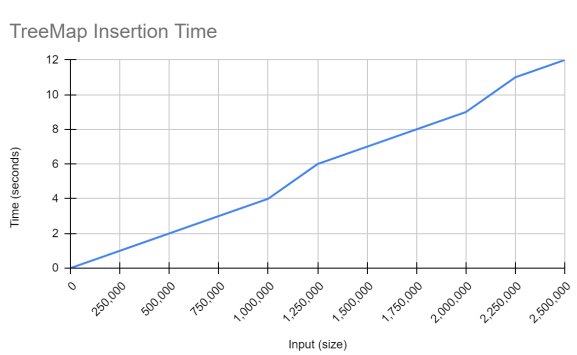
COS285 Data Structures
Alex Bard & Devin Chamberlain
Assignment 6

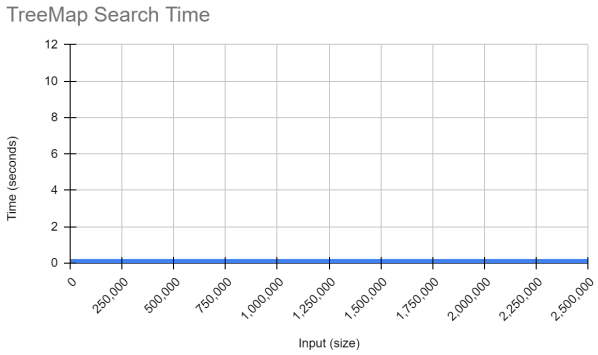
Link to GitHub Repository:
https://github.com/AlexBard122/Assignmnet5

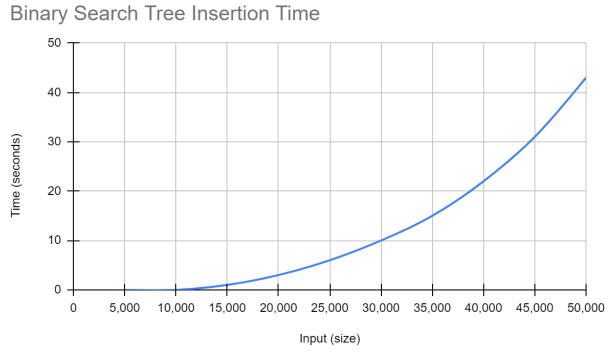
Task 1:
Complexity order is O((log n)(log n)) or O((log n)^2) because we are creating a treemap which is O(log n) within another treemap which results in (log n)*(log n) to get O((log n)(log n)).

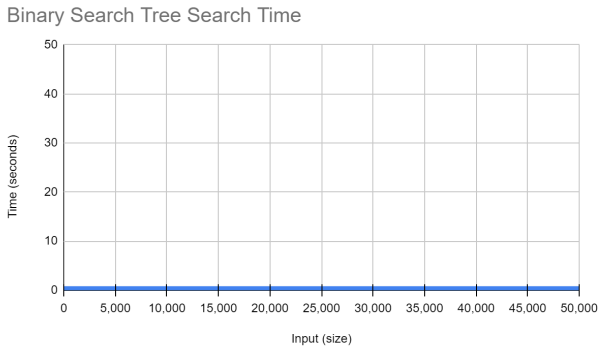
Task 2:
Complexity order is O(1) because the size() method which is used to find the number of reports in a given state on and after a given date is O(1).

Diagrams:









Discussion:

After comparing the effectiveness of Binary Search Trees (BST)s and TreeMaps, it is clear which one is the superior option. When looking at the graphs for each data structure's insertion time, you can see that TreeMap's insertion time increases linearly

as the input size increases. Meanwhile, the BST's insertion time increases exponentially as the input size increases. Furthermore, the BST is only able to reach an input size of 50k before the program ceases to be effective due to its execution time. The TreeMap's insertion time is able to reach an input size of 2.5 million in less than a third of the time it takes for BST to reach just 50k, showing how much more efficient TreeMaps are over BSTs for storing information.

When comparing the graphs for search time between BSTs and TreeMaps, it is clear that both methods return very similar results. For both methods, the search time is the same regardless of input size.

These graphs show that although the search times are identical for each tree regardless of input size, the insertion time for TreeMaps is superior to BSTs. This means that TreeMaps are better than BSTs, especially for large inputs.