238. Product of Array Except Self

Solved

Medium | Topics | Companies | Hint

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 <= nums.length <= 10^5`

C++ Auto

```cpp
class Solution {
public:
    vector<int> productExceptSelf(vector<int>& nums) {
        int n = nums.size();
        vector<int> pre(n); // store prefix multiplication
        vector<int> suf(n); // store suffix multiplication
        vector<int> answer(n); // prefix * suffix
        pre=nums; suf=nums;

        pre[0]=1;
        suf[n-1]=1;
        for(int i=1; i<n; i++){
            pre[i]=nums[i-1]*pre[i-1];
           // cout<<pre[i];
        }
        for(int j=n-2;j>=0;j--){
            suf[j]= nums[j+1]*suf[j+1];
        }
        for(int k=0; k<n;k++){
            answer[k]=pre[k]*suf[k];
        }
        return answer;
    }
```