# 题目

给定一个排序链表，删除所有含有重复数字的节点，只保留原始链表中 没有重复出现 的数字。

**示例 1:**

输入: 1->2->3->3->4->4->5

输出: 1->2->5

**示例 2:**

输入: 1->1->1->2->3

输出: 2->3

# 分析

## 方法一：递归法

class Solution {

public:

ListNode\* deleteDuplicates(ListNode\* head) {

if (head == NULL)

return head;

if (head->next != NULL && head->val == head->next->val) {

//找相同数

while (head->next != NULL && head->val == head->next->val) head = head->next; //忽略所有相同数

return deleteDuplicates(head->next);

//从下一个不同数再开始递归

}

else head->next = deleteDuplicates(head->next);

return head;

}

};

## 方法二：快慢指针

**思路：**

**代码：**

/\*\*

\* Definition for singly-linked list.

\* struct ListNode {

\* int val;

\* ListNode \*next;

\* ListNode(int x) : val(x), next(NULL) {}

\* };

\*/

class Solution {

public:

ListNode\* deleteDuplicates(ListNode\* head) {

ListNode \*dummy = new ListNode(-1),\*slow = dummy,\*fast = head;

dummy->next = head;

while(fast!=NULL&&fast->next!=NULL){

//快指针用于查找相同元素

if(fast->val!=fast->next->val) {

if(slow->next == fast)

slow = fast;//快指针先前没查找到相同数

else

slow->next = fast->next;

//先前查找到相同数，将出现相同数的前一个数连接到相同数后第一个不同数

}

fast = fast->next;

}

if(slow->next != fast) slow->next = fast->next;

//针对类似[1,1]的情况

return dummy->next;

}

};