# 题目

给定一个排序链表，删除所有重复的元素，使得每个元素只出现一次。

**示例 1:**

输入: 1->1->2

输出: 1->2

**示例 2:**

输入: 1->1->2->3->3

输出: 1->2->3

# 分析

## 方法一：排序

## 方法二：快慢指针

**思路：**

删除重复的与判断环形（即是否存在两个相等的节点）是同一类问题，这里可以使用快慢指针的方法。

说明：类似题目203。

**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(INT\_MAX);

dummy->next = head;

ListNode \*p = dummy;

ListNode \*tmp;//用于删除节点使用

while(p->next)

{

if(p->val == p->next->val)

{

tmp = p->next;

p->next = tmp->next;

delete tmp;

//删除节点

}

else

{

p = p->next;

}

}

return dummy->next;

}

};

也可以这样写：

/\*\*

 \* 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\* deleteDuplicates(ListNode\* head) {

        ListNode \*headNode = new ListNode(0);

        headNode->next = head;

        ListNode \*curNode = headNode;

        while(curNode->next)

        {

            if(curNode->val == curNode->next->val)

            {

                ListNode \*tmpNode = curNode->next;

                curNode->next = tmpNode->next;

                delete tmpNode;

            }

            else

            {

                curNode = curNode->next;

            }

        }

        return headNode->next;

    }

};