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?
David Gillespie

2. How often did you and your programming partner switch roles? Would you have preferred to switch less/more often? Why or why not?
10

3. Evaluate your programming partner. Do you plan to work with this person again?
David is great, and I definitely plan on working with him again. We collaborate well together.

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.)
get() would have been used for the first() and last() method, and all other methods with the exception of add() and addAll() have already been implemented. Development time would be exponentially shorter, while the running time ought to be the same depending on the type of List chosen.

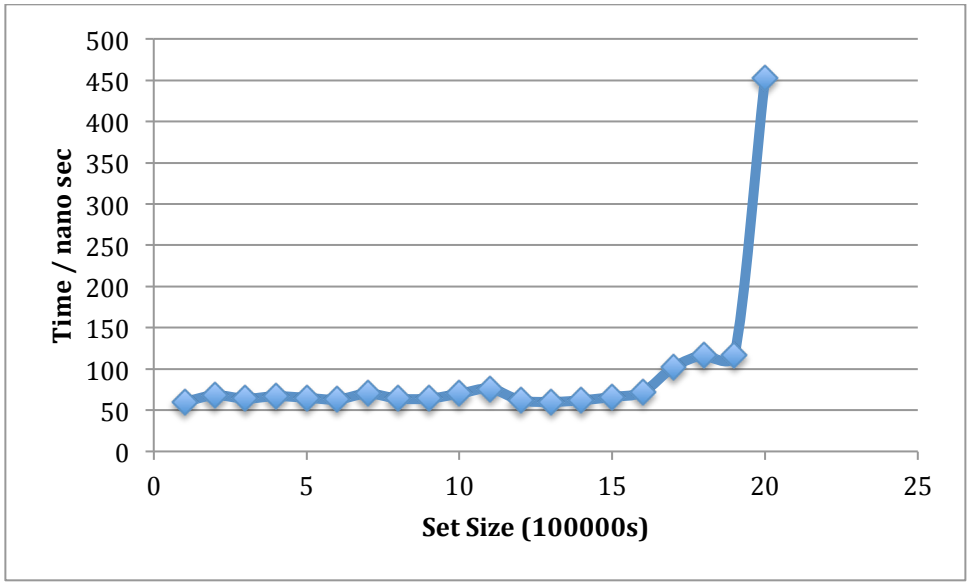
5. What do you expect the Big-O behavior of MySortedSet's contains method to be and why?
O(log N). The contains method implements the binary search, which divides the set in two each pass.

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?

Disregarding the last value ran at 2,000,000, the run time looks either constant or log n, the latter being the most likely given the structure of the algorithm.

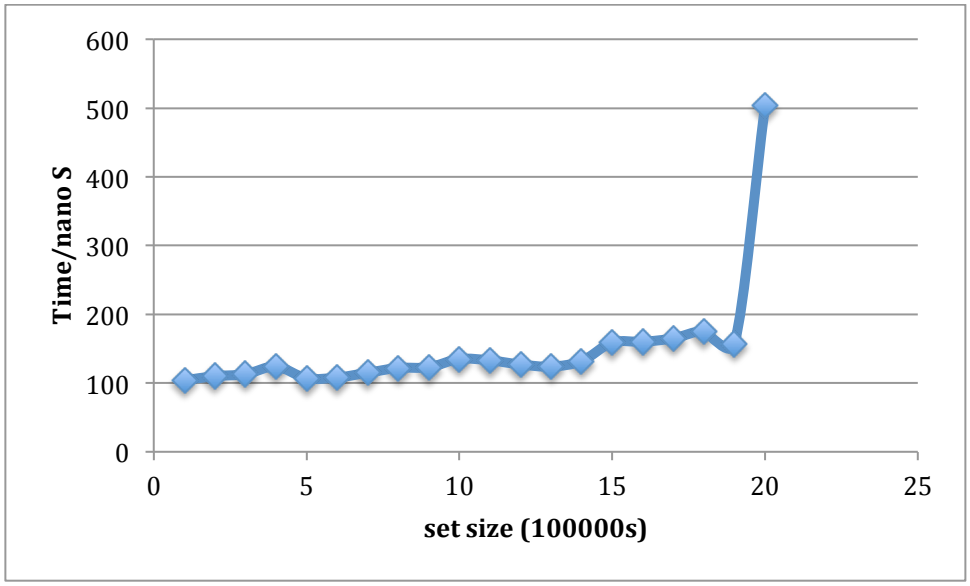
The test was ran by creating a set of the specified size from 0 to size-1. A number is randomly generated as the target, before running a timed test with loops of 1 million.

60	68	64	67	65	63	70	64	64	70
76	62	60	62	66	72	102	117	117	453



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)?
By algorithm analysis, the program ought to run in O(N log N) complexity. However, the graph does not seem to reflect this, even if the 2,000,000 set was discarded.

104	110	113	124	107	108	115	122	123	135
133	127	124	132	159	160	165	175	157	503



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

8

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.