# 206. 反转链表 + 24. 两两交换链表中的节点

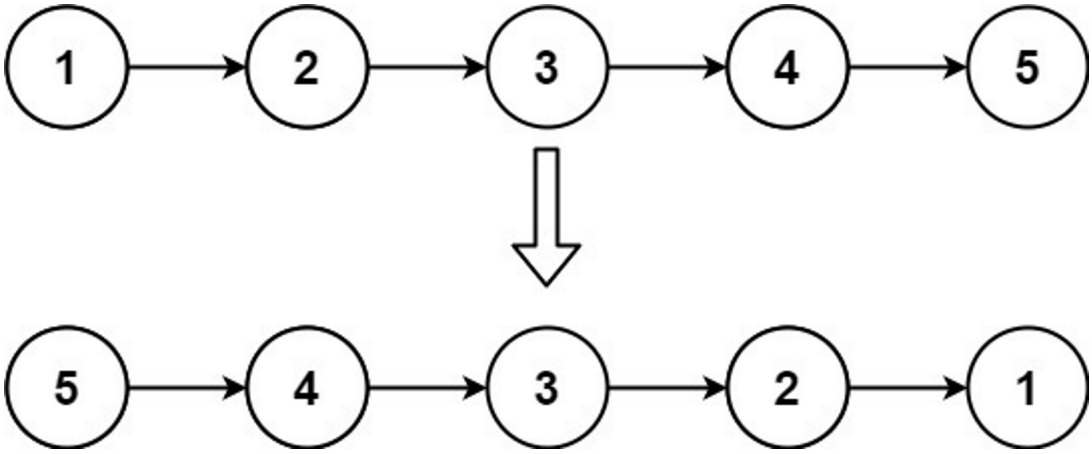
Tuesday, July 5, 2022 9:22 AM

https://leetcode.cn/problems/reverse-linked-list/

简单

给你单链表的头节点 head ，请你反转链表，并返回反转后的链表。

**示例 1：**




**输入：** head = [1,2,3,4,5]

**输出：** [5,4,3,2,1]

解法：pre指针开始为空，cur指针指向head，temp指针指向null

while cur不为空：先temp保存cur.next的位置，再将cur的next指向pre，再移动pre至cur，再将cur指向temp也就是cur之前的next位置。

最后返回pre。

```
class Solution {
    public ListNode reverseList(ListNode head) {
        if(head==null) return null;
        ListNode pre=null;
        ListNode cur=head;
        ListNode temp=null;
        while(cur!=null){
            temp=cur.next;
            cur.next=pre;
            pre=cur;
            cur=temp;
        }
        return pre;
    }
}
```

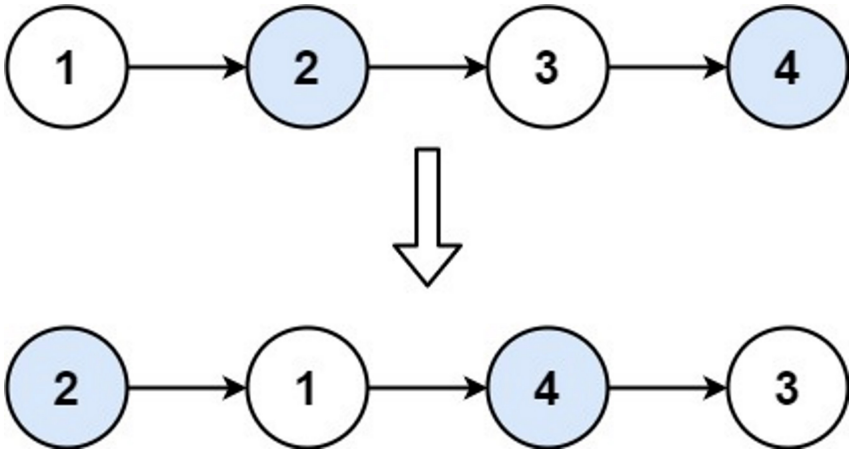
24. 两两交换链表中的节点

https://leetcode.cn/problems/swap-nodes-in-pairs/

中等

给你一个链表，两两交换其中相邻的节点，并返回交换后链表的头节点。你必须在不修改节点内部的值的情况下完成本题（即，只能进行节点交换）。

示例 1：




输入：head = [1,2,3,4]

输出：[2,1,4,3]

解法： 注意： "="：节点变量赋值/节点指针指向； ".next"：节点next指针的指向

```
class Solution {
  public ListNode swapPairs(ListNode head) {
      // 结果节点，作为虚拟头部，最后返回res.next作为结果。
    ListNode res = new ListNode(0);
    res.next = head;
    ListNode pre = res;
    ListNode cur = head;
    while (pre.next != null && pre.next.next != null) {
      ListNode temp = cur.next.next; // 存 next
      pre.next = cur.next;         //第一次 res的next指向2
      cur.next.next = cur;          // 2的next指向1
      cur.next = temp;             // 1的next指向3
      pre = cur;                    //pre指针后移
      cur = cur.next;           // cur指针后移
    }
    return res.next;
  }
}
```