

第33题——移除元素

给你一个数组 `nums` 和一个值 `val`，你需要 原地 移除所有数值等于 `val` 的元素，并返回移除后数组的新长度。

不要使用额外的数组空间，你必须仅使用 $O(1)$ 额外空间并 原地 修改输入数组。

元素的顺序可以改变。你不需要考虑数组中超出新长度后面的元素。

说明:

为什么返回数值是整数，但输出的答案是数组呢？

请注意，输入数组是以「引用」方式传递的，这意味着在函数里修改输入数组对于调用者是可见的。

你可以想象内部操作如下:

```
// nums 是以“引用”方式传递的。也就是说，不对实参作任何拷贝 int len =  
removeElement(nums, val);
```

```
// 在函数里修改输入数组对于调用者是可见的。 // 根据你的函数返回的长度, 它会打印出数  
组中 该长度范围内 的所有元素。 for (int i = 0; i < len; i++) { print(nums[i]); }
```

示例1:

输入: `nums = [3,2,2,3]`, `val = 3` 输出: 2, `nums = [2,2]` 解释: 函数应该返回新的长度 2, 并且 `nums` 中的前两个元素均为 2。你不需要考虑数组中超出新长度后面的元素。例如, 函数返回的新长度为 2, 而 `nums = [2,2,3,3]` 或 `nums = [2,2,0,0]`, 也会被视作正确答案。

示例2:

输入: `nums = [0,1,2,2,3,0,4,2]`, `val = 2` 输出: 5, `nums = [0,1,4,0,3]` 解释: 函数应该返回新的长度 5, 并且 `nums` 中的前五个元素为 0, 1, 3, 0, 4。注意这五个元素可为任意顺序。你不需要考虑数组中超出新长度后面的元素。

解题思路:

这道题看起来花里胡哨的,实际上非常简单:

首先这道题不允许拷贝额外数组,那我们可以设定一个索引值`index`,然后我们遍历整个数组,该元素的值不为`val`,那么我们让它覆盖下标为`index`的元素,如果为`val`则遍历下一个元素,这样一来,我们的数组的前`index`个元素一定都是非`val`的,接下来,我们直接重新指定数组大小,然后返回数组大小就ok啦.

```
class Solution {  
public:  
    int removeElement(vector<int>& nums, int val) {  
        .....  
        int index=0;
```

```
for(int i=0;i<nums.size();i++)
{
    if(nums[i]!=val)
    {
        nums[index]=nums[i];
        index++;
    }
}
nums.resize(index);
return nums.size();
}
};
```

作者: sakura7301

链接: <https://leetcode-cn.com/problems/remove-element/solution/yi-wei-fa-by-sakura7301-pu3q/>

来源: 力扣 (LeetCode)

著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。