Remove Duplicates from Sorted Array II

Follow up for "Remove Duplicates":
What if duplicates are allowed at most *twice*?

For example,
Given sorted array *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`. It doesn't matter what you leave beyond the new length.

Solution 1

Same simple solution written in several languages. Just go through the numbers and include those in the result that haven't been included twice already.

C++

```
int removeDuplicates(vector<int>& nums) {
    int i = 0;
    for (int n : nums)
        if (i < 2 || n > nums[i-2])
            nums[i++] = n;
    return i;
}
```

Java

```
public int removeDuplicates(int[] nums) {
    int i = 0;
    for (int n : nums)
        if (i < 2 || n > nums[i-2])
            nums[i++] = n;
    return i;
}
```

Python

```
def removeDuplicates(self, nums):
    i = 0
    for n in nums:
        if i < 2 or n > nums[i-2]:
            nums[i] = n
            i += 1
    return i
```

Ruby

```
def remove_duplicates(nums)
    i = 0
    nums.each { |n| nums[(i+=1)-1] = n if i < 2 || n > nums[i-2] }
    i
end
```

written by StefanPochmann original link here

Solution 2

I think both Remove Duplicates from Sorted Array I and II could be solved in a consistent and more general way by allowing the duplicates to appear k times (k = 1 for problem I and k = 2 for problem II). Here is my way: we need a count variable to keep how many times the duplicated element appears, if we encounter a different element, just set counter to 1, if we encounter a duplicated one, we need to check this count, if it is already k, then we need to skip it, otherwise, we can keep this element. The following is the implementation and can pass both OJ:

```
int removeDuplicates(int A[], int n, int k) {

            if (n <= k) return n;

            int i = 1, j = 1;
            int cnt = 1;
            while (j < n) {
                if (A[j] != A[j-1]) {
                    cnt = 1;
                    A[i++] = A[j];
                }
                else {
                    if (cnt < k) {
                        A[i++] = A[j];
                        cnt++;
                    }
                }
                ++j;
            }
            return i;
}
```

For more details, you can also see this post: LeetCode Remove Duplicates from Sorted Array I and II: O(N) Time and O(1) Space

written by tech-wonderland.net original link here

Solution 3

```
int removeDuplicates(vector<int>& nums) {
    int n = nums.size(), count = 0;
    for (int i = 2; i < n; i++)
        if (nums[i] == nums[i - 2 - count]) count++;
        else nums[i - count] = nums[i];
    return n - count;
}
```

written by aserdega original link here