026 Remove Duplicates from Sorted Array

2018年3月29日 11:39

Question:

Given a sorted array, remove the duplicates in-place such that each element appear only *once* and return the new length.
Do not allocate extra space for another array, you must do this by **modifying the input array in-place** with O(1) extra memory.
Example:

Given *nums* = [1,1,2],
Your function should return length = 2, with the first two elements of *nums* being 1 and 2 respectively.
It doesn't matter what you leave beyond the new length.

来自 <https://leetcode.com/problems/remove-duplicates-from-sorted-array/description/>
给定一个有序数组，你需要原地删除其中的重复内容，使每个元素只出现一次,并返回新的长度。
不要另外定义一个数组，您必须通过用 O(1) 额外内存原地修改输入的数组来做到这一点。
给定数组: *nums* = [1,1,2],
你的函数应该返回新长度 2, 并且原数组nums的前两个元素必须是1和2
不需要理会新的数组长度后面的元素

Solution for Python3:

```python
class Solution:
    def removeDuplicates(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        c, n = 0, len(nums)
        for i in range(1, n):
            if nums[i] == nums[i-1]:
                c += 1
            else:
                nums[i-c] = nums[i]
        return n - c
#c:重复元素个数
#遍历时只有在遇到不重复元素时才把它放到它本该在的位置
#而这个位置是通过当前位置-重复元素个数算出的。

class Solution:
    def removeDuplicates(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        c, n = 0, len(nums)
        if n == 0:
            return 0
        for i in range(1, n):
            if nums[i] != nums[c]:
```

```
                c += 1
                nums[c] = nums[i]
        return c + 1
```

Solution for C++:

```
class Solution {
public:
    int removeDuplicates(vector<int>& nums) {
        int count = 0, n = nums.size();
        for (int i = 1; i < n; i++) {
            if (nums[i] == nums[i-1]) {
                count++;
            } else {
                nums[i - count] = nums[i];
            }
        }
        return n - count;
    }
};
```

Appendix: 很巧妙，值得推敲。