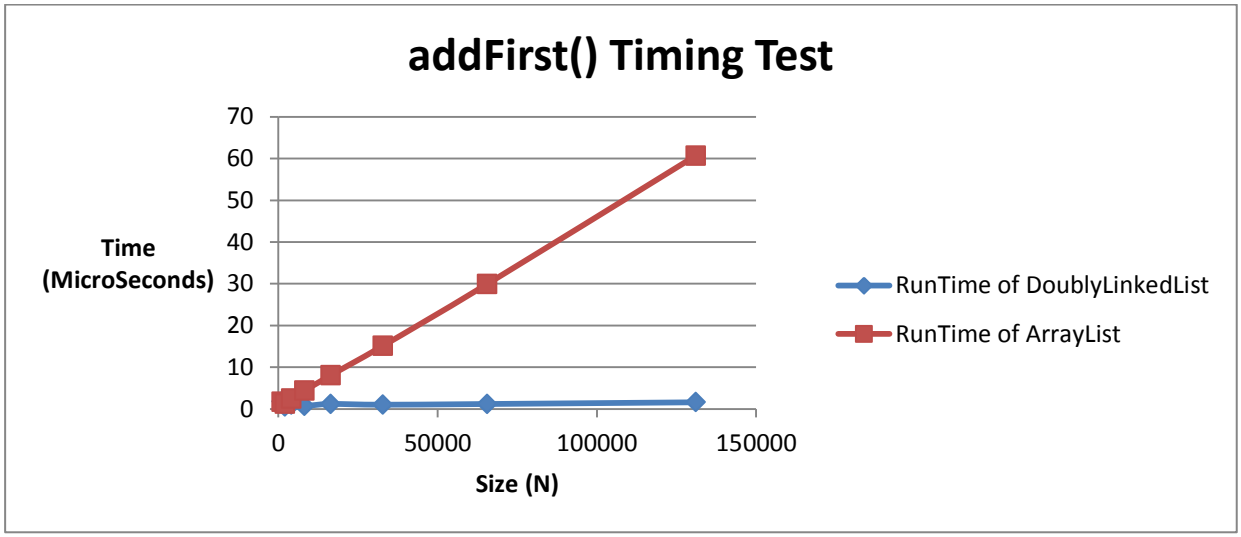
Christian Hansen
U0621884

Analysis Document of Assignment 6

1. Collect and plot running times in order to answer each of the following questions. Note that this is this first assignment that does not specify the exact procedure for creating plots. You must design your own timing experiments that sufficiently analyze the problems. Be sure to explain all plots and answers.

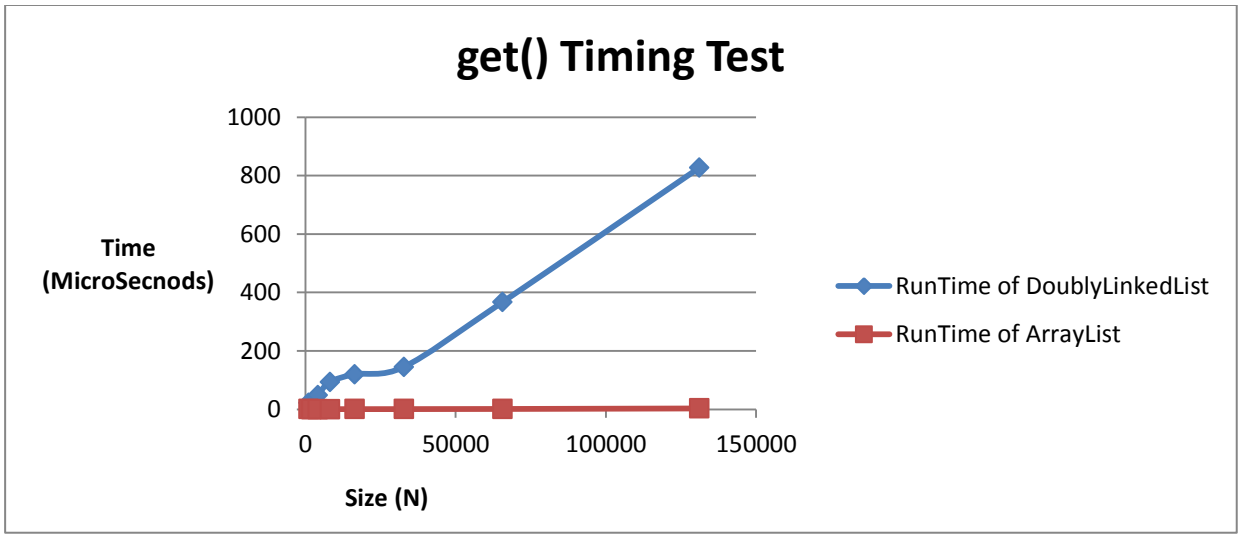
Is the running time of the addFirst method O(c) as expected? How does the running time of addFirst(item) for DoublyLinkedList compare to add(0, item) for ArrayList?

