README

Justifications:

MyList:
MyList uses array to implement a basic ArrayList data structure with get(), add(), isEmpty(), and size() methods. This data structure is used for implementing MyMultimap and search 6. This is an easy-to-implement basic data structure.

MyMap:
MyMap uses an array of map entry of key-value pairs to implement a basic Map data structure with get(), put(), and size() methods. This data structure is used for implementing MyMultimap.

MyMultimap:
MyList and MyMap is utilised for implementation of MyMultimap. MyMulti MyMultimap Map data structure has entries with key-List<value> pattern, that is one key can have a list of values. For the first search in this assignment, we can use dialler number as key and a list of all call records called by this dialler number as list of value. Similarly, search 2 would take receiver number as key and a list of all call records received by this receiver number as list of value.

For search 1, getting all call records called by a specific dialler number will take O(n) time in the worst case, witch is all call records are called by different dialler number. But in reality, usually a dialler number has multiple call records, so the actual run time might be less than O(n). Similarly, search 2 shares the same runtime efficiency.

Multimap data structure was chosen for performing search 1 and search 2 because it is suitable for the scenario of one dialler/receiver number might have multiple call records. AVL Tree was also considered at the first time, it has better performance in terms of searching. But AVL Tree is much more complicate to implements and inserting new item might result re-structuring in order to keep AVL Tree balance.

MyBSTree:
MyBSTree is a binary search tree data structure used for storing Switch ID and count of each switch id connections. The binary search tree data structure makes searching very efficient, normally it take O(log n) time to search an item in the tree. This definitely would be beneficial for validating a switch identifier when reading call records from file. Besides, it is also very efficient for search 4 and 5 since switch id and the count associating with this id are both stored in MyBSTree nodes.

Reference:

Binary Search Tree Complete Implementation in JAVA. (2017). Algorithms. Retrieved 15 October 2017, from http://algorithms.tutorialhorizon.com/binary-search-tree-complete-implementation/

Goodrich, M., Tamassia, R., & Goldwasser, M. (2015). Data structures and algorithms in Java (6th ed., pp. 449-450). New York: John Wiley.

Lars Vogel. (2016, 09 29). How to implement an ArrayList structure in Java - Tutorial. Retrieved from Vogella: http://www.vogella.com/tutorials/JavaDatastructureList/article.html

Lars Vogel. (2016, 09 26). How to implement common datastructures (List, Stack, Map) in plain Java - Tutorial. Retrieved from Vogella: http://www.vogella.com/tutorials/JavaDatastructures/article.html#map