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?

Yiliang Shi - she did.

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

~10. I like how it is right now. we seem to be getting in a decent groove.

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

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.)

development time would be much shorter, run time would be about the same to shorter, and the code would be much shorter

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

Log(N)- binary search

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?

The contains method was fairly constant and grew very slowly. Much like a logrithmic curve.

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)?

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

~8 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.

Data Plot:

It takes exactly 2029408.0 nanoseconds to add an object to 200000 many existing elements
It takes exactly 4158.0 nanoseconds to search for an object out of 200000 many elements
It takes exactly 3776999.0 nanoseconds to add an object to 300000 many existing elements
It takes exactly 8386.0 nanoseconds to search for an object out of 300000 many elements
It takes exactly 5983477.0 nanoseconds to add an object to 400000 many existing elements
It takes exactly 6120.0 nanoseconds to search for an object out of 400000 many elements
It takes exactly 6125611.0 nanoseconds to add an object to 500000 many existing elements
It takes exactly 6724.0 nanoseconds to search for an object out of 500000 many elements
It takes exactly 7904320.0 nanoseconds to add an object to 600000 many existing elements
It takes exactly 6861.0 nanoseconds to search for an object out of 600000 many elements
It takes exactly 9645848.0 nanoseconds to add an object to 700000 many existing elements
It takes exactly 7356.0 nanoseconds to search for an object out of 700000 many elements
It takes exactly 1.0801844E7 nanoseconds to add an object to 800000 many existing elements
It takes exactly 7554.0 nanoseconds to search for an object out of 800000 many elements
It takes exactly 1.1965884E7 nanoseconds to add an object to 900000 many existing elements
It takes exactly 7786.0 nanoseconds to search for an object out of 900000 many elements
It takes exactly 1.6740603E7 nanoseconds to add an object to 1000000 many existing elements
It takes exactly 8407.0 nanoseconds to search for an object out of 1000000 many elements
It takes exactly 1.7034203E7 nanoseconds to add an object to 1100000 many existing elements
It takes exactly 10605.0 nanoseconds to search for an object out of 1100000 many elements