算法讨论班第 22 期————2 月 29 日

主讲人：李依霖

1.Remove Duplicates from Sorted Array II

Follow up for "Remove Duplicates":
What if duplicates are allowed at most *twice*?
For example,
Given sorted array *nums* = `[1,1,1,2,2,3]`,
Your function should return length = 5, with the first five elements of *nums* being 1, 1, 2, 2 and 3. It doesn't matter what you leave beyond the new length.

思路：题目的含义是每个数字不能出现两次或者两次以上，在原来链表的基础上，只要将出现两次以上的数字去掉，则剩余的链表即为所求链表。

```
class Solution {
public:
    int removeDuplicates(vector<int>& nums) {
        if (nums.size()==0 || nums.size()==1) {
            return nums.size();
        }
        vector<int>::iterator itr = nums.begin();
        while (itr!=nums.end()-2)
        {
            if (*itr==*(itr+1) && *(itr+1)==*(itr+2))
            {
                nums.erase(itr+2);
            }
            else{
                itr++;
            }
        }
        return nums.size();
    }
};
```

2.Remove Duplicates from Sorted List

Given a sorted linked list, delete all duplicates such that each element appear only *once*.

For example,

Given 1->1->2, return 1->2.

Given 1->1->2->3->3, return 1->2->3.

思路：不管数组中有多少个重复的元素，遍历链表只要当前指针指向的元素和它下一个指向的元素相同，则删除下一个元素，否则指针后移。这样就保证了数组中所有的元素只出现一次。

```
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
        ListNode *p = head;
          if (!p) {
            return nullptr;
        }
        while (p && p->next) {
            if (p->val == p->next->val) {
                p->next = p->next->next;
            }else{
                p = p->next;
            }
        }
        return head;
    }
};
```

3.Remove Duplicates from Sorted List II

Given a sorted linked list, delete all nodes that have duplicate numbers, leaving only *distinct* numbers from the original list.

For example,

Given 1->2->3->3->4->4->5, return 1->2->5.

Given 1->1->1->2->3, return 2->3.

思路：

1->2->3->3->4->4->5

题目要求去除出现两次及以上的元素，要删除上述列表的 3，需要保留 3 前面的指针，但是对于 1->1->1->2->3，第一个 1 是头结点，所以为了

统一操作，为每个单链表附加一个头结点，附一个不可能的值：INT16_MAX->1->1->1->2->3。用 p 指针向后查找到第一个不重复的元素为止，然后删除掉中间所有的元素。

```
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
        vector<int> nums;
        if (!head) {
            return nullptr;
        }
        ListNode *preHead = new ListNode(INT16_MAX);
        preHead->next = head;

        ListNode *pre = preHead;
        ListNode *p = head;
        nums.push_back(preHead->val);

        while (p && p->next) {
            if(p->val == p->next->val){
                while(p && p->next && p->val == p->next->val){
                    p = p->next;
                }
                if(p->val!=nums[nums.size()-1]){
                    pre->next = p->next;
                    p = p->next;
                }
            }else{
                pre = p;
                p = p->next;
            }
        }
        return preHead->next;
    }
};
```