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?

Gradey Cullins. He submitted the code.

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 roles when we completed a new step in the assignment. I would not prefer to switch roles more or less often. The way we switch roles allows us to both get a good amount of time in each role, while also being able to complete each method and test without changing the style of programming halfway through.

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

I do plan to work with my programming partner again. He knows what he is doing and our skills are very complimentary in that his strongest skills are my weakest skills and vise versa.

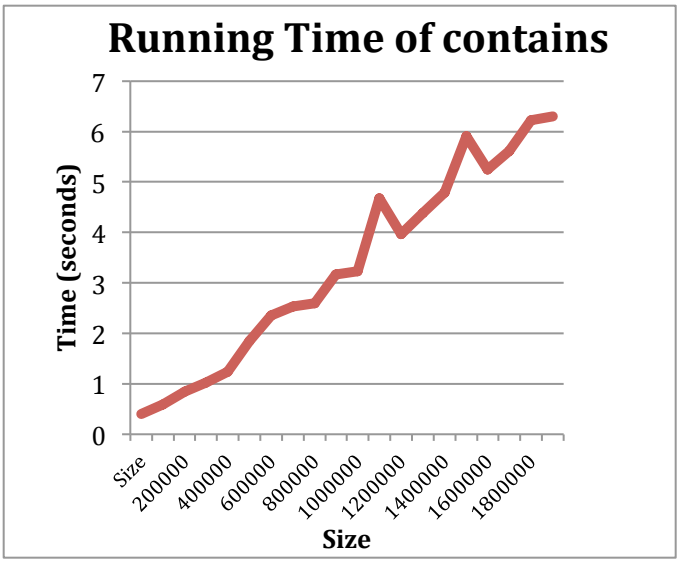
4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development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

The Java List does not sort new elements as they are added, so any search, contains, or remove method would be less efficient in running time, or a sort method would need to be added in during program development that would find the desired position in the list. However, many of the methods in the sorted set would already be defined by Java and thus would not be coded in our implementation.

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

Log N. The program runs using a binary search, and thus will be dividing by two at each step. This division creates an order Log N method.

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?



Yes it does.

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)?

In the worst-case it takes O(Log N).

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

About 7 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.