

238. Product of Array Except Self

Medium

Topics

Companies

Given an integer array `nums`, return an array `answer` such that `answer[i]` is equal to the product of all the elements of `nums` except `nums[i]`.

The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

You must write an algorithm that runs in $O(n)$ time and without using the division operation.

Example 1:

Input: `nums = [1,2,3,4]`

Output: `[24,12,8,6]`

Example 2:

Input: `nums = [-1,1,0,-3,3]`

Output: `[0,0,9,0,0]`

Constraints:

- $2 \leq \text{nums.length} \leq 10^5$
- $-30 \leq \text{nums}[i] \leq 30$
- The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

Follow up: Can you solve the problem in $O(1)$ extra space complexity? (The output array **does not** count as extra space for space complexity analysis.)

Accepted 1.9M

Submissions 2.9M

Acceptance Rate 65.0%

Approach 1: Brute force using division operator

$$T(n) = O(n)$$

$$S(n) = O(1)$$

Approach 2: Brute force without using division operator.

At every index i , calculate

P_1 = product of 0 to $i-1$

P_2 = product of $i+1$ to $n-1$

Then

$$P = P_1 * P_2$$

$$T(n) = O(n^2)$$

$$S(n) = O(1)$$

Approach 3: Using extra space and without division operator

[1 2 3 4]

1. Calculate Prefix array
[1 1 2 6]

2. Calculate suffix array
[24 12 4 1]

3. now

multiply

	[1	1	2	6]
	↑	↑	↑	↑
	[24	12	4	1]

output array = [24 12 8 6]

$$T(n) : O(n) + O(n) + O(n)$$

$$S(n) : \underset{\substack{\downarrow \\ \text{Prefix}}}{O(n)} + \underset{\substack{\downarrow \\ \text{Suffix}}}{O(n)}$$

Approach 4:

As per question the output array is not considered as extra space. So why use 2 extra arrays along with output array.

Let output array be the prefix array. now we can calculate suffix into a variable and then multiply it with respective index element of output array.

[1 2 3 4]

1. Output array i.e. Prefix array

[1 1 2 6]

2. let

```
suffix = 1
for (i: n-1 to 0)
{
    output[i] = output[i] * suffix
    suffix = suffix * nums[i]
}
```

3. return output array

$$T(n) : O(n) + O(n)$$

$$S(n) : O(1)$$

becoz output array
is not considered
as extra space