Ensure that your analysis document addresses the following.
1. Who is your programming partner? Which of you submitted the source code of your program?
Tyler Centini, I, Stefan Kapetanovic, will submit the code for program.

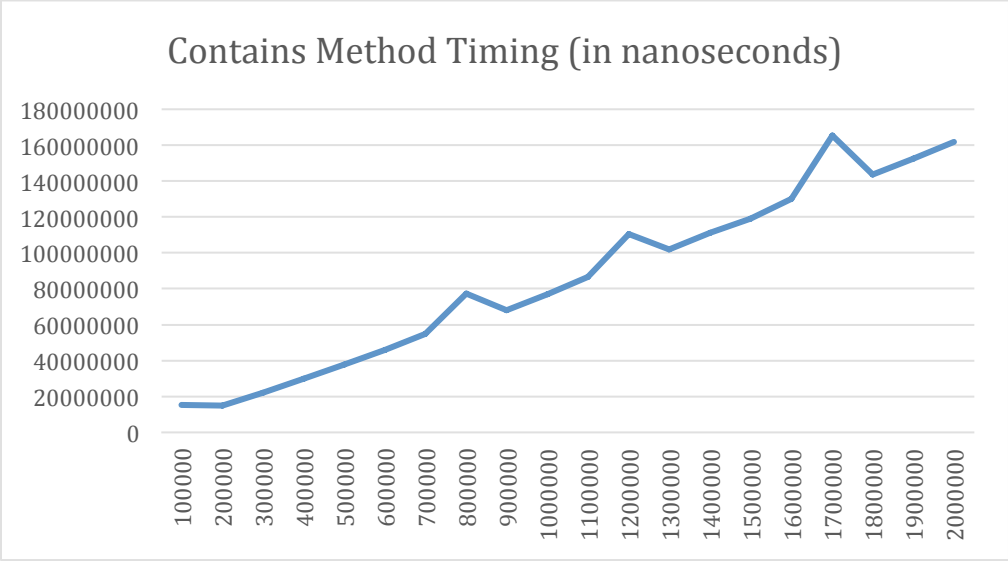
2. How often did you and your programming partner switch roles? Would you have preferred to switch less/more often? Why or why not?
We would switch roles every time an individual had a better understanding of what concept we were working on. I would like to code just a tad more but understand that Tyler also needs the experience coding.

3. Evaluate your programming partner. Do you plan to work with this person again?
Tyler is awesome. We work great together and manage our time very efficiently. I definitely plan on working with him again.

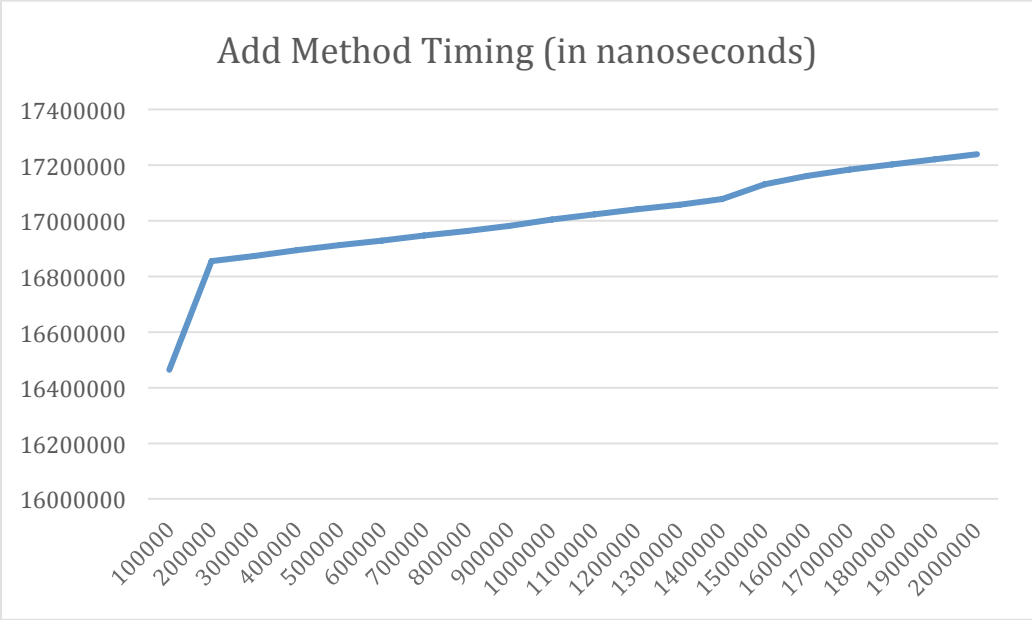
4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development time.)
A Java List includes numerous methods that would not need to be written unlike a basic array that we created. Some methods a Java List naturally includes are isEmpty, RemoveAll, Remove, etc. which we would not have had to implement like with our basic array.
A Java List would have been much more efficient due to the less implementation but the run time would be approximately the same.

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

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 our graph matches our prediction in question 5. The timer returned a N(logN) graph which is exactly what we predicted.



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?
We spent approximately 24 hours on this assignment.