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

Andy Ford is my partner
Andy Ford will submit our 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?

like 30 minutes
I prefer less switch, I think it will be more efficient.

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

He is good at the logic and very calm when we fixing the little problems, yes I think so.

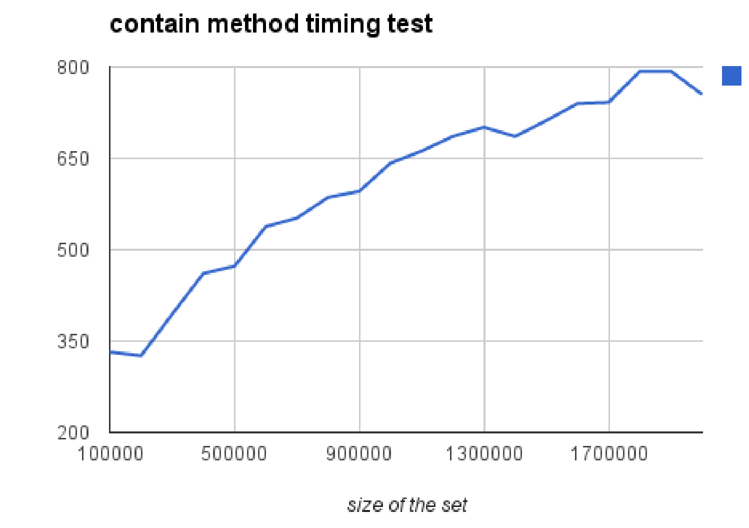
4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development time.)
In development time, I think it should be more efficient, because we don’t need to double the size of the list when it full and don’t need to implements some other methods because they are already there; and about the running time, I think a Java list probably more efficient.

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

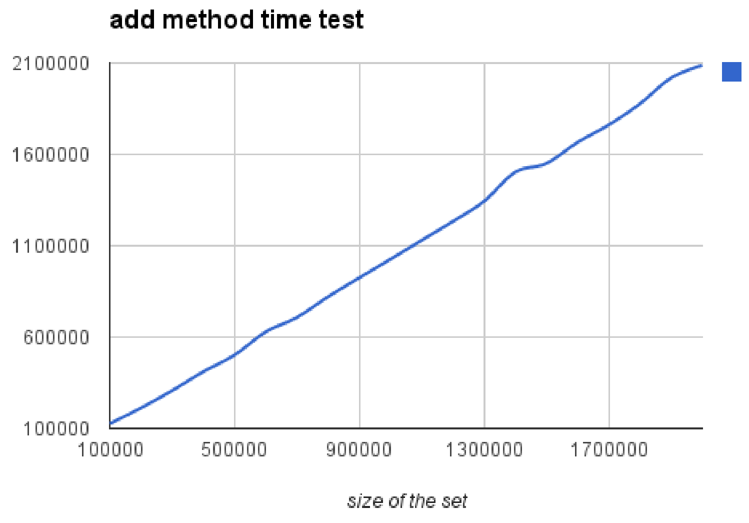
the Big-O behavior should be O(logN), because it using the binary searching method, so, for a N amount data, it should be at most lnN/ln2 times.

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, I think it is match the log(N) Big-o behavior.



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)?



I think it should be O(N), because every time when it insert one elements, it will move the elements after it backward one index, so it is linear.

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

   around 5-6 hours, we spend a lot of time to fix some problems.