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

My programming partner is Isabelle Chalhoub. I submitted the source 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 every hour or so. I am at the keyboard a lot, so I should let her type more often.

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

Yes, I plan to work with her again. I was sick this week, and I left my partner hanging, but we were able to scrounge something up. She is still doing very well.

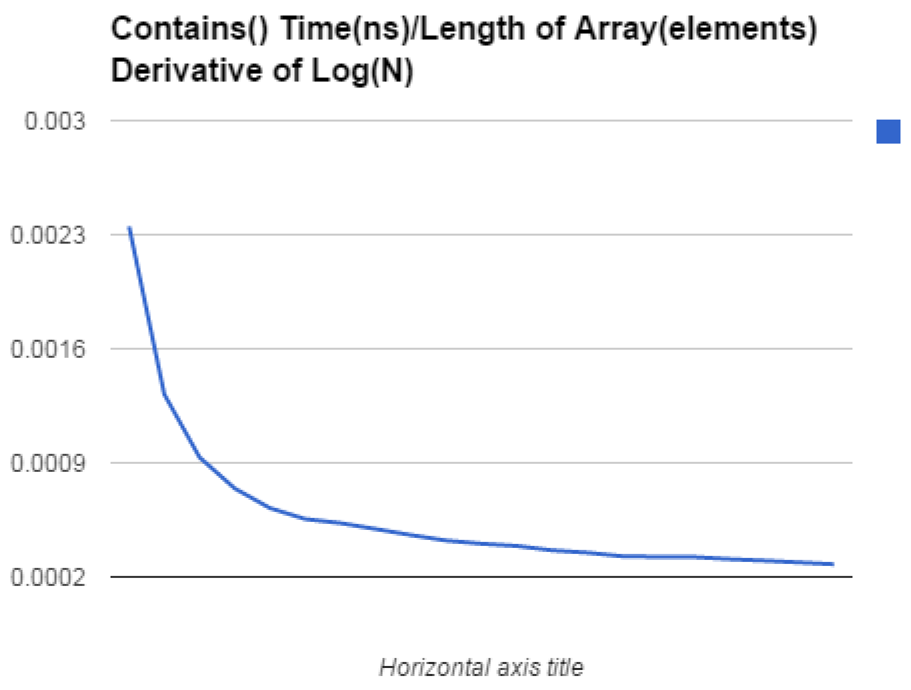
4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development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

Lists and arrays have very similar performance, simply because we are sorting and adding to both of them. I think in this case, arrays are faster because of how many calls to specific indexes we have to do.

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

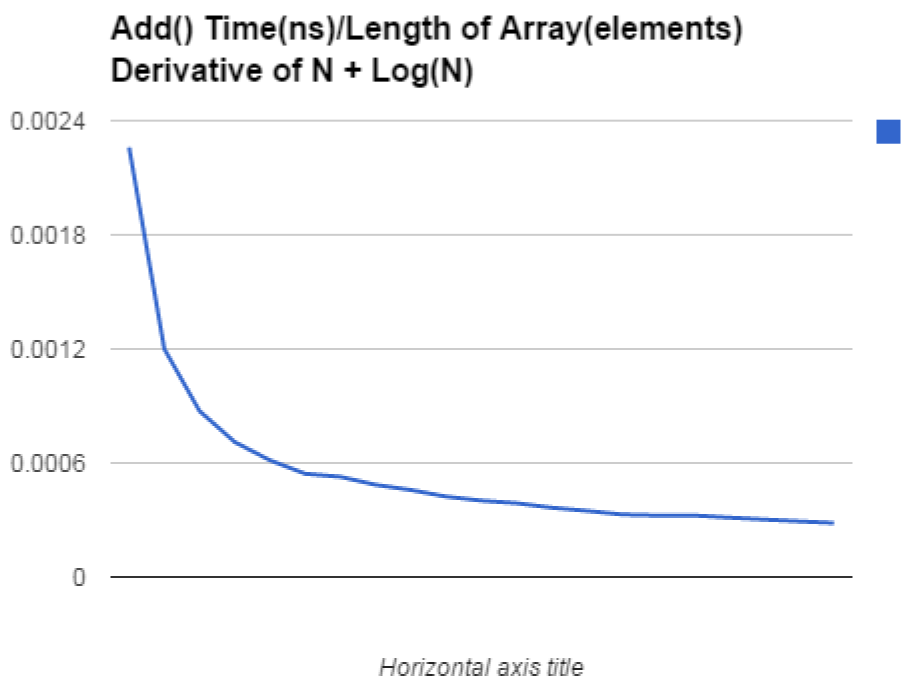
Log(n), simply because it is a binary search function.

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 seems to work that way. The derivative of the running time is t/n, so the integral of the running time is tLog(n), which makes sense from a big o perspective.

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 seems to take something like (N-1)/2 + log(n). That makes sense, because it requires that the program copy the average part of the array and shift it forward and add on the time it takes to do a binary search.

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

We spent about 24 hours on this assignment. This was incredibly hard.