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?

Darold Benjamin Green. 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?

Every two methods or so, I believe.

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

He was great. Again, I can say he did about 50% of the work. I plan to work with him on the next assignment.

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.)

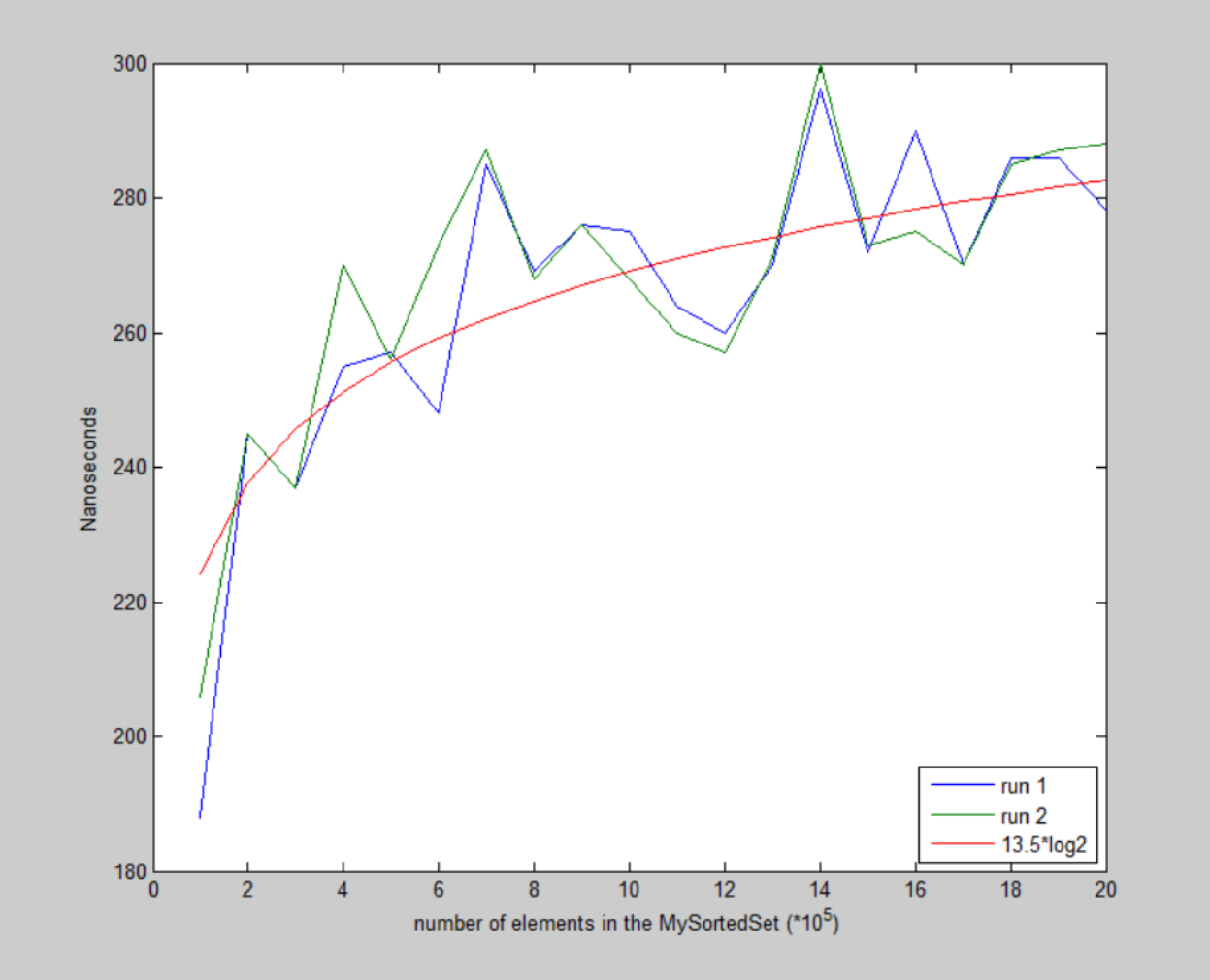
We would have had to implement a few index related methods (remove at index, add at index) and it would also have a few obsolete methods (add at index wouldn't have been good, the array would then be unsorted)

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

Log(n) because it's consistently dividing by two.

6. Plot the running time of MySortedSet's contains method for sets of sizes 100,000 to 2,000,000 by steps of 100,000. 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. Plotted in red with the plots is log base 2 * 13.5. This fits nearly perfect as a general describing function.



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)?

Up to Log2(n) + up to n. Worst case, it takes exactly log base 2 of n times to locate the position.

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

I'm going to go with 6.

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.