Your report goes in this file. Remove this description and replace it with your report. The report consists of two parts:

1. Two tables showing speed comparison between polymorphic tree and Javas’ TreeMap. Use TreeSpeed.java for information on how to obtain time information. Each table should have two columns: data size (number of values used) and the time (in milliseconds). Each table should have at least five entries. The first table will show results for trees created with numbers in a sequence and the second table with trees created with random numbers.

SEQUENCE

|  |  |  |
| --- | --- | --- |
| Data size | Time \_ BST | Time\_TreeMap |
| 500 | 3 | 2 |
| 1000 | 9 | 2 |
| 2000 | 20 | 1 |
| 5000 | 143 | 5 |
| 8000 | 377 | 6 |

RANDOM NUMBERS

|  |  |  |
| --- | --- | --- |
| Data size | Time \_ BST | Time\_TreeMap |
| 500 | 209 | 2 |
| 1000 | 8 | 4 |
| 2000 | 7 | 4 |
| 5000 | 15 | 5 |
| 8000 | 12 | 9 |

1. Two or three lines explaining the table results.

Results show that as data size increases, time of making sequence using PolymorphicBST is taking a lot longer than using TreeMap. This is because as BST make sequence trees in which the key only increases, it only inserts a new tree to the right side, and thus uses the most inefficient way to make a tree (in a linear line with the longest possible height, making it n complexity). On the other hand, TreeMap balances the tree and maintains the lowest height, making the next inserting the most efficient (thus maintaining the log n complexity).