Analysis Document – Assignment #3 Emily Dennis

When you are satisfied that your program is correct, write a brief analysis document. The analysis document is 30% of your Assignment 3 grade. Ensure that your analysis document addresses the following.

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

Rithie Penn. I 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 often enough that I felt comfortable with the paired programming. If we had switched more, then it would have been less efficient and silly to do.

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

My partner was good to work with. I found that I was actually pretty good with the navigating and when we got caught up in the details of the code, he really picked it up where I was struggling. I would definitely work with this partner 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.)

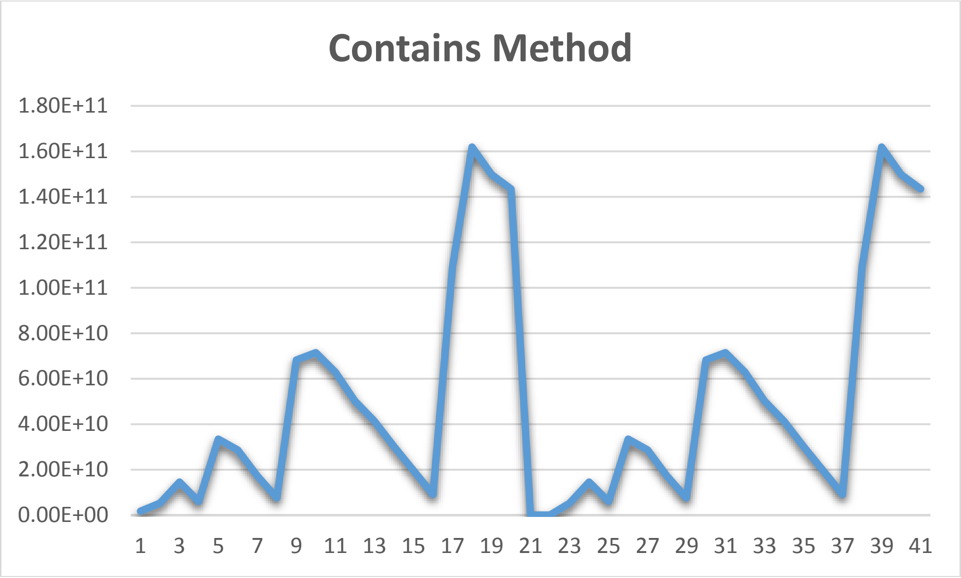
Using a Java List would be much faster in program development time. The running time is roughly the same since we basically wrote a list/set based off an array. ArrayLists are also based off of arrays. The main ways the code would have differed would be in that we wouldn't have to implement add, contains, and remove methods.

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

O(log N), because the search is more or less dividing the search in two every iteration. This is because the search's domain is halved every time through.

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?

The growth rate doesn't appear regular at all, and this seems to be because all of the stuff running in the background of Java.



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?

15 hours

Programming partners are encouraged to collaborate on the answers to these questions. However, each partner must write and submit his/her own solutions.

Upload your document (.pdf only!) to the Assignment 3 page by 11:59pm on February 5.