Name (First and Last):_____

CS II: Data Structures
Pre-Lab 4: Exam 1 Review

Next Tuesday is your exam. This week's lab will be review work. For Pre-lab fill out this review worksheet. It will take some time, so start early. **You will turn this in to your TA at the beginning of class!**

1. What does it print?

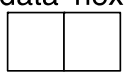
Circle one answer for each program. Does the program print true, print false, or is there not enough information to know what the program prints?

`public static void main(String[] args) {`
`  String a = new String("Cloud");`
`  String b = new String("Cloud");`
`  b = mysteryFunction(a, b);`
`  System.out.println(a == b);`
`}`
	a) true
	b) false
	c) not enough information

`public static void main(String[] args) {`
`  String a = new String("Cloud");`
`  String b = new String("Cloud");`
`  anotherMysteryFunction(a, b);`
`  System.out.println(a == b);`
`}`
	a) true
	b) false
	c) not enough information

`public static void main(String[] args) {`
`  String a = new String("Cloud");`
`  String b = new String("Cloud");`
`  System.out.println(a.equals(b));`
`}`
	a) true
	b) false
	c) not enough information

2. Manipulate objects and references

In the blank space to the right of each segment of code, draw a diagram showing the state of the variables and objects in the program AFTER those lines have executed. ListNode *objects* are drawn as two adjacent containers 

with the boxes filled in. **There are 5 diagrams**

for you to draw. We drew the first diagram for you. Don't worry: even if you get a step wrong, we'll grade the next steps based on your diagram.

`ListNode p = null;`	p /
`p = new ListNode(10);` `p.next = new ListNode(20);`	
`ListNode x = p.next;`	
`ListNode y = p.next.next;`	
`x = y;`	
`p.next = new ListNode(30);`	

3. Write a method for a linked list

We want our LinkedList class to have a method that removes multiple elements from the list and returns them in a *new* LinkedList.

`removeStartingAt` takes one argument, the index of the first element to remove. The *new* list that it returns will contain all elements starting from that index to the end of the list.

Example

```
1  LinkedList mylist = new LinkedList();
2  mylist.append(10);
3  mylist.append(20);
4  mylist.append(30);
5  mylist.append(40);
6  // mylist contains [10,20,30,40]
7  LinkedList theend = mylist.removeStartingAt(2);
8  // mylist contains [10, 20] and theend contains [30, 40]
```

a) Draw the state of *all* variables and objects right BEFORE line 7 executes. (LinkedList is defined on the following page)

b) Draw the state of *all* variables and objects AFTER line 7 executes. (LinkedList is defined on the following page)

c) We've provided the important parts of the `LinkedList` class. Fill in the definition of the `removeStartingAt` method. You may write private helper methods on the following blank page if needed. Assume the method will never be called on a bad argument.

You will be graded on the correctness of your solution at a conceptual level. Syntax errors only matter if they create ambiguity that your answer is correct.

```
public class LinkedList {
      // header points to a sentinel node
      private ListNode header;

      public LinkedList() {
            header = new ListNode(-1);
      }

      private class ListNode {
            public int data;
            public ListNode next;

            public ListNode(int data) {
                  this.data = data;
                  this.next = null;
            }
      }

      public LinkedList removeStartingAt(int start) {









      }
}
```