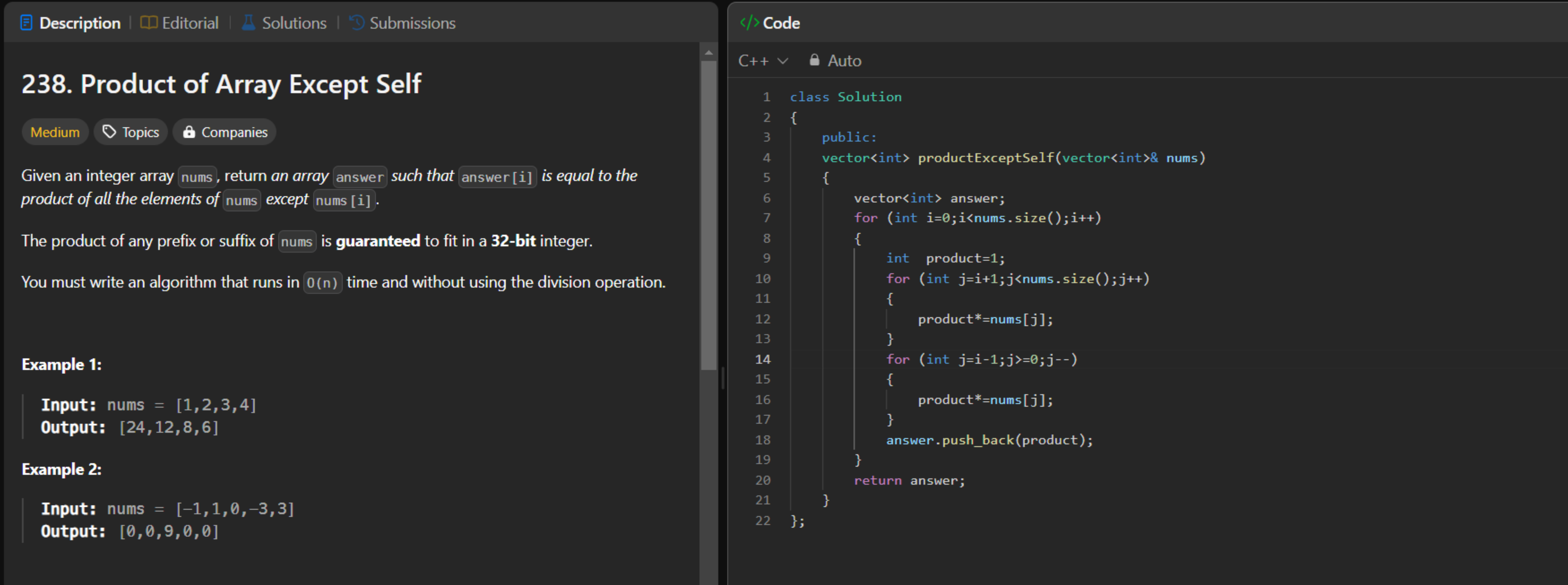


238. Product of Array Except Self

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.

