文本

中度可信度描述已自动生成

解题思路：

不带头节点，本题先考虑正常情况，然后再考虑一开始就是需要删除的情况。

Ps：此时返回的是地址，所以不需要使用二级指针。

代码：

/\*\*

 \* Definition for singly-linked list.

 \* struct ListNode {

 \*     int val;

 \*     struct ListNode \*next;

 \* };

 \*/

struct ListNode\* removeElements(struct ListNode\* head, int val){

    struct ListNode\* cur = head;

    struct ListNode\* prev = NULL;

    while(cur)

    {

        if(cur->val == val)

        {

            struct ListNode\* next = cur->next;

            if(prev == NULL)  //说明cur是头

            {

                free(cur);

                head = next;

                cur = next;

            }

            else

            {

                prev->next = next;

                free(cur);

                cur = next;

            }

        }

        else

        {

            prev = cur;

            cur = cur->next;

        }

    }

    return head;

}

图形用户界面, 应用程序

描述已自动生成

## 第二种思路就是设计一个哨兵位存放头的地址：

/\*\*

 \* Definition for singly-linked list.

 \* struct ListNode {

 \*     int val;

 \*     struct ListNode \*next;

 \* };

 \*/

struct ListNode\* removeElements(struct ListNode\* head, int val){

    struct ListNode\* GuardHead = (struct ListNode\*)malloc(sizeof(struct ListNode));

    GuardHead->next = head;

    struct ListNode\* prev = GuardHead;

    struct ListNode\* cur = head;

    while(cur)

    {

        if(cur->val == val)

        {

            struct ListNode\* next = cur->next;

            prev->next = next;

            free(cur);

            cur = next;

        }

        else

        {

            prev = cur;

            cur = cur->next;

        }

    }

    head = GuardHead->next;

    free(GuardHead);

    return head;

}

两种方法都可以，差别并不是太大。