example files (these are not intended to be complete)
          1. House EXAMPLE.csv is provided with two outputs:
          2. HouseOrderByRating EXAMPLE.txt and
          3. HouseAveragesByZip EXAMPLE.txt

Submission checklist

1. Please make sure to pay attention to any submittal instructions the GTA may have given you.
2. All materials should be delivered in a single .zip (not .rar) file and named lastname\_firstname\_ID Four (4) files total as follows.
3. Java files - you should have 3 java files:
   1. RealEstateClass.java
   2. RealEstateMainFunctional.java
   3. ZipRateTable.java
4. You should have 4 \*.txt (files)
   1. House Data.csv
   2. ZipRateTable.csv
   3. HouseOrderByRating.txt
   4. HouseAveragesByZip.txt

Uniqueness of solution. Your submittal needs to be distinct from what everyone else submits. GTAs will review each submittal - exact appearances in submittals will result in an investigation - if the investigation determines that two submittals are identical both submittals will receive a zero for the entire assignment.