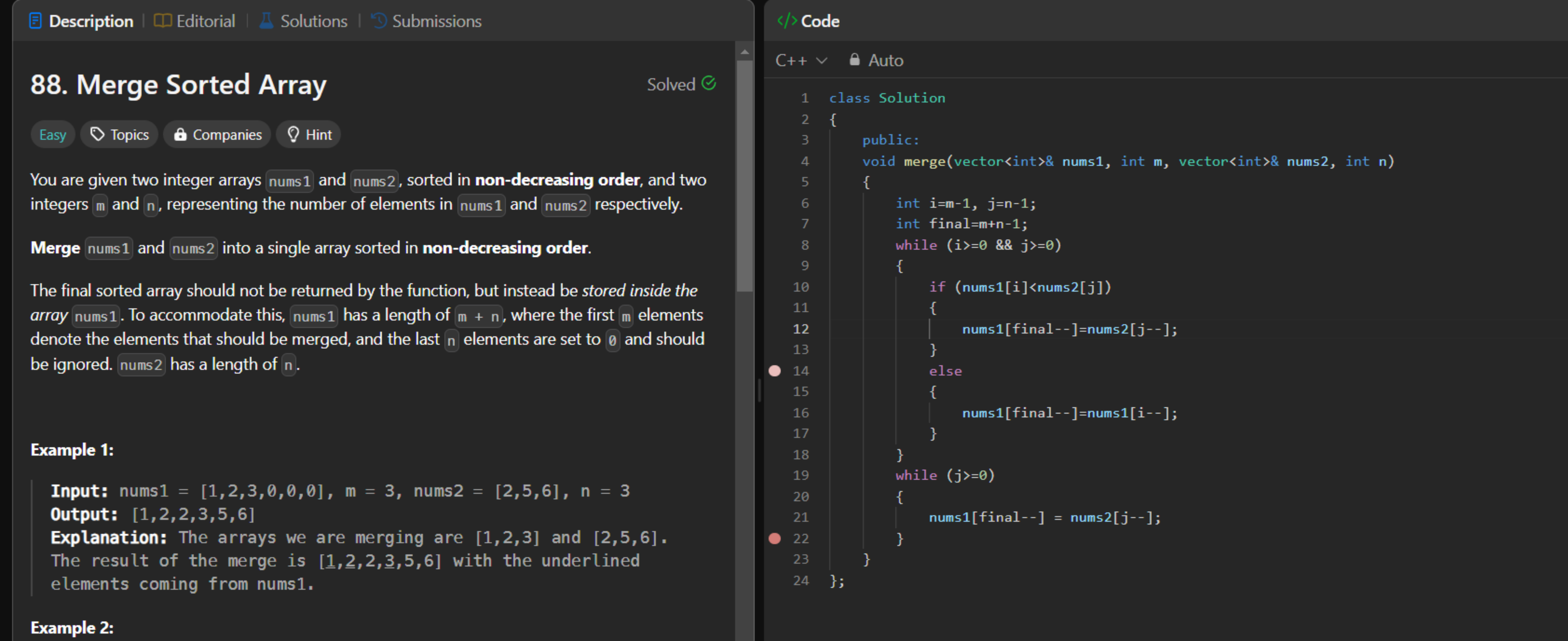
Example 1:

```
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]
```

```cpp
class Solution
{
    public:
    vector<int> productExceptSelf(vector<int>& nums)
    {
        vector<int> answer;
        for (int i=0;i<nums.size();i++)
        {
            int  product=1;
            for (int j=i+1;j<nums.size();j++)
            {
                product*=nums[j];
            }
            for (int j=i-1;j>=0;j--)
            {
                product*=nums[j];
            }
            answer.push_back(product);
        }
        return answer;
    }
};
```
]



88. Merge Sorted Array

Solved ✓

Easy Topics Companies Hint

You are given two integer arrays `nums1` and `nums2`, sorted in **non-decreasing order**, and two integers `m` and `n`, representing the number of elements in `nums1` and `nums2` respectively.

Merge `nums1` and `nums2` into a single array sorted in **non-decreasing order**.

The final sorted array should not be returned by the function, but instead be *stored inside the array* `nums1`. To accommodate this, `nums1` has a length of `m + n`, where the first `m` elements denote the elements that should be merged, and the last `n` elements are set to `0` and should be ignored. `nums2` has a length of `n`.

Example 1:

```
Input: nums1 = [1,2,3,0,0,0], m = 3, nums2 = [2,5,6], n = 3
Output: [1,2,2,3,5,6]
Explanation: The arrays we are merging are [1,2,3] and [2,5,6].
The result of the merge is [1,2,2,3,5,6] with the underlined
elements coming from nums1.
```

Example 2:

```cpp
class Solution
{
    public:
    void merge(vector<int>& nums1, int m, vector<int>& nums2, int n)
    {
        int i=m-1, j=n-1;
        int final=m+n-1;
        while (i>=0 && j>=0)
        {
            if (nums1[i]<nums2[j])
            {
                nums1[final--]=nums2[j--];
            }
            else
            {
                nums1[final--]=nums1[i--];
            }
        }
        while (j>=0)
        {
            nums1[final--] = nums2[j--];
        }
    }
};
```