**PROJECT REPORT**

**Problem Statement:**

Comparison of Lists, Balanced Trees and Hashing.Consider implementations of dictionaries with just the operations Insert, Find, and, Delete (sometimes called add, contains, and, remove).

Compare the performances of Lists, Trees, and, Hashing for these operations. Use data structures in Java Library (Linked/Array lists,Tree Set/Map, Hash Set/Map). Experiment with millions of elements.

**Results:**

Following table shows the performance of HashSet is fast compared with performance of TreeSet. Linked List is taken very long time

**Part 1:**

Size: 10000000

Elements are sorted

|  |  |  |
| --- | --- | --- |
| **HashSet** | **TreeSet** | **LinkedList** |
| 14900 | 29800 | More than an hour |

Note: All the reading are calculated after taking average of 5 outputs.

**Conclusion:** Time taken to execute insert, delete and contains operation on millions of dataset using Hash Set is less when compared with Tree Set. While to perform basic operation on Linked List takes very long time. I had to wait for an hour but still didn’t have the result. As to delete any element from Linked List we need to traverse up to that element and then delete it. So in worst case the time complexity is O(N).

**References:**

Class notes - Balaji

Introduction of Algorithms - Thomas Cormen, Charles Leiserson, Ronald Rivest , Clifford Stein

http://www.java2s.com/Tutorial/Java/0140\_\_Collections/Quicksortwithmedianofthreepartitioning.htm

http://www.java-tips.org/java-se-tips/java.lang/quick-sort-implementation-with-median-of-three-partitioning-and-cutoff-for-small-a.html