链滴

算法 03 链表的基本算法

作者：AshShawn

原文链接：https://ld246.com/article/1646200080749

来源网站：链滴

1.单向链表

```
public class Node {
    public int value;
    public Node next;
    public Node(int data) {
        value = data;
    }
}
```

2.双向链表

```
public class DoubleNode {
    public int value;
    public DoubleNode pre;
    public DoubleNode next;

    public DoubleNode(int data) {
        value = data;
    }
}
```

3.链表反转

3.1 单链表反转

```
    /**
     * 单链表反转
     */
    public static Node reverse(Node node) {
        //省略node校验
        Node temp = null;
        Node pre = null;
        while (node != null) {
            //temp -> 下一个节点
            temp = node.next;
            //当前节点的next -> 上一个节点
            node.next = pre;
            //上个节点切换为node
            pre = node;
            //node切换成temp
            node = temp;
        }
        return node;
    }
```

3.2 双链表反转

```
    /**
     * 双向链表反转
```

```
 * */
public static DoubleNode reverse(DoubleNode node) {
    //省略node校验
    DoubleNode temp = null;
    DoubleNode pre = null;
    while (node != null) {
        //temp -> 下一个节点
        temp = node.next;
        //当前节点的next -> 上一个节点
        node.next = pre;
        //比单链表多了这异步,当前节点的pre  -> 下个节点即temp
        node.pre = temp;
        //上个节点切换为node
        pre = node;
        //node切换成temp
        node = temp;
    }
    return node;
}
```

4.删除节点

```
/**
 * 删除所有值为val的节点
 * 注意: 有可能删除头节点,所以需要返回链表头
 */
public static Node removeNode(Node node, Integer val) {
    Node head = node;
    //如果链表前几个值都需要删除,则需找到新链表头
    while (head != null) {
        if (head.value != val) {
            break;
        }
        head = head.next;

    }

    //  a  -> b -> c -> d
    Node pre = head;
    Node cur = head;
    while (cur != null) {
        if (cur.value == val) {
            pre.next = cur.next;
        } else {
            pre = cur;
        }
        cur = cur.next;
    }
    return head;

}
```