1 Questions

Please answer these questions as though they had been given in a testing situation. These are designed to represent the likely kinds of questions that you will encounter on exams, as well as to test your general knowledge of the kinds of topics most likely included on exams.

1. Write the recursive method that removes the last item from any LinkedList<T>. Note, your method should delegate to a private helper method:

   ```java
   public void removeLast( LinkedList< T > list ) {
       list.head = removeLast_aux( list.head ) ;
   }
   private Node removeLast_aux( Node head ) { // start here
   ```
2. The following is in three parts. Write a recursive method that computes sequence \( \{2n - 1\} \), for the positive integer \( n \). In other words, your method should compute for a given a positive integer \( n \): \( 1 + 3 + \cdots + 2n - 1 \). Next, you will provide a detailed graph of that function computing a value, and then you will re-write the function using a stack.

(b) Begin by writing the function:

```java
public static int mystery( int n ) { // start here
```

(b) Show, by using the “graph” of the function, the result of computing `mystery(5)`. 
(c) Rewrite the mystery function by using Java’s Stack (see the appendix at the end of the document) to remove the recursion.

```java
public static int mystery(int n) { // start here
```

```
3. Override the equals method below for Vector objects. Two Vector objects are equal only when they both contain the same objects in the same order.

    // inside the class definition for Vector ...
    public boolean equals( Object other ) { // start here
4. This question requires you to implement the copy-constructor for the Vector class to
ensure that a shallow copy strategy is used, and then write a unit-test to ensure the
constructor works as intended.
(a) Write the copy-constructor, below.

(b) Complete the body of the unit-test below:
   
   ```java
   @Test
   public void testCopyVector() {
   ```
5. In the space below, provide the method signature for a public, static method named `bsearch`. `bsearch` that takes an `item` of the appropriate type and performs a `binary search` over an `ArrayList` of the appropriate types of objects. The `bsearch` method returns `true` or `false`. 
6. Use Java’s Stack class to implement the rev_aux method that is called from within your FunctionalList class. Note, rev_aux must return a fresh list (a copy) of the original list whose items appear in reversed order.

```java
public FunctionalList<T> reverse() {
    return new FunctionalList<>()
        .rev_aux(this.head);
}

private Node rev_aux(Node head) { // start here
```
Scratch Paper
Definitions used on this worksheet

Use only the following definitions when answering questions on this worksheet.

```java
class LinkedList<T> {
    class Node {
        T value;
        Node next;

        Node( T val ) { ... }
        Node( T val, Node next ) { ... }
    } // closes class Node.

    private Node head = null;
    ...
}
```

Stacks

From Java’s `Stack<T>` implementation, use only the following methods:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public void push( T item )</td>
<td>Pushes the item onto the top of the stack.</td>
</tr>
<tr>
<td>public T pop()</td>
<td>Pops (and returns) the item at the top of the stack. Note: it is an error to pop an empty stack.</td>
</tr>
<tr>
<td>public T peek()</td>
<td>Shows what is currently at the top of the stack without removing that item. Note, it is an error to call this on an empty stack.</td>
</tr>
<tr>
<td>public boolean isEmpty()</td>
<td>Returns true if the underlying stack is empty.</td>
</tr>
</tbody>
</table>

In addition, you may use the normal constructor.

Vector class

A Vector<T> is defined as:

```java
public class Vector<T> implements RandomAccess, Iterable<T> {

    public void add(T ele);
    public T get( int index );
    public int size();
    ...
}
```