Yes, DoublyLinkedList is O(c) as expected. The chart below shows the performance of the two types with adding an item to the beginning of the list. The DoublyLinkedList performs superior to ArrayList. ArrayList is O(n) while DoublyLinkedList is O(c) in terms of Big O notation. Constant is superior to linear.



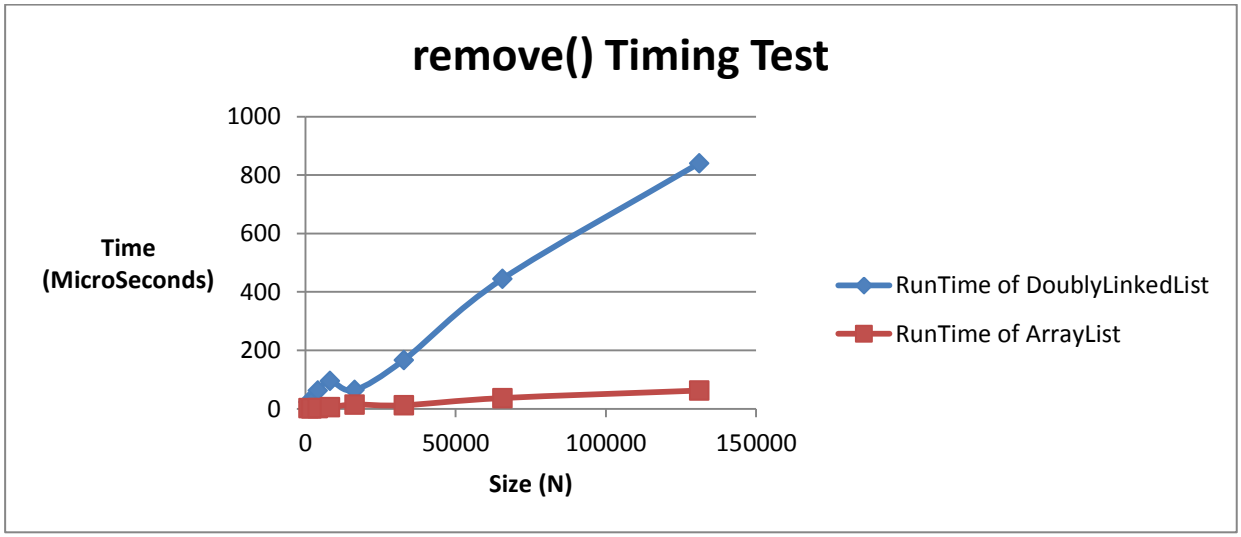
Is the running time of the get method O(N) as expected? How does the running time of get(i) for DoublyLinkedList compare to get(i) for ArrayList?

The DoublyLinkedList is O(N) as expected. The graph below demonstrates that the get method of ArrayList is superior to DoublyLinkedList, because the ArrayList is O(c).



Is the running time of the remove method O(N) as expected? How does the running time of remove(i) for DoublyLinkedList compare to remove(i) for ArrayList?

The running time for DoublyLinkedList is O(N) as expected. ArrayList is superior with O(C). The graph below shows the results.



2. In general, how does DoublyLinkedList compare to ArrayList, both in functionality and performance? Please refer to Java's ArrayList documentation.

While DoublyLinkedList and ArrayList have mostly the same functionality, the performance is completely different. The ArrayList is backed by an array that is expanded when it needs to be. This means that each item in the Array has a memory address, which means we can access it instantly. In some cases this leads to superior performance. It runs into problems when we want to slide items down one. If we were to insert an item at the beginning, the DoublyLinkedList can do this in constant time, the ArrayList must slide all the items to the right which takes linear time.

3. In general, how does DoublyLinkedList compare to Java's LinkedList, both in functionality and performance?Please refer to Java's LinkedList documentation.

There are many things similar between the class that I have created and Javas. The nodes are written very similarly. There is a lot more functions in the LinkedList that Java created. They used transient instead of private with their member variables head and tail. One function in particular, addAll, would be very useful as one could quickly create a LinkedList from another Collection.

4. Compare and contrast using a LinkedList vs an ArrayList as the backing data structure for the BinarySearchSet (Assignment 3). Would the Big-Oh complexity change on add / remove / contains?

The biggest differences would be adding items to the end and beginning as LinkedList can do that in constant time, and There would be no need to expand or shift elements to the right or left. LinkedList would still need to be searched for items, but if they were to be removed the LinkedList could do this in constant time. So the Big-O complexity would change and it would be a better fit for the BinarySearchSet since the BinarySearch set did not take advantage of ArrayList's best feature the fact that the array positions each have a memory address. The BinarySearchSet did not implent a get method

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

I spent 14 hours on this assignment.