反转部分单向链表

【题目】

给定一个单向链表的头节点 head，以及两个整数 from 和 to，在单向链表上把第 from 个节点到第 to 个节点这一部分进行反转。

例如：

1->2->3->4->5->null，from=2，to=4

调整结果为：1->4->3->2->5->null

再如：

1->2->3->null，from=1，to=3

调整结果为：3->2->1->null

【要求】

1．如果链表长度为 N，时间复杂度要求为 $O(N)$，额外空间复杂度要求为 $O(1)$。

2．如果不满足 1<=from<=to<=N，则不用调整。

【难度】

士 ★☆☆☆

【解答】

本题有可能存在换头的问题，比如题目的第二个例子，所以函数应该返回调整后的新头节点，整个处理过程如下：

1．先判断是否满足 1<=from<=to<=N，如果不满足，则直接返回原来的头节点。

2．找到第 from-1 个节点 fPre 和第 to+1 个节点 tPos。fPre 即是要反转部分的前一个节点，tPos 是反转部分的后一个节点。把反转的部分先反转，然后正确地连接 fPre 和 tPos。

例如：1->2->3->4->null，假设 fPre 为节点 1，tPos 为节点 4，要反转部分为 2->3。先反转成 3->2，然后 fPre 连向节点 3，节点 2 连向 tPos，就变成了 1->3->2->4->null。

3．如果 fPre 为 null，说明反转部分是包含头节点的，则返回新的头节点，也就是没反转之前反转部分的最后一个节点，也是反转之后反转部分的第一个节点；如果 fPre 不为

null，则返回旧的头节点。

全部过程请参看如下代码中的 reversePart 方法。

```
public Node reversePart(Node head, int from, int to) {
        int len = 0;
        Node node1 = head;
        Node fPre = null;
        Node tPos = null;
        while (node1 != null) {
                len++;
                fPre = len == from - 1 ? node1 : fPre;
                tPos = len == to + 1 ? node1 : tPos;
                node1 = node1.next;
        }
        if (from > to || from < 1 || to > len) {
                return head;
        }
        node1 = fPre == null ? head : fPre.next;
        Node node2 = node1.next;
        node1.next = tPos;
        Node next = null;
        while (node2 != tPos) {
                next = node2.next;
                node2.next = node1;
                node1 = node2;
                node2 = next;
        }
        if (fPre != null) {
                fPre.next = node1;
                return head;
        }
        return node1;
}
```

环形单链表的约瑟夫问题

【题目】

据说著名犹太历史学家 Josephus 有过以下故事：在罗马人占领乔塔帕特后，39 个犹太人与 Josephus 及他的朋友躲到一个洞中，39 个犹太人决定宁愿死也不要被敌人抓到，于是决定了一个自杀方式，41 个人排成一个圆圈，由第 1 个人开始报数，报数到 3 的人就自杀，然后再由下一个人重新报 1，报数到 3 的人再自杀，这样依次下去，直到剩下最后一个人时，那个人可以自由选择自己的命运。这就是著名的约瑟夫问题。现在请用单向环形链表描述该结构并呈现整个自杀过程。