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.

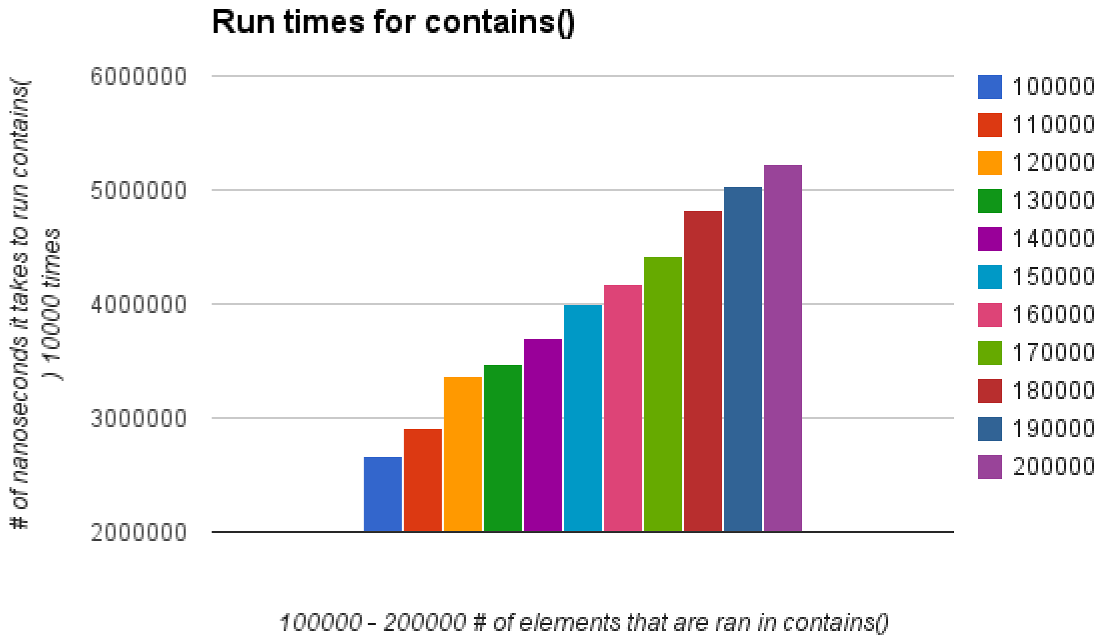
1. Who is your programming partner? Which of you submitted the source code of your program?
Ryan Fletcher…..me
2. How often did you and your programming partner switch roles? Would you have preferred to switch less/more often? Why or why not?
We switched a couple times. I'm fine with either way. It didn't matter to me.

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

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.)
There would have been no need to double the array size or set each corresponding array item to a different value if something was removed or added to the array.
Not having a for loop to loop through and change the array each time an item was added or removed would have been much more efficient. I believe these were the most time consuming lines in the add and remove method calls. And just being able to call something that had already been implemented instead of inventing the wheel again would have saved a few hours.

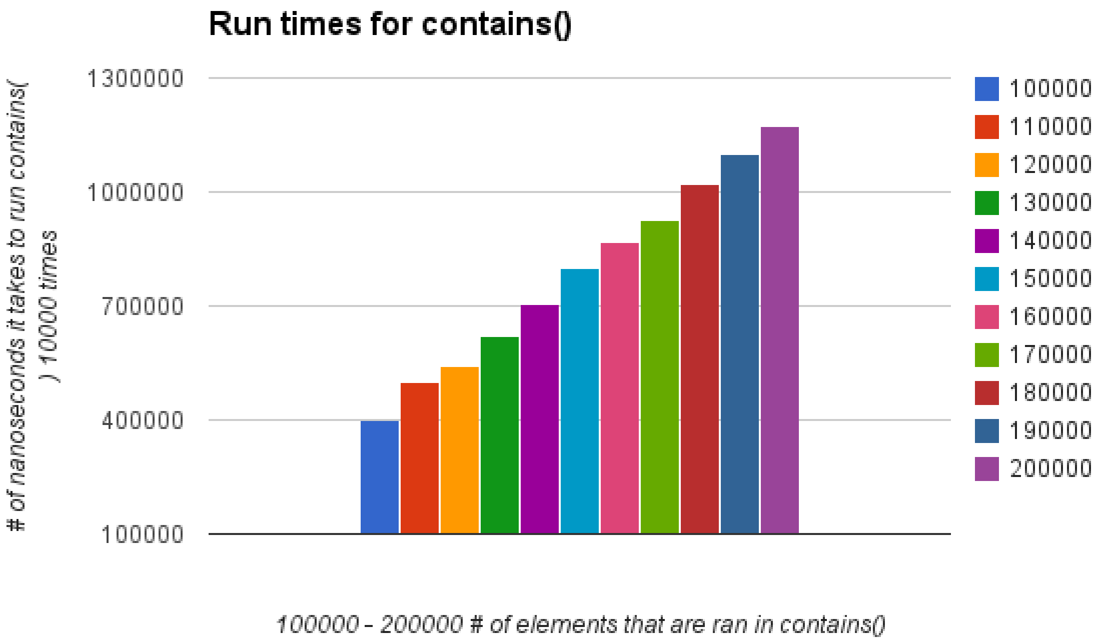
5. What do you expect the Big-O behavior of MySortedSet's contains method to be and why?
I believe the contains method had logN behavior because it implemented a binary search method would is a proven logN Big-O behavior

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 appears to be more of N Big-O behavior



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)?

Worst case would be adding an item to the very front because every item would have to be shift up by one. I believe this function would be Nlog(N) because it has to find where it needs to go and then move every item after it by one to insert, and possibly having to double a new array.



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

20 hours

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.