Linked List Test 1

Consider the following data fields and methods.

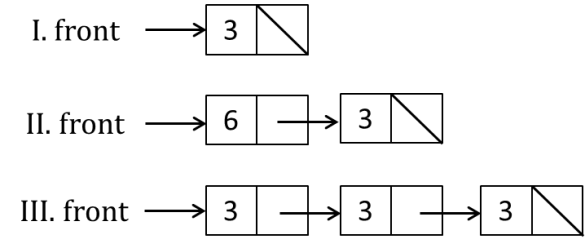
private ListNode front;

public void processList(Integer val)  
 {  
 ListNode temp, prev;  
  
 while (front != null && front.getValue().equals(val))  
 {  
 front = front.getNext();  
 }  
  
 if (front == null)  
 return;  
  
 prev = front;  
 temp = front.getNext();  
  
 while (temp != null)  
 {  
 if (temp.getValue().equals(val))  
 {  
 prev.setNext(temp.getNext());  
 }  
 else  
 {

prev = temp;  
 }  
 temp = temp.getNext();  
 }

}

1. Which of the following best describes what processList does?
2. It removes all consecutive nodes at the front of the list with value val.
3. It removes the first occurrence of the node with value val.
4. **It removes all nodes with value val.**
5. It removes all nodes with value val except the last such node.
6. Consider modifying processList by removing the if statement following the first while loop. For which of the following linked lists would the call processList(new Integer(3)) result in a run-time error?



1. I only
2. III only
3. I and II only
4. **I and III only**
5. Consider the following method:

public ListNode mystery(ListNode head)  
{  
 ListNode r = null, p = null;  
  
 while (head != null)  
 {  
 r = head.getNext();  
 head.setNext(p);  
 p = head;  
 head = r;  
 }  
 return p;  
}

If head refers to the first node of a linked list with five nodes, A → B → C → D → E, which of the   
 following lists is returned by mystery(head)?

1. B → C → D → E
2. A → B → C → D
3. A → B → C → D → E
4. **E → D → C → B → A**
5. Consider the following method.

public ListNode mystery(ListNode node)  
{

if (node == null)  
 return null;  
 else

return new ListNode(node.getValue(), mystery(node.getNext()));

}

1. Always returns null
2. **Creates and returns a copy of the given list**
3. Creates and returns a reversed copy of the given list
4. Creates and returns a copy of the first node of the given list
5. Consider the following class:

public class MyLinkedList  
{  
 private ListNode front;  
  
 public MyLinkedList()

{

front = null;  
 }

public void addLast(Object val)  
 {  
 if (front == null)  
 {

front = new ListNode(val, null);

}

else

{

*< missing code >*

}

}

}

The lastNode method should add new values to the end of the linked List. Which of the following   
 code segments can replace *< missing code >?*

1. while (front != null)  
   {  
    front = front.getNext();  
   }  
   front = new ListNode(val, null);
2. while (front.getNext() != null)  
   {  
    front = front.getNext();  
   }  
   front.setNext(new ListNode(val, null);
3. ListNode p = front;  
   while (p != null)  
   {  
    p = p.getNext();  
   }  
   p = new ListNode(val, null);
4. **ListNode p = front;  
   while (p.getNext() != null)  
   {  
    p = p.getNext();  
   }  
   p.setNext(new ListNode(val, null));**