LeetCode 217 [Contains Duplicate]

Given an array nums, return true if any value appears at least twice in the array, and return false if every element is distinct.

nums = [1,4,3,5,4] → return true

nums = [1,2,3,4] → return false

Brute Force

nums = [(1),2,3,4,4]
[1,(2),3,4,4]
[1,2,(3),4,4]
⋮

Time complexity is $O(n^2)$ which is pretty expensive

Optimal Approach

1) Sorting — nums = [3,5,1,5], sort = [1,3,5,5]

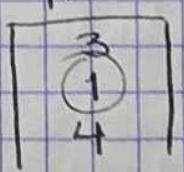
Time complexity is $O(n \log n)$ because of sorting time

2) HashSet

O(1) ↳ when we check if its present or not

nums = [3,1,4,(1),5]

H.S.



Time Complexity = O(n)
Space Complexity = O(n)

Java Solution

```
public boolean containsDuplicate(int[] nums) {
    HashSet<Integer> seenNumbers = new HashSet<>();

    for (int num : nums) {
        if (seenNumbers.contains(num)) {
            return true;
        }
        seenNumbers.add(num);
    }
    return false;
}
```

Kotlin Solution

```
fun containsDuplicate(nums: IntArray): Boolean {
    val seenNumbers = hashSetOf<Int>()

    for (num in nums) {
        if (seenNumbers.contains(num)) {
            return true
        }
        seenNumbers.add(num)
    }

    return false
}
```