Given a sorted array *nums*, remove the duplicates **in-place** such that duplicates appeared at most *twice* 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 1:

Given *nums* = **[1,1,1,2,2,3]**,

Your function should return length = **5**, with the first five elements of *nums* being **1, 1, 2, 2** and **3** respectively.

It doesn't matter what you leave beyond the returned length.

Example 2:

Given *nums* = **[0,0,1,1,1,1,2,3,3]**,

Your function should return length = **7**, with the first seven elements of *nums* being modified to **0**, **0**, **1**, **1**, **2**, **3** and **3** respectively.

It doesn't matter what values are set beyond the returned length.

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 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]);
}
```