19-删除链表的倒数第N个结点

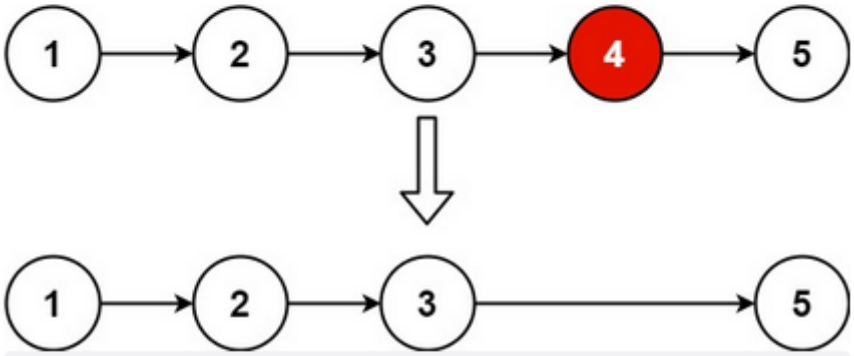
题述

19. 删除链表的倒数第 N 个结点

难度 中等 1726

给你一个链表，删除链表的倒数第 `n` 个结点，并且返回链表的头结点。

示例 1:




```
输入：head = [1,2,3,4,5], n = 2
输出：[1,2,3,5]
```

示例 2:

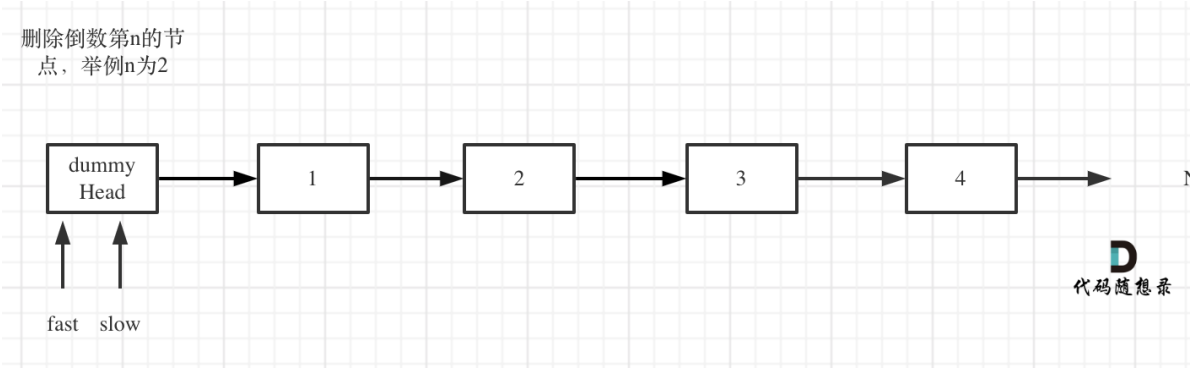
```
输入：head = [1], n = 1
输出：[]
```

示例 3:

