Nguyen Truong

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

- Partner: Richard Campbell
- I'm the one who will be submitting 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? Every 2 hr, I think I'm fine by how often we switch because I'm able to keep track of thing easier

3. Evaluate your programming partner. Do you plan to work with this person again? We work well together so no

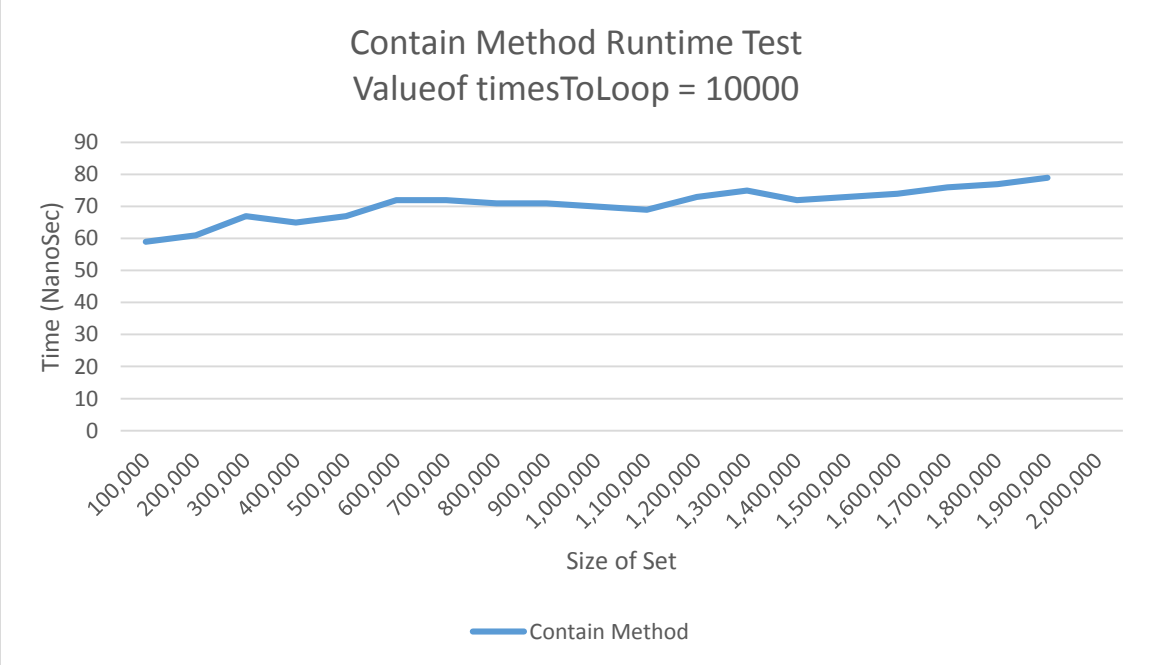
4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development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

- It will be more efficient because java List itself have several method build into Java that's useful to implement the sort. I would have to declare all of the arrays variable in my code to ArrayList of Collections

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

- With the use of binary search it should be O(log N)

6. Plot the running time of MySortedSet's contains method for sets of sizes 100,000 to 2,000,000 by steps of 100,000. 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 does

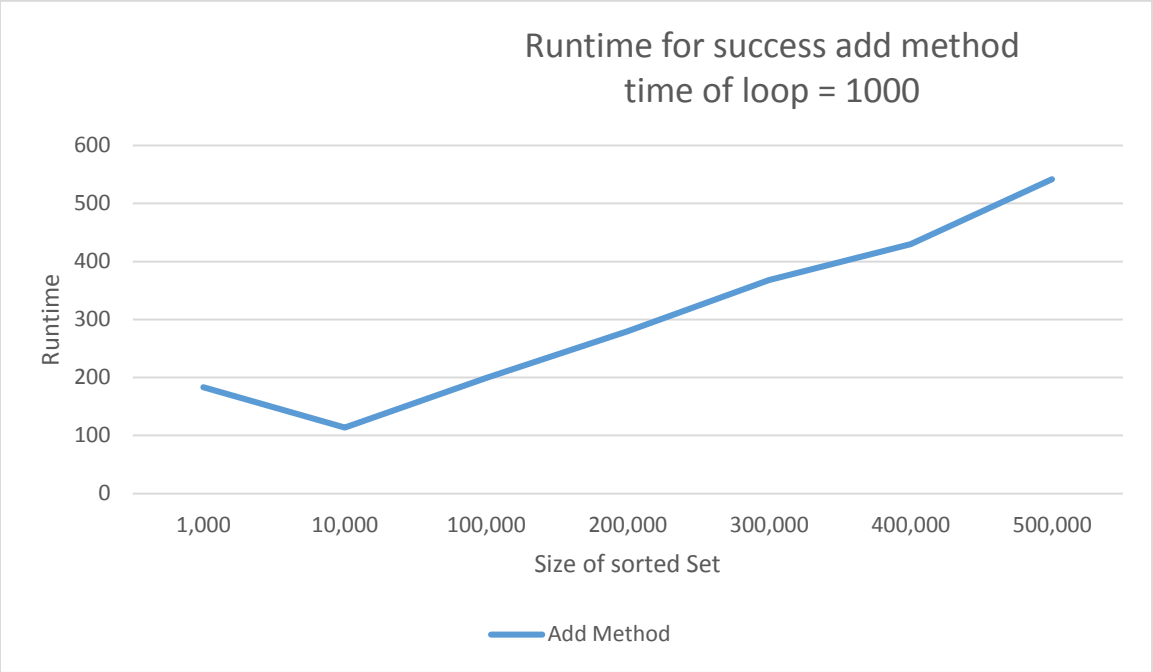


7. Consider your add method. For an element not already contained in the set, how long does it take to locate the correct position at which to insert the element?

- O(log N) because we use binary search to search for the correct position.

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 worst case it's still O(log N) because we still have to go through the whole array using binary search.



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

- 9 hr.