[**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  public static long sumEven(List<Integer> numbers) {
2    if(numbers==null) {
3      throw new IllegalArgumentException("List cannot be null");
4    }
5    long sum = 0;
6    for(int i=0; i<numbers.size(); ++i) {
7      if(numbers.get(i)%2==0) {
8        sum += numbers.get(i);
9
10     }
11   }
12   return sum;
13 }
```

Assume n is the size of the List passed to sumEven()

1. (20 points) True/False (**T** or **F**)

____ The asymptotic time complexity for sumEven() is $O(n)$ for a java.util.ArrayList.

____ The asymptotic time complexity would be different for a LinkedList containing all even numbers than for a LinkedList containing all odd numbers.

____ numbers could be a Collection<Integer> instead of List<Integer> and not require **any** other changes.

Suppose we pass an object from a LinkedList to sumEven(). For sumEven() to function correctly, the following methods must be implemented in the LinkedList class: (indicate T or F for each method)

____ size()

____ iterator()

____ isEven()

____ isNull()

Suppose numbers.remove(i–); is inserted on line 9.

____ sumEven() will crash.

____ sumEven() will run forever.

____ The asymptotic time complexity for sumEven() when operating on a java.util.LinkedList will change.

2. (10 points) Describe the differences in the java.util.LinkedList implementation compared to the one created in lecture that affect the asymptotic time complexity of any of the methods in the List interface.

3. (10 points) Explain the purpose of Big-O notation.

4. (5 points) Suppose $T(n) = \log n + 3n + 8$. Determine the Big-O notation.

5. Recall that our ArrayList had one attribute: data and our LinkedList had one attribute: head.
(a) (5 points) Implement the one argument ArrayList.get() method.

(b) (10 points) Implement the one argument LinkedList.get() method.

6. (20 points) Suppose a second attribute, 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.

7.

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

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

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

(c) (5 points) Justify your answer to part **(b)**.

Additional work area for any problem. Clearly identify which problem is associated with the work on this page.