Assignment 4, Question 2(e) Solutions

Mark Eramian

Question 2(e) Answers

1. Numbers may vary very slightly, but not by much, and certainly not by enough to affect the trend in growth. Significant variation is an indication that either the calculations were done incorrectly, or obtainWithCount() was not implemented using the correct algorithm (e.g. checking all items via iteration over the whole hash table, rather than hashing the correct list and then iterating over only that list).

Filename	Avg. Queries for `hashQuestLog`	Avg. Queries for `treeQuestLog`
quests4.csv	1.25	2.0
quests16.csv	1.125	4.0
quests250.csv	1.232	13.34
quests1000.csv	2.148	18.08
quests100000.csv	1.45	30.8778

2. 1, constant function, $O(1)$ are all acceptable answers.

3. $\log n$, logarithmic function, $O(\log n)$ are all acceptable answers. Base of the logarithm doesn't matter (since $\log_a n$ is $O(\log_b n)$ for any a and $b > 1$).

4. If the primary use is to display all the elements in alphabetical order, then they tree-based quest log would be preferred. You would simply have to do an in-order traversal to obtain the `QuestLogEntry` objects in the desired order, examining at most n such objects where n is the number of quests in the log.

   In the case of the hash-table-based quest log, this operation is much more costly. first one would have to use `keys()` to get all of the keys, which would examine n `questLogEntry` objects, with a cost of $O(n)$, then the resulting array of keys would have to be sorted, (which costs at least $O(n \log n)$), then the quest log entries have to be re-queried in order by key from the hash table, which incurs additional $O(n)$ cost. Even if an advanced-linear time sort is used (which is possible, you'll learn this sort in March), the hash table version would still be about 3 times more costly than the tree version (three $O(n)$ operations vs. one).

5. If the primary purpose is random access to retrieve details of quests, then the hash table version would be preferred, because, as the experiments show, one should only have to examine one or two `QuestLogEntry` objects on average. Whereas with the tree-based log, the cost increases with the size of the log. Another way to say this is that looking up a specific quest in the hash-table costs $O(1)$, but looking up a specific quest in the tree-based table costs $O(\log n)$. Thus, the hash-table is preferred for this purpose.