AGENDA

- Prefix Sum Technique:
    - Product of Array Except Self
- Two Pointer Technique:
    - Pair with a given sum in a sorted array
    - Triplet with a given sum in an array
    - Remove duplicates from a sorted array

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 <= nums.length <= 10^5`
- `-30 <= nums[i] <= 30`

TWO-POINTER TECHNIQUE

Pair with a given sum in a sorted array

Given a sorted array and a target number, check whether there exists a pair with a sum equal to the target.

Example One:
A[] = {1, 2, 5, 6, 10}
target = 8

Output: True

Example Two:
A[] = {1, 2, 5, 6, 10}
target = 9

Output: False

Triplet Sum In Array

Given an array arr of size n and an integer X. Find if there's a triplet in the array which sums up to the given integer X.

Example One:
A[] = {1, 4, 45, 6, 10, 8}
X = 13

Output: True
{1, 4, 8}

Example Two:
A[] = {1, 4, 45, 6, 10, 8}
X = 30

Output: False

Remove Duplicates From Sorted Array

Given a sorted array **A** consisting of duplicate elements.

Your task is to remove all the duplicates and return a sorted array of distinct elements consisting of all distinct elements present in **A**.

But, instead of returning an answer array, you have to **rearrange the given array in-place** such that it resembles what has been described above.
Hence, return a single integer, the index(1-based) till which the answer array would reside in the given array **A**.

Note: This integer is the same as the number of integers remaining inside **A** had we removed all the duplicates.
Look at the example explanations for better understanding.

Example Input

Input 1:

```
A = [1, 1, 2]
```

Input 2:

```
A = [1, 2, 2, 3, 3]
```

Example Output

Output 1:

```
2
```

Output 2:

```
3
```

Example Explanation

Explanation 1:

```
Updated Array: [1, 2, X] after rearranging. Note
that there could be any number in place of x since
we dont need it.
We return 2 here.
```

Explanation 2:

```
Updated Array: [1, 2, 3, X, X] after rearranging
duplicates of 2 and 3.
We return 3 from here.
```