Sihan Tu | Lei Wang (Team 11)

CIT 594 Group Project

December 9, 2019

Project Report

1. Additional Feature

When the user enters 6, the ratios of total residential market value per capita and total fines per capita for all areas are displayed on the screen in ascending numerical order of the zip codes.

We use market_value from the properties.csv, total fines from parking file, and total population from population.txt. This ratio indicates whether there is a relation between market_value per capita and fines per capita.

To ensure the feature is working correctly, data processors created for feature 2 and 5, where error-proofs are implemented, are called to calculate the ratios, instead of creating new processors. One additional check against fine amounts is implemented to avoid the situation when the fine is smaller than or equal to 0, the situation of which is meaningless in real life. Lastly, we verified the results returned from sample dataset via Excel.

2. Use of Data Structures

Data Structure	**Applications**	**Alternative Data Structure**	**Reason**
HashMap<String, LinkedList>	Property reader and Parking reader	LinkedList/ ArrayList	The data structure takes the same O(n) time to restore data as the other two list structures. But we can access data in O(1) time later by zip codes. Alternatively, if we use lists to store data, we will need to use memoization later to recalculate the data by zip codes, which costs O(n) time.
TreeMap	To store calculated total fines per capita for each zip code in Processor.getTotalFinesPerCapita	HashMap	Keys or values are not sorted in HashMap, while TreeMap is ordered by keys. When presenting calculated total fines per capita to the user, we would like to print the results based on the zip codes (i.e. keys), it's natural to use TreeMap to avoid extra sorting.

Data Structure	Applications	Alternative Data Structure	Reason
LinkedList	To store Parking/ Property objects in corresponding HashMaps	ArrayList	Insertion is most frequently used when reading from input data for Parking and Property objects. ArrayList takes O(n) to add data in worst case as it requires resizing, while LinkedList takes O(1) to add each piece of data, thus the latter is more efficient comparing with the former.
StringBuilder	PropertyTextReader class	String	Since String is immutable, normal String manipulations will produce a new String object. StringBuilder has memory space and speed advantages without recreating String objects.