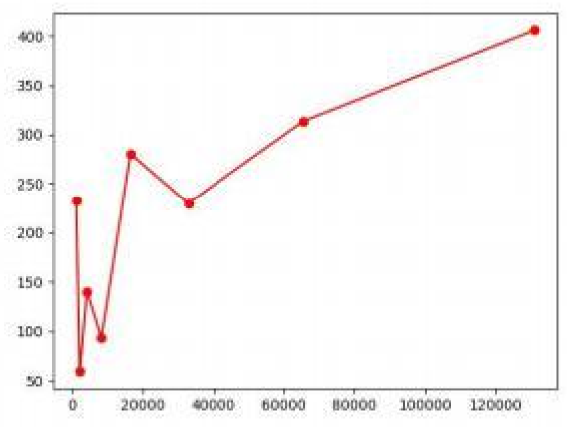
1. **If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed.Do you expect that using a Java List would have more or less efficient and why?(Consider efficiency both in running time and in program development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

- **Flexibility and Ease of Use:** Java Lists provide more flexibility and built-in methods, making development quicker and potentially simpler. Lists handle resizing automatically, reducing the complexity of code needed to manage array size.
- **Performance Efficiency:** While Lists offer convenience, they may be less efficient in running time compared to arrays, especially for large datasets. This is due to the overhead associated with List operations (like auto-resizing) and potentially less efficient memory usage.
- **Development Time:** Using a Java List could reduce development time due to its higher-level functionality and abstraction. However, this could come at the cost of reduced runtime efficiency, especially for operations that require frequent resizing or reordering of elements.

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

Expect:O(logN), because I use binary search, each time I find specific position in half of current size.

3.Plot the running time of BinarySearchSet's contains method,using the timing techniques demonstrated in previous labs. Be sure to use a decent iteration count 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 2?



contains method for binnarySearchSet

Yes, almost like O(logN)

4.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.

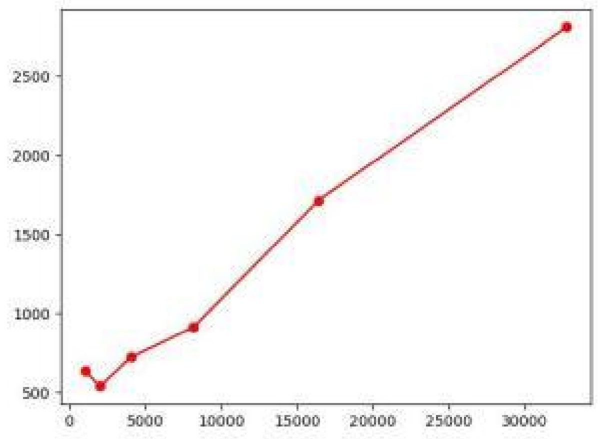


Table 2 add method

The locating operation alone is O(log n), while the entire add operation, including insertion, is O(n) in the worst case.