Interviewer A Packet - TIPS #5

# Interviewer:

## Behavioral:

Describe a time where you disagreed with one of your team members on a project.

## Question:

<https://leetcode.com/problems/how-many-numbers-are-smaller-than-the-current-number/>

How Many Numbers Are Smaller Than the Current Number?

2401

52

Add to List

Share

Given the array nums, for each nums[i] find out how many numbers in the array are smaller than it. That is, for each nums[i] you have to count the number of valid j's such that j != i and nums[j] < nums[i].

Return the answer in an array.

## Examples:

Example 1:

Input: nums = [8,1,2,2,3]

Output: [4,0,1,1,3]

Explanation:

For nums[0]=8 there exist four smaller numbers than it (1, 2, 2 and 3).

For nums[1]=1 does not exist any smaller number than it.

For nums[2]=2 there exist one smaller number than it (1).

For nums[3]=2 there exist one smaller number than it (1).

For nums[4]=3 there exist three smaller numbers than it (1, 2 and 2).

Example 2:

Input: nums = [6,5,4,8]

Output: [2,1,0,3]

Example 3:

Input: nums = [7,7,7,7]

Output: [0,0,0,0]

Constraints:

* 2 <= nums.length <= 500
* 0 <= nums[i] <= 100

## Follow up Q&A:

## Hint(s):

*Ask if they would like a hint before giving a hint*

## Solution(s): (General concept and time/space complexity)

### Approach #1: Count Array (Java)

public int[] smallerNumbersThanCurrent(int[] nums) {

int[] count = new int[101];

int[] res = new int[nums.length];

for (int i =0; i < nums.length; i++) {

count[nums[i]]++;

}

for (int i = 1 ; i <= 100; i++) {

count[i] += count[i-1];

}

for (int i = 0; i < nums.length; i++) {

if (nums[i] == 0)

res[i] = 0;

else

res[i] = count[nums[i] - 1];

}

return res;

}

Description: Use array to track counts of numbers

Time complexity: O(n)

Space complexity: O(n)

### Approach #2: Use HashMap (C++)

vector<int> smallerNumbersThanCurrent(vector<int>& nums) {

unordered\_map<int, int> map;

vector<int> arr = nums;

sort(arr.begin(), arr.end());

for (int i = 0; i < arr.size(); i++) {

if (map.find(arr[i]) == map.end()) {

map[arr[i]] = i;

}

}

for (int i = 0; i < nums.size(); i++) {

nums[i] = map[nums[i]];

}

return nums;

}

Description: Sort numbers, map them to their index, and build a result array.

Time complexity: O(n log n)

Space complexity: O(n)

### Other questions follow up

*Ask if there is more than 5 minutes remaining when they finish their code and testing.*

# 

# Interviewee:

## Question:

<https://leetcode.com/problems/how-many-numbers-are-smaller-than-the-current-number/>

How Many Numbers Are Smaller Than the Current Number?

2401

52

Add to List

Share

Given the array nums, for each nums[i] find out how many numbers in the array are smaller than it. That is, for each nums[i] you have to count the number of valid j's such that j != i and nums[j] < nums[i].

Return the answer in an array.

## Examples:

Example 1:

Input: nums = [8,1,2,2,3]

Output: [4,0,1,1,3]

Explanation:

For nums[0]=8 there exist four smaller numbers than it (1, 2, 2 and 3).

For nums[1]=1 does not exist any smaller number than it.

For nums[2]=2 there exist one smaller number than it (1).

For nums[3]=2 there exist one smaller number than it (1).

For nums[4]=3 there exist three smaller numbers than it (1, 2 and 2).

Example 2:

Input: nums = [6,5,4,8]

Output: [2,1,0,3]

Example 3:

Input: nums = [7,7,7,7]

Output: [0,0,0,0]

Constraints:

* 2 <= nums.length <= 500
* 0 <= nums[i] <= 100

## Follow up Q&A:

## Code below or on leetcode