```
                } else {
                    pre = next;
                }
                next = next.next;
            }
            cur = cur.next;
        }
    }
```

在单链表中删除指定值的节点

【题目】

给定一个链表的头节点 head 和一个整数 num，请实现函数将值为 num 的节点全部删除。

例如，链表为 1->2->3->4->null，num=3，链表调整后为：1->2->4->null。

【难度】

士 ★☆☆☆

【解答】

方法一：利用栈或者其他容器收集节点的方法。时间复杂度为 $O(N)$，额外空间复杂度为 $O(N)$。

将值不等于 num 的节点用栈收集起来，收集完成后重新连接即可。最后将栈底的节点作为新的头节点返回，具体过程请参看如下代码中的 removeValue1 方法。

```
public Node removeValue1(Node head, int num) {
    Stack<Node> stack = new Stack<Node>();
    while (head != null) {
        if (head.value != num) {
            stack.push(head);
        }
        head = head.next;
    }
    while (!stack.isEmpty()) {
        stack.peek().next = head;
        head = stack.pop();
    }
    return head;
}
```

方法二：不用任何容器而直接调整的方法。时间复杂度为 $O(N)$，额外空间复杂度为

$O(1)$。

首先从链表头开始，找到第一个值不等于 num 的节点，作为新的头节点，这个节点是肯定不用删除的，记为 newHead。继续往后遍历，假设当前节点为 cur，如果 cur 节点值等于 num，就将 cur 节点删除，删除的方式是将之前最近一个值不等于 num 的节点 pre 连接到 cur 的下一个节点，即 pre.next=cur.next；如果 cur 节点值不等于 num，就令 pre=cur，即更新最近一个值不等于 num 的节点。

具体实现过程请参看如下代码中的 removeValue2 方法。

```
public Node removeValue2(Node head, int num) {
        while (head != null) {
                if (head.value != num) {
                        break;
                }
                head = head.next;
        }
        Node pre = head;
        Node cur = head;
        while (cur != null) {
                if (cur.value == num) {
                        pre.next = cur.next;
                } else {
                        pre = cur;
                }
                cur = cur.next;
        }
        return head;
}
```

将搜索二叉树转换成双向链表

【题目】

对二叉树的节点来说，有本身的值域，有指向左孩子和右孩子的两个指针；对双向链表的节点来说，有本身的值域，有指向上一个节点和下一个节点的指针。在结构上，两种结构有相似性，现在有一棵搜索二叉树，请将其转换为一个有序的双向链表。

例如，节点定义为：

```
public class Node {
        public int value;
        public Node left;
        public Node right;
        public Node(int data) {
                this.value = data;
```