1. Who is your programming partner? Which of you submitted the source code of your program?
John Chambers
John Chambers 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 about 5 or 6 times.

I thought it was fine how much we switched because when one was getting stuck then the other would jump in and help get things going again.

3. Evaluate your programming partner. Do you plan to work with this person again?
My Programming partner is pretty good at seeing code and trying to make it better. He thinks a lot on the code that is written.

I do plan on working with him 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.)

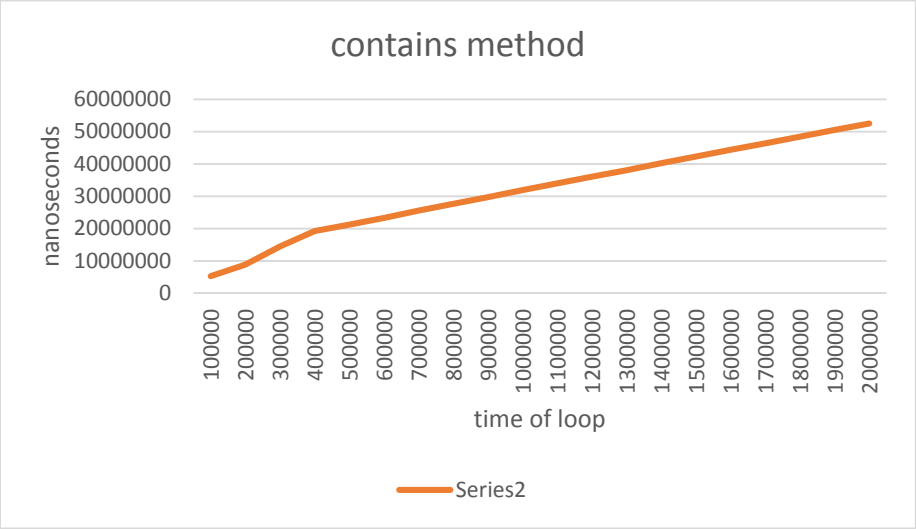
It probably would have differed how we searched parts of it out. Also it would have been differed the way we add and removed objects in the List or Set.

I think it would have been more efficient to do it with a Java List, because in some steps it would have been quicker to get to the different parts in the list the way they are set up.

5. What do you expect the Big-O behavior of MySortedSet's contains method to be and why?
log N because in some cases it would be quick but would get longer as the set its bigger because it needs to compare more items.

NEXT PAGE……

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?

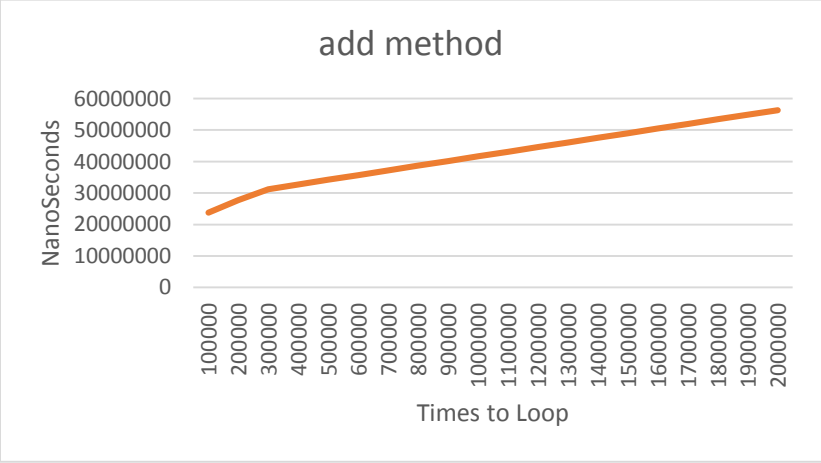


It matches a little because it takes longer as it gets farther.

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)?

For an element that's not in there it takes as long is it does to check every element.

Linear O(N)



8. How many hours did you spend on this assignment?
About 25 hrs