Frederick Javalera

When you are satisfied that your program is correct, write a brief analysis document. The analysis document is 30% of your Assignment 3 grade. Ensure that your analysis document addresses the following.

1. Who is your programming partner? Which of you submitted the source code of your program?

Thuy Nguyen. Thuy submitted it.

2. How often did you and your programming partner switch roles? Would you have preferred to switch less/more often? Why or why not?

We switched about 50/50. I think this assignment was about the right ratio.

3. Evaluate your programming partner. Do you plan to work with this person again?

I feel the same as last week. The end product turned out really well and I am happy with it so I'll deal with the negative aspects of working with this partner. I'm not sure if I'd work with this partner again. If it were something I were already familiar with and not getting tripped up on then yes. Otherwise, I'm not sure.

4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development time.)

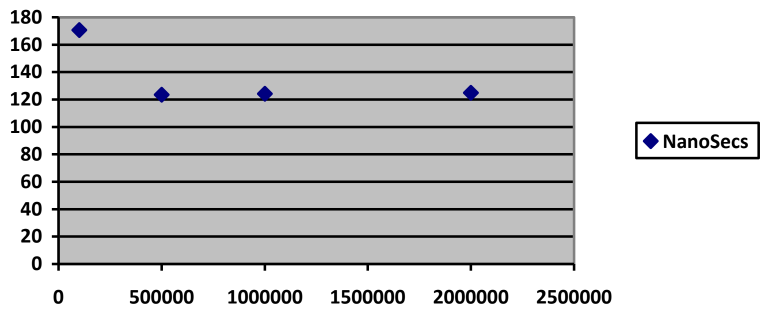
I think a java list would have been much more efficient in terms of programming development time due to the fact that java lists already possess many of the of the methods that we implemented or could be implemented rather quickly. In terms of running time I don't think a java list would be much more efficient since every element must be accessed sequentially, whereas with a regular array the can be accessed randomly via index.

5. What do you expect the Big-O behavior of MySortedSet's contains method to be and why?

Since the contains method implements a binary search method I would expect the Big-O behavior to be logN since binary search uses the halving principle to find the element.

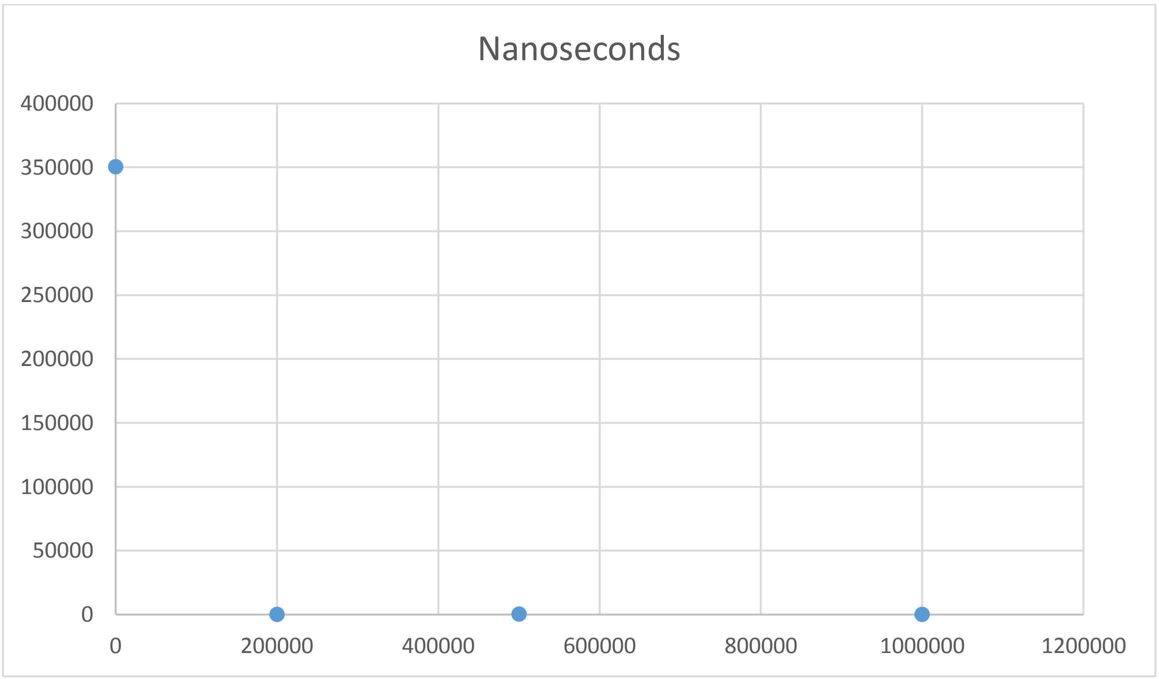
6. Plot the running time of MySortedSet's contains method for sets of sizes 100000 to 2000000 by steps of 100000. Use the timing techniques demonstrated in Lab 1. Be sure to choose a large enough value of

timesToLoop to get a reasonable average of running times. Include your plot in your analysis document. Does the growth rate of these running times match the Big-oh behavior you predicted in question 5?



It does not appear to run in logn time since the value for n = 100000 is higher than the other values and logn time would have it be lower.

7. Consider your add method. For an element not already contained in the set, how long does it take to locate the correct position at which to insert the element? Create a plot of running times. Pay close attention to the problem size for which you are collecting running times. Beware that if you simply add N items, the size of the sorted set is always changing. A good strategy is to fill a sorted set with N items and time how long it takes to add one additional item. To do this repeatedly (i.e., timesToLoop), remove the item and add it again, being careful not to include the time required to call remove() in your total. In the worst-case, how much time does it take to locate the position to add an element (give your answer using Big-oh)?



It takes O(N) to locate the position to add an element.

8. How many hours did you spend on this assignment?

10-12

Programming partners are encouraged to collaborate on the answers to these questions. However, each partner must write and submit his/her own solutions.

Upload your document (.pdf only!) to the Assignment 3 page by 11:59pm on February 5.