

3. Remove Duplicates from Sorted Array

EASY

■ What You Will Learn

- ✓ The two-pointer technique for in-place modification
- ✓ Why the array being sorted makes this easy
- ✓ How to return the count of unique elements

■ Intuition First

Imagine you're copying a list of names but you only want unique ones. Since the list is alphabetically sorted, duplicates sit right next to each other. You use one pointer to scan and another to write — only writing when you see a new name.

■ Two Pointer Strategy

- i = slow pointer (tracks where to write next unique element)
- j = fast pointer (scans the array)
- If $arr[j] \neq arr[i]$: increment i , copy $arr[j]$ to $arr[i]$
- After loop: first $i+1$ elements are the unique values

■ Visual Trace

Input (sorted with duplicates)

1	1	2	2	3
---	---	---	---	---

```
i=0, j=0: arr[1]=1 == arr[0]=1 → skip j
i=0, j=2: arr[2]=2 != arr[0]=1 → i=1, arr[1]=2
i=1, j=3: arr[3]=2 == arr[1]=2 → skip j
i=1, j=4: arr[4]=3 != arr[1]=2 → i=2, arr[2]=3
```

First 3 elements are unique ✓ — return $i+1 = 3$

1	2	3	2	3
---	---	---	---	---

■ Java Code

```
public static int removeDuplicates(int[] arr) {
    if (arr.length == 0) return 0;
    int i = 0; // Slow pointer – last unique index
```

```
for (int j = 1; j < arr.length; j++) { // Fast pointer
    if (arr[j] != arr[i]) {
        i++; // Move write pointer
        arr[i] = arr[j]; // Write new unique element
    }
}

return i + 1; // Count of unique elements
}
```

■ Complexity

Case	Time	Space	What happens?
All Cases	O(n)	O(1)	Single scan, two pointers — no extra array

■ Common Mistakes

- This ONLY works on sorted arrays — unsorted requires a HashSet
- Return i+1, not i (i is a 0-based index)
- You don't need to clear elements beyond i+1 — problem only cares about first i+1 elements

■ Pro Tip

- This exact technique is LeetCode #26 — memorize it
- For unsorted: use a HashSet to track seen elements

■ Practice Problems

[Easy] Remove duplicates from [0,0,1,1,1,2,2,3,3,4]

[Med] Remove duplicates from unsorted array using extra space

[Med] Allow at most 2 duplicates (LeetCode #80)

■ Quick Revision

- Sorted array only — duplicates are adjacent
- Two pointers: i=write, j=scan
- Write only when arr[j] != arr[i]
- Return i+1, not i

■ $O(n)$ time, $O(1)$ space