Brian Klein, Thomas Winters, and Joe Denzer

CS 341

Dictionary Report

We implemented our Dictionary Interface using an ArrayList, HashMap, and TreeMap. The implementation of these data structures varied in difficulty for this assignment.

TREEMAP-

We had to learn about Tree Maps in order to use them, but once we learned how to use it, it worked fairly well for this assignment. This is the second-best structure we used for efficiency because it uses O(log n) for searching the dictionary.

ARRAYLIST-

The Array List took a little bit more work because we had to create an Entry Class, but it made up for this because it is an ADT that we have used in the past and know well. This is the least efficient structure we used for this assignment, using O(n) for searching the dictionary.

HASHMAP-

Hash Map was slightly confusing to implement at first because we had some difficulty changing the JAVA documentation, which utilized <K, V> (<entry, value>) to match our dictionary interface of taking in a complete String and then splitting it up into entry, value. In the end, the way we did it was easier to understand than the Java API because it was more familiar to us. This is the best structure we used because it used O(1) or constant time for searching the dictionary.

OVERALL-

Having learned more about each of these data structures during this assignment, we conclude that a HashMap is the most efficient structure and considering that we now know how to use it, it should be the go to ADT next time we are asked to do a similar project. TreeMap is second most efficient and very similar in implementation to HashMap therefore would be the second choice of ADT for an assignment of this nature. The ArrayList is very easy to understand, but it takes a little longer to implement because of the necessary use of a separate Entry class. That, along with the fact that it is the least efficient ADT using BigO, means that ArrayList is the ADT we are least likely to use for an assignment like this in the future.