Description | Editorial | Solutions (9.8K) | Submissions

# 268. Missing Number

Easy 9.8K 3.2K

Companies

Given an array `nums` containing `n` distinct numbers in the range `[0, n]`, return *the only number in the range that is missing from the array.*

**Example 1:**

```
Input: nums = [3,0,1]
Output: 2
Explanation: n = 3 since there are 3 numbers, so all numbers are in the range [0,3]. 2 is the missing number in the range since it does not appear in nums.
```

**Example 2:**

```
Input: nums = [0,1]
Output: 2
Explanation: n = 2 since there are 2 numbers, so all numbers are in the range [0,2]. 2 is the missing number in the range since it does not appear in nums.
```

**Example 3:**

```
Input: nums = [9,6,4,2,3,5,7,0,1]
Output: 8
Explanation: n = 9 since there are 9 numbers, so all numbers are in the range [0,9]. 8 is the missing number in the range since it does not appear in nums.
```

Java | Auto

```java
class Solution {
    public int missingNumber(int[] nums) {
        int sum_range = 0;
        int sum_org = 0;
        for(int i =0;i<=nums.length;i++){
            sum_range += i;
        }
        for(int j =0;j<nums.length;j++){
            sum_org += nums[j];
        }
        return sum_range - sum_org;

    }
}
```

Testcase | Result

Accepted Runtime: 0 ms

• Case 1 • Case 2 • Case 3

Input

nums =
[3,0,1]

Output

2

Expected

Console Run Submit

Description | Editorial | Solutions (702) | Submissions

## 2305. Fair Distribution of Cookies

Hint

Medium | 1.7K | 73

Companies

You are given an integer array `cookies`, where `cookies[i]` denotes the number of cookies in the $i^{th}$ bag. You are also given an integer `k` that denotes the number of children to distribute **all** the bags of cookies to. All the cookies in the same bag must go to the same child and cannot be split up.

The **unfairness** of a distribution is defined as the **maximum total** cookies obtained by a single child in the distribution.

Return *the **minimum** unfairness of all distributions.*

**Example 1:**

**Input:** cookies = [8,15,10,20,8], k = 2
**Output:** 31
**Explanation:** One optimal distribution is [8,15,8] and [10,20]
- The 1st child receives [8,15,8] which has a total of 8 + 15 + 8 = 31 cookies.
- The 2nd child receives [10,20] which has a total of 10 + 20 = 30 cookies.
The unfairness of the distribution is max(31,30) = 31.
It can be shown that there is no distribution with an unfairness less than 31.

**Example 2:**

**Input:** cookies = [6,1,3,2,2,4,1,2], k = 3
**Output:** 7
**Explanation:** One optimal distribution is [6,1], [3,2,2], and [4,1,2]
- The 1st child receives [6,1] which has a total of 6 + 1 = 7

Java | Auto

```java
class Solution {
    int ans = Integer.MAX_VALUE;

    void helper(int[] cookies, int start, int k, int[] temp) {
        if (start == cookies.length) {
            int max = 0;
            for (int c : temp)
                max = Math.max(max, c);
            ans = Math.min(ans, max);
            return;
        }
        for (int i = 0; i < k; i++) {
            temp[i] += cookies[start];
            helper(cookies, start + 1, k, temp);
            temp[i] -= cookies[start];
        }
    }

    public int distributeCookies(int[] cookies, int k) {
        helper(cookies, 0, k, new int[k]);
```

Testcase | Result

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

cookies =
[8,15,10,20,8]

k =
2

Output

Console | Run | Submit