# 27. 移除元素（Remove Element）

## 题目：

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。

**示例1:**

给定 nums = [3,2,2,3], val = 3,   
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。

**示例 2:**

给定 nums = [0,1,2,2,3,0,4,2], val = 2,   
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。

**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]);   
}

## 解答：

|  |
| --- |
| public static int RemoveElement(int[] nums, int val)  {  //过程：和上题类似p前面的是整理好的数字，第一步i和p都指向第一个数0，i这时指向的不是目标数2，  //就把i的值赋值给p，接着p往后移，也就是说p点是有可能要更新的位置（也是第一个无效的位置），i也接着往后移，  //i现在指向1，还是不等于2，继续把值赋给p并且往后移，接着i继续向后走指向2这时等于目标数，这时p不动，i继续移动  //直到3时，与目标数不一样，就把值赋给p值并且给p赋值  if (nums == null || nums.Length == 0)  {  return 0;  }  int p = 0;  for (int i = 0; i < nums.Length; i++)  {  if (nums[i] != val)  {  nums[p] = nums[i];  p++;  }  }  return p;  } |