Richard Campbell
U0718855
CS 2420

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

My programming partner was Nguyen Truong and she 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 roles each day we met, except for the last day we switched between both roles a little bit. I think I like switching a little bit, because I think it helped make things go faster.

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

Nguyen is a good programming partner to have. We are able to work together well to complete the assignments in a timely manner. We plan on working on the next assignment together.

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.)

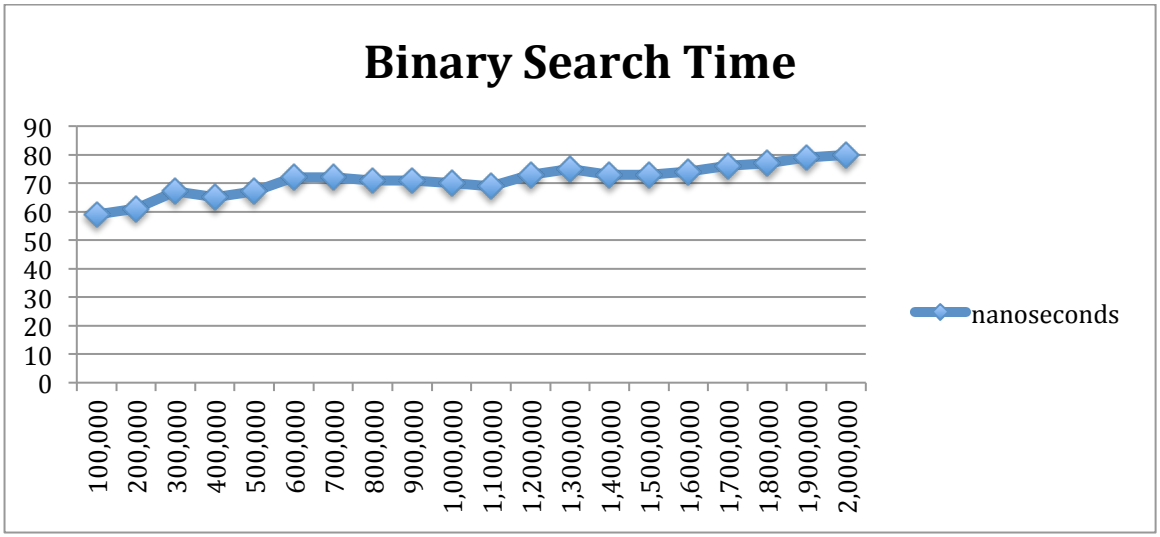
I think it would have been more efficient in both running time and program development time. In order to be included as a standard package in java it would have to be super efficient, and arguably the most efficient implementation. A lot of time was spent working with the array, and having one that already works would have saved time.

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

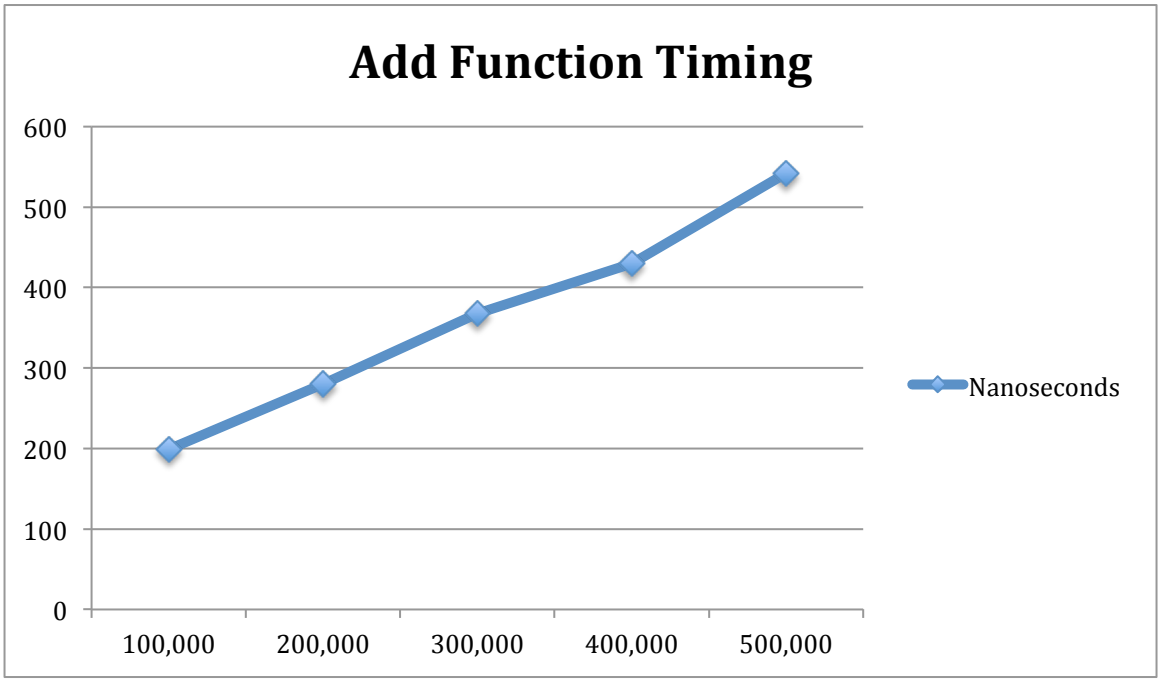
O(log N) because it halves the set each time while finding it.

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?

timesToLoop value was set to 10,000 in order to get a reasonable average. The growth rate seems to be fairly consistent with O(log N). Graph is included below.



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 is O(log N) to locate the position an element would be added because we use just a regular binary search to find the

position. Now that it's located it has to add it to the array and shift places accordingly. Depending on where in the array it is, depends on how many things need to be shifted in the actual adding to array part of the add function. This could be as large as N if it needs to shift every item. timesToLoop was 1000 and the graph shows the entire add function which is somewhere between O(N log N) and linear.

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

We spent roughly 9-10 hours working on this assignment.