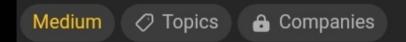
# 238. Product of Array Except Self



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 \ll \text{nums.length} \ll 10^5
```

```
−30 <= nums[i] <= 30</li>
```

• 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 0(1) extra space complexity? (The output array does not count as extra space for space complexity analysis.)

Approach 1: Brute force using division operator

$$\beta(n) = O(n)$$

Approach 2: Brute force without using division operator.

At every index i, calculate
$$P_1 = \text{product of D to i-1}$$

$$P_2 = \text{product of it to n-1}$$
Then
$$P = P_1 * P_2$$

$$f(n) = O(n^2)$$
  
 $f(n) = O(1)$ 

Approach 3: Using xtra space and without division operator

- 1. Calculate Prefix array
- 2. Calculate suffin array [ 24 12 4 1]
- 3. now

  (1 1 2 6)

  multiply

  (24 8 4)

output assay = [24 12 8 6]

î(n) : O(n) +O(n) +0(n)

S(n): O(n) + O(n)Prefix Suffix

As per question the output array is not considered as xtra space. So why use 2 xtra assays along with output assay.

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 assay.

1. Output array i.e. Prefix array [1 | 2 6]

2. let

suffix=1 for ( i: n-1 to 0) output [i] = output[i] \*Seffix suffix= suffix \* nums[i]

3. return output assay

i(n): O(n) + O(n) S(n): O(1) becoz output assay is not considered as xtra space