给定一个排序数组，你需要在原地删除重复出现的元素，使得每个元素只出现一次，返回移除后数组的新长度。

不要使用额外的数组空间，你必须在原地修改输入数组并在使用 O(1) 额外空间的条件下完成。

示例 1:

给定数组 nums = [1,1,2],

函数应该返回新的长度 2, 并且原数组 nums 的前两个元素被修改为 1, 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 = [0,0,1,1,1,2,2,3,3,4],

函数应该返回新的长度 5, 并且原数组 nums 的前五个元素被修改为 0, 1, 2, 3, 4。

你不需要考虑数组中超出新长度后面的元素。

Python：

class Solution:

def removeDuplicates(self, nums):

"""

:type nums: List[int]

:rtype: int

"""

i = 0

j = 0

while i<len(nums):

if(nums[i] != nums[j]):

j = j + 1

nums[j] = nums[i]

i = i + 1

return len(set(nums))

C：

int removeDuplicates(int\* nums, int numsSize){

int j=0;

if (numsSize == 0)

return 0;

for (int i = 0 ; i < numsSize ; i++){

if(nums[i] != nums[j]){

j++;

nums[j] = nums[i];

}

}

return j+1;

}

Java：

class Solution {

public int removeDuplicates(int[] nums) {

int len = nums.length;

int i=0;

int j=1;

while(j<len){

if(nums[i]==nums[j]){

j++;

}

if(j>=len)break;

if(nums[i]!=nums[j]){

i++;

nums[i]=nums[j];

j++;

}

}

return i+1;

}

}