You have 50 minutes to complete this exam.

You do not need to show templates or an Examples class.

You do not need to write Javadoc or any other documentation.
1. (30 points) A library keeps track of items in its collection using a `LinkedList<ILibraryItem>`, where there are two types of library items: Books and CDs. The library software needs to be able to generate lists of items that meet certain criteria (such as overdue items, or books by a given author, or cd’s of classical music, etc.) The selection criteria is specified by a visitor interface called `IVisitor`. Here is the relevant code:

```java
interface ILibraryItem{
    // return the result of processing this ILibraryItem with the given IVisitor
    boolean accept(IVisitor select);
}

interface IVisitor{
    boolean selectBook(Book b);
    boolean selectCD(CD c);
}
```

(a) (10 points) The library needs to be able to select Books by the author named “Wolfe”. Define a class `BookByWolfe` that implements `IVisitor` (you should write all of the code for the class).
(b) (10 points) Here is a partial implementation of the Book class. Complete the Book class by writing the method `accept`, which is specified in the `ILibraryItem` interface.

```java
public class Book implements ILibraryItem{
    private String title;
    private String author;
    private int isbn;

    public Book (String title, String author, int isbn){
        this.title = title;
        this.author = author;
        this.isbn = isbn;
    }

    public String getAuthor(){
        return this.author;
    }

    // implement the accept method here
}
```

(c) (10 points) The library wants to add DVDs as a new kind of `ILibraryItem`. Describe all the changes that will need to be made to the existing hierarchy (including the `BookByWolfe` class) to accommodate the addition of DVDs. (Don’t write code, just describe what needs to be done.)
2. (40 points) Using the Node class definition that we developed in lecture, here is an implementation of a method canReach() that returns a list of the names of cities reachable from this Node:

```java
public class Node{
    private String cityname;
    private LinkedList<Node> connects;

    public Node(String cityname){
        this.cityname = cityname;
        this.connects = new LinkedList<Node>();
    }

    public void addEdge(Node toNode){
        this.connects.add(toNode);
    }

    public LinkedList<String> canReach(){
        LinkedList<String> reachable = new LinkedList<String>();
        canReachVisited(reachable, new LinkedList<Node>());
        return reachable;
    }

    private void canReachVisited(LinkedList<String> reachable, LinkedList<Node> visited){
        if (!visited.contains(this)){
            visited.add(this);
            reachable.add(this.cityname);
            canReachConnects(reachable, visited);
        }
    }

    private void canReachConnects(LinkedList<String> reachable, LinkedList<Node> visited){
        for (Node c: this.connects){
            c.canReachVisited(reachable, visited);
        }
    }
}
```
(a) (15 points) Write down an argument for why the list returned by `canReach()` won’t have any duplicates.

(b) (5 points) Assume that `Nodes` are created and edges are added with the `addEdge()` method to represent the following graph:

```
  Manchester
   ↓
  Boston          Worcester
   ↓                   ↓
 Providence        Hartford
```

Suppose the Node that represents Boston in the pictured graph has the name `bost`, and you run a test on the method `canReach()` with this call:

```
bost.canReach()
```

How many possible correct answers are there for this method call?

i. 1
ii. 4
iii. 6
iv. 24
v. none of the above
(c) (20 points) Describe how you would test `canReach()`. Either provide `checkExpect` or `JUnit` type test(s), or just describe in words the method(s) you would develop to do the testing.
3. (20 points) A class called `SampleClass` contains the following method signature:

```
public String foo (int x, String s);
```

You decide to implement the function `foo` recursively, and to use memoization to store results. Write a list of steps that outline what you need to do in order to create a memoization table and effectively use memoization. Be as explicit as you can as you describe each step, given the information provided above in the method signature. You may answer this question with any combination of code/prose.
4. (10 points) Suppose you have a `LinkedList<Integer>` called `numList`, and you want to count the number of negative Integers in `numList`. Here is the Javadoc for some of the methods in the class `LinkedList<E>`:

   - `E get(int index)`
     
     Returns the element at the specified position in this list

   - `E set(int index, E element)`
     
     Replaces the element at the specified position in this list with the specified element

   - `int size()`
     
     Returns the number of elements in this list

Using a looping mechanism of your choice (either `foreach`, `for`, or `while`), write the Java statements to count the number of negative Integers in `numList`. You do not have to write a complete method, just the code for the loop.