Remove Duplicate Elements From Sorted Array

02 March 2025 12:03

26. Remove Duplicates from Sorted Array

Solved

Easy | Topics | Companies | Hint

Given an integer array `nums` sorted in **non-decreasing order**, remove the duplicates in-place such that each unique element appears only **once**. The **relative order** of the elements should be kept the **same**. Then return *the number of unique elements in* `nums`.

Consider the number of unique elements of `nums` to be `k`, to get accepted, you need to do the following things:

- Change the array `nums` such that the first `k` elements of `nums` contain the unique elements in the order they were present in `nums` initially. The remaining elements of `nums` are not important as well as the size of `nums`.
- Return `k`.

Example 1:

> **Input:** nums = [1,1,2]
> **Output:** 2, nums = [1,2,_]
> **Explanation:** Your function should return k = 2, with the first two elements of nums being 1 and 2 respectively.
> It does not matter what you leave beyond the returned k (hence they are underscores).

Example 2:

> **Input:** nums = [0,0,1,1,1,1,2,2,3,3,4]
> **Output:** 5, nums = [0,1,2,3,4,_,_,_,_,_,_]
> **Explanation:** Your function should return k = 5, with the first five elements of nums being 0, 1, 2, 3, and 4 respectively.
> It does not matter what you leave beyond the returned k (hence they are underscores).

The problem requires you to **remove duplicate elements from a sorted array** while keeping the relative order of unique elements the same. You must do this **in-place**, meaning you cannot use extra space (like another array).

What You Need to Do

1. Modify the input array nums so that the first k elements contain only unique values from nums in the same order.
2. Return k, which represents the number of unique elements.
3. The values beyond k in the array do not matter.

Example 1

- **Input:**

python Copy Edit

```
nums = [1,1,2]
```

- **Step-by-Step Execution:**
  - The array is sorted, so all duplicates are adjacent.
  - Remove the extra `1` and shift elements to the left if needed.
  - The modified array should look like `[1,2,_]`, where `_` means we don't care about the remaining values.
  - Return `k = 2` (since `[1,2]` are the unique values).
- **Output:**

Copy Edit

```
2, nums = [1,2,_]
```

Constraints:

- 1 <= nums.length <= 3 * 104
- -100 <= nums[i] <= 100
- nums is sorted in non-decreasing order.

Code:

```python
def removeDuplicates(nums):
    if not nums:
        return 0  # Edge case: empty array

    # Pointer for the position of the next unique element
    k = 1  # The first element is always unique

    for i in range(1, len(nums)):
        if nums[i] != nums[i - 1]:  # Found a new unique element
            nums[k] = nums[i]  # Place it at index k
            k += 1  # Move pointer for next unique position

    return k  # Number of unique elements
```

Example Walkthrough:

Dry Run

Input:

python Copy Edit

```
nums = [0,0,1,1,1,2,2,3,3,4]
```

Step-by-Step Execution:

i	nums[i]	nums[k] (modifies array)	k (updated)
1	0	-	1
2	1	nums[1] = 1	2
3	1	-	2
4	1	-	2
5	2	nums[2] = 2	3
6	2	-	3
7	3	nums[3] = 3	4
8	3	-	4
9	4	nums[4] = 4	5

Final `nums` (first `k=5` elements matter):

csharp Copy Edit

```
[0,1,2,3,4,_,_,_,_,_]
```