**26. Remove Duplicates from Sorted Array**

Easy

33306213Add to ListShare

Given a sorted array *nums*, remove the duplicates [**in-place**](https://en.wikipedia.org/wiki/In-place_algorithm) such that each element appears only *once* and returns the new length.

Do not allocate extra space for another array, you must do this by **modifying the input array**[**in-place**](https://en.wikipedia.org/wiki/In-place_algorithm) with O(1) extra memory.

**Clarification:**

Confused why the returned value is an integer but your answer is an array?

Note that the input array is passed in by **reference**, which means a modification to the input array will be known to the caller as well.

Internally you can think of this:

// **nums** is passed in by reference. (i.e., without making a copy)

int len = removeDuplicates(nums);

// any modification to **nums** in your function would be known by the caller.

// using the length returned by your function, it prints the first **len** elements.

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

    print(nums[i]);

}

**Example 1:**

**Input:** nums = [1,1,2]

**Output:** 2, nums = [1,2]

**Explanation:** 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 returned length.

**Example 2:**

**Input:** nums = [0,0,1,1,1,2,2,3,3,4]

**Output:** 5, nums = [0,1,2,3,4]

**Explanation:** Your function should return length = **5**, with the first five elements of *nums* being modified to **0**, **1**, **2**, **3**, and **4** respectively. It doesn't matter what values are set beyond the returned length.

**Constraints:**

* 0 <= nums.length <= 3 \* 104
* -104 <= nums[i] <= 104
* nums is sorted in ascending order.