[**Closed book and notes.**] Assume ArrayList and LinkedList refer to the implementations developed in lecture. If asked to implement a method, you may not assume that any other methods from the class have already been implemented unless explicitly stated. Show all of your work clearly in the space provided or on the additional page at the end of the exam. If the additional page is used, clearly identify to which exam question it is related. Be sure to **read each problem carefully**. Note that the exam is double sided.

1. (10 points) Consider the following method:

```
1   public static boolean areUnique(List<String> strings) {
2       boolean areUnique = true;
3       int size = strings.size();
4       for(int i=0; areUnique && i<size; ++i) {
5           for(int j=0; areUnique && j<size; ++j) {
6               if(i!=j && strings.get(i).equals(strings.get(j)) {
7                   areUnique = false;
8               }
9           }
10      }
11      return areUnique;
12  }
```

(a) (5 points) Suppose an ArrayList is passed to areUnique(), give the Big-O time complexity for the method. Justify your answer.

(b) (5 points) Suppose an LinkedList is passed to areUnique(), give the Big-O time complexity for the method. Justify your answer.

2. (5 points) Describe the differences in the java.util.ArrayList implementation compared to the one created in lecture that affect the asymptotic time complexity of any of the methods in the List interface.

3. (5 points) Explain how the Collection and List interfaces differ. Give examples of at least three methods that are not common to both interfaces.

4. (2 points) Write the class declaration for the ArrayList class, i.e., the first line of the class beginning with `public class`...

5. (15 points) Recall that our ArrayList had one attribute: E[] data. Implement the ArrayList.remove(int index) method.

6. Recall that our LinkedList had one attribute: Node head.
(a) (7 points) Suppose that a second attribute, Node tail, is added to the class which points to the last node in the LinkedList. Implement the LinkedList.add(E element) method.

(b) (3 points) What is the Big-O time complexity for your implementation in part **(a)**? Justify your answer.

(c) (8 points) Implement the following method that returns the node in the list at the specified index. You may assume only valid values of index are passed to the method.

```
private Node getNode(int index) {
```

(d) (10 points) Suppose a third attribute, int size, is added to our LinkedList. The attribute should contain a value that corresponds to the size of the list. Implement the add(int index, E element) method for the modified class. You may assume only valid values of index are passed to the method.

7.

(a) (10 points) Complete the implementation of the following method **without** using an enhanced for loop or the Collections.max() method.

```
public static double findLargest(Collection<Double> numbers) {
```

(b) (5 points) Give the Big-O time complexity for your implementation of in part **(a)**. Justify your answer.