Name: Jingyi Ouyang
PID: A53108909

1. Making a copy of the list.

LinkedList: Runtime: Ω(n). Using an iterator to get each element in a LinkedList.
And copy this element into a new node and add it into a new LinkedList. There are
n elements in total. So it runs n times which corresponds to Ω(n).

ArrayList: Runtime: Ω(n). We can use its index to get an element directly which costs Ω(1). Copy it into the corresponding position in a new ArrayList which takes
Ω(1). There are n elements that need to be copied and added, which takes Ω(n).

2. Adding a value to the end of the list.

LinkedList: Runtime: Ω(1). Using tail node to get last node. Adding the new node containing new value after the last node and make the tail point at new node. The whole process takes Ω(1).

ArrayList: Runtime: Ω(n) (worst case). ArrayList is operated based on array. When the ArrayList is small which needs to grow size, the ArrayList will grow to a twice larger one. And then copy all the elements from the old ArrayList to the new ArrayList. This process takes Ω(n).

3. Removing the first value from the list.

LinkedList: Runtime: Ω(1). Making the head point to head.next.next to remove the first node in the LinkedList, which takes Ω(1).

ArrayList: Runtime: Ω(n). Using index to get first element and remove it. But all the elements after the first element needs to be shifted forward by one position. The whole process takes Ω(n).

4. Removing the last value from the list.

LinkedList: Runtime: Ω(1). Using the tail to get the last node and tail.prev to get the previous node. Remove the last node and make the tail point to tail.prev, which takes Ω(1).

ArrayList: Runtime: Ω(1). Using index to get last element and remove it directly.
This takes Ω(1).

5. Determining whether the list contains some value V.

Worst case is that the value we want to find is at the end of the list.
LinkedList: Runtime: Ω(n). Using iterator traverses the LinkedList to get the value.
This takes Ω(n).

ArrayList: Runtime: Ω(n). Using for loop to traverse the ArrayList to get the value:

This also takes $\Omega(n)$.