Maximum Product Subarray

152. Maximum Product Subarray

Medium | Topics | Companies

Given an integer array `nums`, find a subarray that has the largest product, and return *the product*.

The test cases are generated so that the answer will fit in a **32-bit** integer.

Example 1:

```
Input: nums = [2,3,-2,4]
Output: 6
Explanation: [2,3] has the largest product 6.
```

Example 2:

```
Input: nums = [-2,0,-1]
Output: 0
Explanation: The result cannot be 2, because [-2,-1] is not a subarray.
```

Constraints:

- $1 \le nums.length \le 2 * 10^4$
- $-10 \le nums[i] \le 10$
- The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

Question is pretty straight forward

* which ever Subarray gives you max product return the product value

Observations

What if,

① all are +Ves,

↳ product of whole array is our ans

② Even negatives, rest all are positives

↳ product of whole array is our ans

③ Odd negatives, rest all are positives

arr: [2, 3, -2, 4]

The possible answer be of left side of '-2' or right side of '-2'

∴ Answer can be from (2,3) (or) 4

2*2 = (6, 4) max ⇒ 6

Ex2 : arr[] = {2, 3, 4, -5, 6, -2, 3, -1, 4, 3}

* if we try to remove one negative ⇒ we can make the count of negatives as even → so no problem will come in product

So, from which side you will remove that negative

1. from front
2. from Back
3. from Middle

① From front

Ex2 : arr[] = {2, 3, 4, -5, 6, -2, 3, -1, 4, 3}

possible arrays are : {2, 3, 4} , {6, -2, 3, -1, 4, 3}

ans = 24 ans = 432

Max({2,3,4}, {6, -2, 3, -1, 4, 3}) = 432

② from Back

Ex2 : arr[] = {2, 3, 4, -5, 6, -2, 3, -1, 4, 3}

possible arrays are : {2, 3, 4, -5, 6, -2, 3} , {4, 3}

Max ans

③ from Middle

Ex2 : arr[] = {2, 2, 4, -5, 6, -2, 3, -1, 4, 3}

possible arrays are : {2, 3, 4, -5, 6} , {3, -1, 4, 3}

negative ans — negative ans

So, no Use

Another example

Ex : arr[] = { 1, 2, -1, -3, 4, -2, 6, 2, -3, -5, -3, 2, 4, 6 }

from middle

arr[] = { 1, 2, -1, -3, 4, -2, 6, 2, -3, -5, -3, 2, 4, 6 }

possible arrays are : {1, 2, -1, -3, 4, -2, 6, 2, -3} {-3, 2, 4, 6}

even negative (OK)

odd negative — no Use

So, Don't prefer going for removing negative from middle

Go with prefix (or) with Suffix

Observation (4)

if array contains zero (0)

Split the array, don't mix 0 with answer / rest everything & move On

Ex: arr[] = { 2, 3, -2, 4 }
0 1 2 3

~from last

i	prefixProduct[i]	SuffixProduct[n-i-1]	Max
0	2	4	~~-∞~~ 4
1	2*3=6	4*-2=-8	4 6
2	6*-2=-12	-8*3=-24	6
3	-12*4=-48	-24*2=-48	6

from prefix

So, from here on if you the problem "Max Product Subarray"

⇒ you think of Prefix or Suffix removal of negatives

Look at prefixproduct & suffixproduct → which gives max

return if

//pseudo Code

```
for (int i=0 ; i<n; i++)

{   if(Prefixproduct ==0)  Prefixproduct = 1;
    if(Suffixproduct==0)   Suffixproduct =1;      } resetting

    prefixProduct  = prefixproduct *arr[i];
    SuffixProduct  = suffixproduct * arr[n-i-1];

    maxVal = max(maxVal, max(prefixproduct, suffixproduct));

}
```