24-两两交换链表中的节点

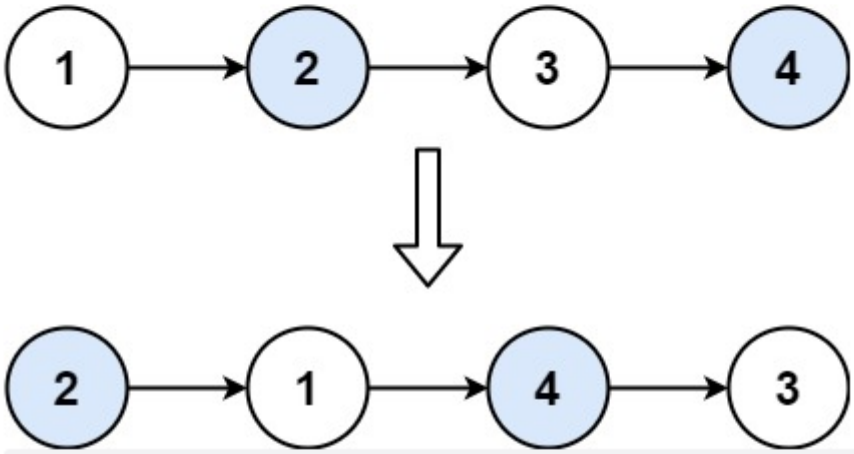
题述

24. 两两交换链表中的节点

难度 中等 1177

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]
```

示例 2：

```
输入：head = []
输出：[]
```

示例 3：

```
输入：head = [1]
输出：[1]
```

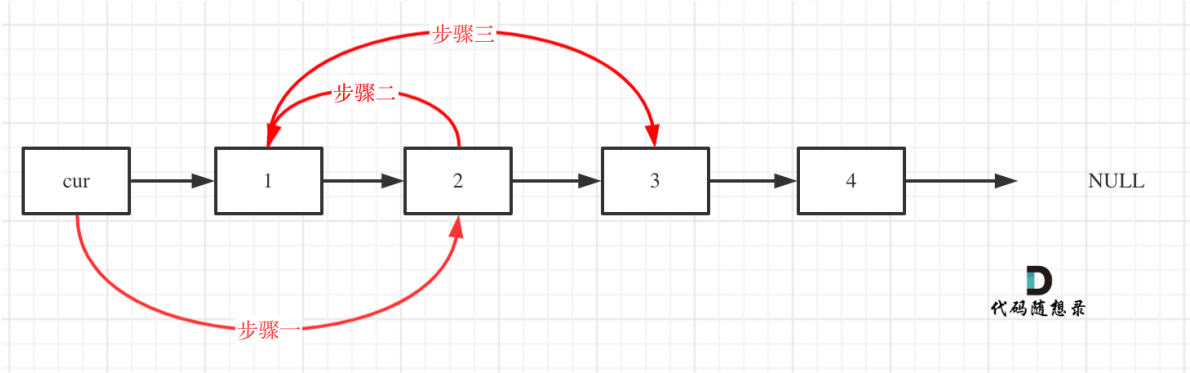
提示：

思考

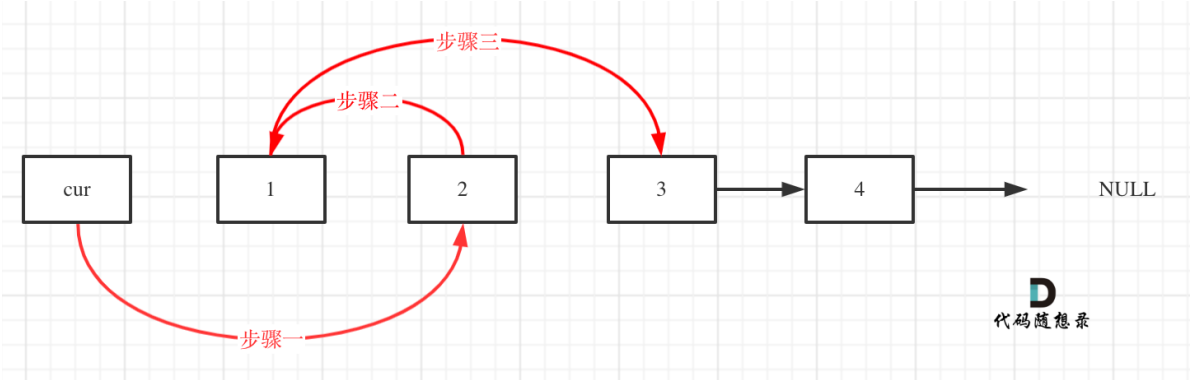
又是一道模拟过程的题

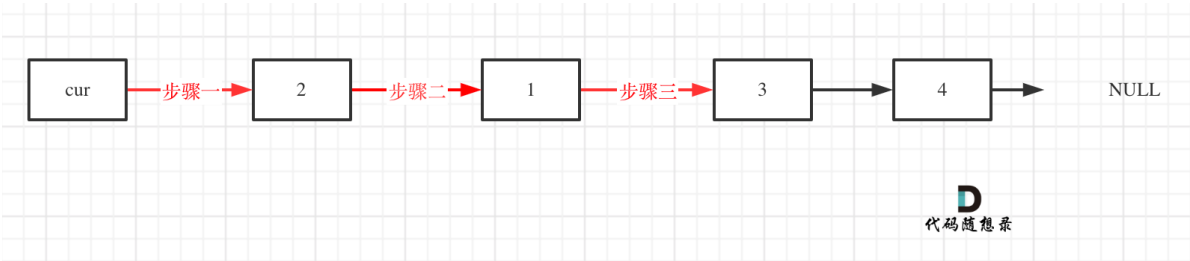
我们使用虚拟头结点法

初始时，cur指向虚拟头结点，然后进行如下三步：



操作之后，链表如下：





题解

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* swapPairs(ListNode* head)
    {
        //两两交换结点 不能只改变数值 要实际交换节点
        //使用虚拟头结点法进行
        ListNode* fakeHead=new ListNode(0); //初始化虚拟头结点
        fakeHead->next=head; //将虚拟头结点指向head 方便后续删除
        ListNode* cur=fakeHead; //工作指针
        while(cur->next!=nullptr && cur->next->next!=nullptr)
        {
            ListNode* temp=cur->next;    //暂存临时结点
            ListNode* temp1=cur->next->next->next;  //暂存临时结点
```

```
            cur->next=cur->next->next;  //STEP1
            cur->next->next=temp;       //STEP2
            cur->next->next->next=temp1;    //STEP3

            cur=cur->next->next;    //cur工作指针移动两步
            //进行下一轮交换
        }
        return fakeHead->next;  //返回实际头结点
    }
};
```

总结

画图思考

写出思路步骤

分解步骤 设置循环条件 逐步骤实现