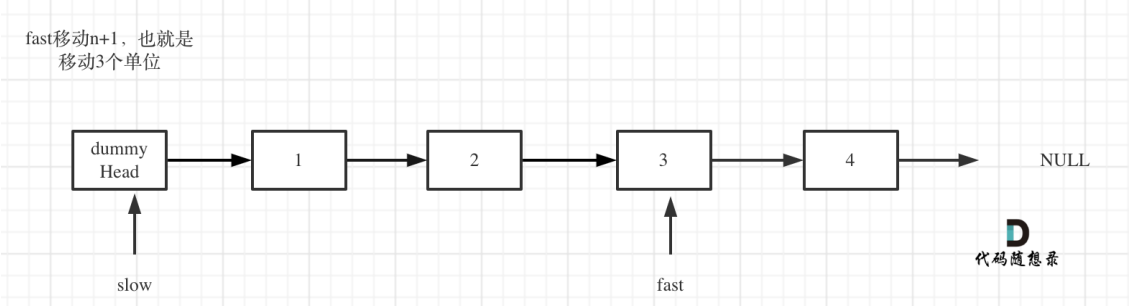
```
输入：head = [1,2], n = 1
输出：[1]
```

思路

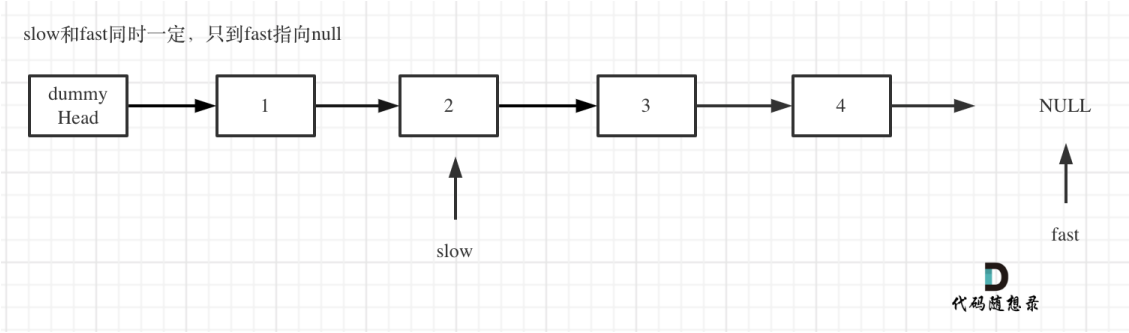
双指针的经典应用，如果要删除倒数第n个节点，让fast移动n步，然后让fast和slow同时移动，直到fast指向链表末尾。删掉slow所指向的节点就可以了。



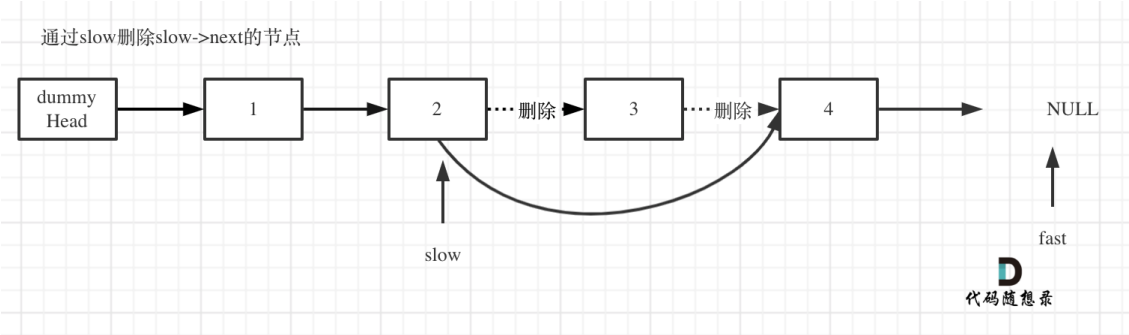
- fast首先走n + 1步 ，为什么是n+1呢，因为只有这样同时移动的时候slow才能指向删除节点的上一个节点（方便做删除操作），如图：



- fast和slow同时移动，直到fast指向末尾，如题：



- 删除slow指向的下一个节点，如图：



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:
```

```cpp
    ListNode* removeNthFromEnd(ListNode* head, int n)
    {
        //双指针
        //虚拟头结点
        //快慢指针

        //初始化虚拟头节点
        ListNode* fakeHead=new ListNode(0);
        fakeHead->next=head;
        ListNode* slow=fakeHead;
        ListNode* fast=fakeHead;

        while(n-- && fast!=nullptr)
        {
            fast=fast->next;    //将fast指针前移n步
        }

        fast=fast->next;    //fast再往前走一步 因为需要让slow指向删除节点的上一个
        while(fast!=nullptr)
        {
            fast=fast->next;
            slow=slow->next;
        }

        slow->next=slow->next->next;
        return fakeHead->next;
    }
};
```

提交结果	执行用时	内存消耗	语言	提交时间	备注
通过	0 ms	10.3 MB	C++	2021/12/28 15:46	添
通过	0 ms	10.3 MB	C++	2021/12/28 15:41	添
通过	0 ms	10.4 MB	C++	2021/12/28 15:41	添

总结

目前我们接触到的链表方法有

双指针法 （快慢指针法）

虚拟头结点法