Robert Weischedel

2/5/15

CS 2420

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

My Programming Partner is Mackenzie Elliott, and I Robert Weischedel submitted the Java source code for this programming assignment.

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 frequently, around every 30 minutes are so. I think that this amount of time is adequate because these are difficult problems we are solving, so naturally it does take some time to solve them. So I feel if we did any sooner, we would be spending too much time changing roles and not getting the assignment done.

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

Mackenzie is a great partner, she works very hard and is really engaged in the assignment. I definitely plan on working with her 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.)

If we had done this, we wouldn't have had to do anything for this assignment. All of the methods we implemented are already implemented by the ArrayList Class. This includes a sorting method that can use a specified Comparator or the natural ordering of an object. I would say that the ArrayList would be faster and more efficient because first we wouldn't have had to implement any code. And second, the algorithms that the ArrayList uses for the operations are optimized for efficiency and speed, while ours are optimized to the best of our sorting algorithm knowledge, which is quite basic.

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

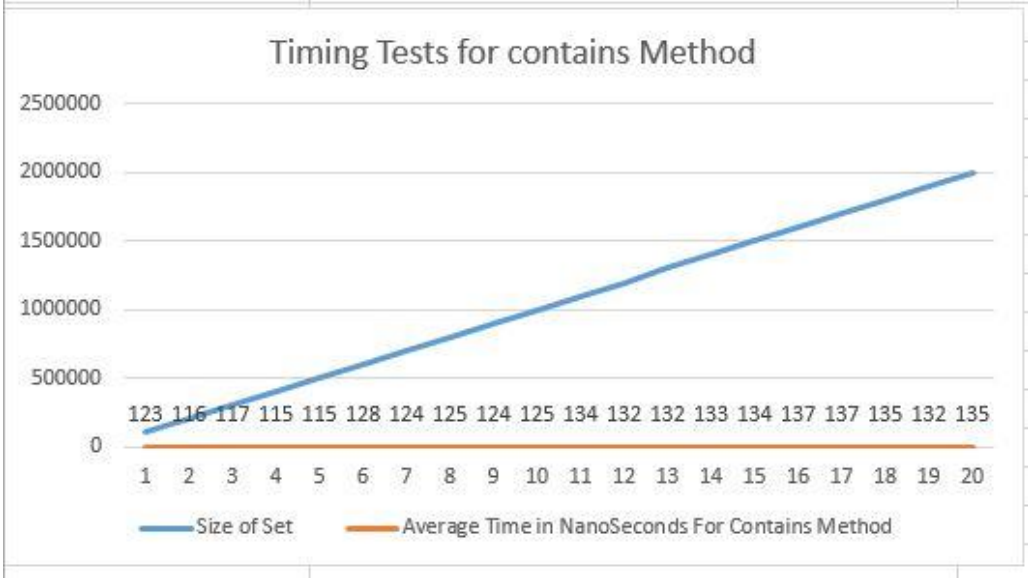
Since out contains method itself contains a few assignments and a few if statements, I would say that the Big-O behavior would have to be (O) c. But since the contains method calls the binary search every time it runs and since that half's lists to search through them, that makes the contains method that of Log N.

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 the growth rate is similar to that of Log(N). For our worst case scenario, we choose to have the contains method find the value of 0 each time.

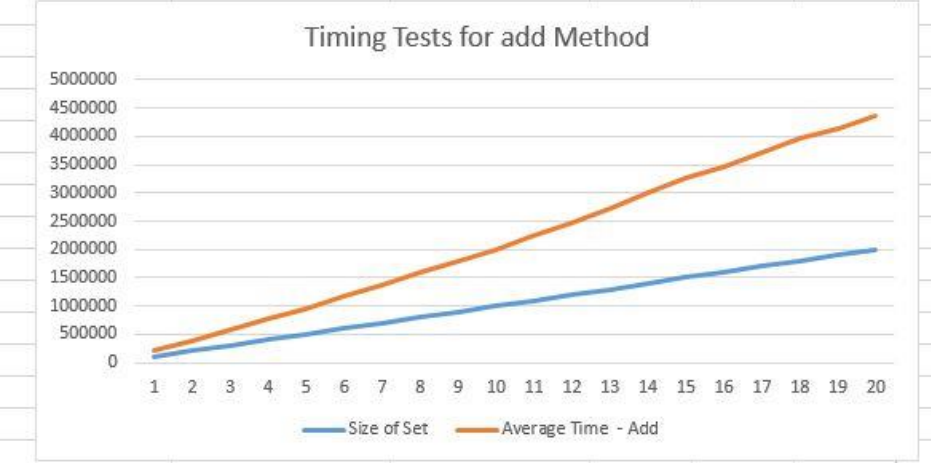
Size of Set	Average Time in NanoSeconds For Contains Method
100000	123
200000	116
300000	117
400000	115
500000	115
600000	128
700000	124
800000	125
900000	124
1000000	125
1100000	134
1200000	132
1300000	132
1400000	133
1500000	134
1600000	137
1700000	137
1800000	135
1900000	132
2000000	135



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 our worst case scenario, we keep adding new values or objects to the end of the list. And from our analysis, we were able to determine that the correlation of the timing of the add method was that of a linear nature or (O) n.

Size of Set	Average Time - Add
100000	213527
200000	383731
300000	583094
400000	774831
500000	960890
600000	1170537
700000	1383170
800000	1599326
900000	1782641
1000000	2003060
1100000	2239309
1200000	2481992
1300000	2724132
1400000	3015361
1500000	3263557
1600000	3472463
1700000	3724590
1800000	3962023
1900000	4131825
2000000	4350124



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