Created by Curtis Kokuloku (kokul003) and Minh Tong (tong0154)

Method	**ArrayList Runtime**	**LinkedList Runtime**	**Explanation**
add(T element)	O(1)	O(n)	A: Doesn't iterate through anything so it is O(1). It does implement isSortedChecker, but that doesn't affect the final outcome, only a variable elsewhere. L: Iterates through each node in the linked list to get to the end of it with a while loop, which doesn't affect the runtime much unless you start using huge lists, so it is O(n).
pairSwap()	O(n)	O(n)	A: Iterates through each element in the array list with one for loop until it reaches the size of the array. Changes based on the size of the one array list you swap, so it's O(n). L: Iterates through each node in the linked list with one while loop until it reaches a .getNext() == null. Changes based on the size of the one linked list you swap, so it's O(n).
merge(List<T>otherList)	O(n)	O(n)	A: Iterates through each element in the two array lists with three while loops until equal to this.counter or other.size Changes based on the size of the two lists you add together, so it's O(n)

			and not O(n^2) because the multiple while loops aren't nested. L: Iterates through each node in the two linked lists with one while loop until the currentSize of the merged list is == the size of the two original lists added together. Changes based on the size of the two lists you add together, so it's O(n) and not O(n^2) because there's only one while loop.
reverse()	O(n)	O(n)	A: Iterates once for every value of counter - 1 until it reaches a negative number. It also uses the add method within the for loop but, since the add method for array list is O(1), then the O complexity for reverse() is just O(n). L: Iterates through one while loop until curr == null, nothing special, so it's O(n).

End of Analysis:

This makes sense in practice, because the runtime was 3.567 seconds for the array list and 3.426 for the linked list at the point where O(n) complexity methods seem faster than O(1) complexity methods with 1000 tests in the Unit Tests. With relatively few tests, the linked list test still seems faster than the array list test. When the same array list and linked list were tested with 10000 tests instead, the linked list test took 25.894 seconds while the array list only took 20.15 seconds. In the end, **Array list is better**, because all of their methods have the same time complexity, besides add, where array list's add method is O(1) and linked list's add method is O(n).