Interview Prep Guidelines: Finding Duplicates in an Array

This guide helps you prepare for common interview questions around finding duplicate elements in an array. It covers requirement gathering, clarifying questions to ask, solution strategies, and pseudocode for all 4 patterns. Use this as a structured guideline during preparation.

A) Requirement Collection & Clarifying Questions

- What exactly counts as a duplicate? Unique values vs. repeated indices?
- Expected output order? (sorted, input order, or arbitrary?)
- Should I return elements or just a boolean?
- Space constraints: Is extra space allowed or should it be in-place?
- Can input be modified?
- Element type: Integers only? Strings allowed?
- What to return if array is empty?
- Return type: Array of duplicates, counts, or print output?

B) Quick Clarification & Answers

- Return type should be an array of duplicate elements. Empty array if no duplicates.
- Homogeneous input assumed (integers/strings).
- Default: Return array, do not print.
- In-place vs extra space depends on interviewer's constraints.
- If input is empty, return an empty array.
- Unique duplicate values should be returned (not all repeated indices).

C) Solution Patterns with Pseudocode

1) Sorted Array + Extra Space

```
function findDuplicatesSortedExtra(arr):
    result = []
    for i from 1 to length(arr)-1:
        if arr[i] == arr[i-1] and (i == 1 OR arr[i] != arr[i-2]):
            append arr[i] to result
    return result
```

2) Unsorted Array + Extra Space

```
function findDuplicatesUnsortedExtra(arr):
    seen = empty set
    duplicates = empty set
    result = []
    for each element in arr:
        if element in seen AND element not in duplicates:
            append element to result
            add element to duplicates
        else:
            add element to seen
```

```
    return result
```

3) Sorted Array + In-Place

```
function findDuplicatesSortedInPlace(arr):
    writeIndex = 0
    for i from 1 to length(arr)-1:
        if arr[i] == arr[i-1] and (i == 1 OR arr[i] != arr[i-2]):
            arr[writeIndex] = arr[i]
            writeIndex = writeIndex + 1
    return arr[0..writeIndex-1]
```

4a) Unsorted Array + In-Place (Sort then Scan)

```
function findDuplicatesUnsortedInPlace_Sort(arr):
    sort arr in-place
    writeIndex = 0
    for i from 1 to length(arr)-1:
        if arr[i] == arr[i-1] and (i == 1 OR arr[i] != arr[i-2]):
            arr[writeIndex] = arr[i]
            writeIndex = writeIndex + 1
    return arr[0..writeIndex-1]
```

4b) Unsorted Array + In-Place (Index Marking, requires integers 1..n)

```
function findDuplicatesUnsortedInPlace_IndexMark(arr):
    result = []
    for each element in arr:
        idx = absolute(element) - 1
        if arr[idx] < 0:
            if element not already in result:
                append absolute(element) to result
        else:
            arr[idx] = -arr[idx]
    return result
```

D) Complexity Summary

Pattern 1: O(n), extra O(k) space Pattern 2: O(n), extra O(u) space Pattern 3: O(n), in-place O(1) Pattern 4A: O(n log n), in-place O(1) Pattern 4B: O(n), in-place O(1), requires bounded integer range

E) Tips for Interviews

- ■ Always clarify requirements before coding.
- ■ State your assumptions clearly (input type, constraints).
- ■ Compare time and space complexity of solutions.
- ■ Communicate edge cases (empty, all unique, all same).
- ■ Mention trade-offs between in-place vs extra space.

■ Remember: Interviews are not only about the final solution but also about how you reason through the problem, ask clarifying questions, and present trade-offs.