152. Maximum Product Subarray

Given an integer array `nums`, find a contiguous non-empty subarray within the array that has the largest product, and return *the product*.

It is **guaranteed** that the answer will fit in a **32-bit** integer.

A **subarray** is a contiguous subsequence of the array.

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 <= nums.length <= 2 * 10<sup>4</sup>`
- `-10 <= nums[i] <= 10`
- The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

```python
import sys
class Solution:
    def maxProduct(self, nums: List[int]) -> int:
        currMax = nums[0]
        currMin = nums[0]
        ans = nums[0]
        for i in range(1,len(nums)):
            x = max(nums[i],currMax*nums[i],currMin*nums[i])
            y = min(nums[i],currMax*nums[i],currMin*nums[i])
            currMax,currMin=x,y
            ans = max(currMax,ans)
        return ans
```

1. Use an example: [2,-3,4,-8,0]

2. Insights:
   What if the array has just positive numbers including zero?
   A solution of this will maintain max_prod[i] where max_prod[i] is the maximum subarray product ending at i. Then max_prod[i+1] = max(max_prod[i] * nums[i+1], nums[i+1]).
   Now how do we change the solution when we allow negative numbers?
   Imagine that we have both max_prod[i] and min_prod[i] i.e. max prod ending at i and min prod ending at i. Now if we have a negative number at nums[i+1] and if min_prod[i] is negative, then the product of the two will be positive and can potentially be largest product. Key point is to maintain both max_prod and min_prod such that at iteration i, they refer to the max and min prod ending at index i -1.

You have three choices to make at any position in array.

1. You can get maximum product by multiplying the current element with maximum product calculated so far. (might work when current element is positive).
2. You can get maximum product by multiplying the current element with minimum product calculated so far. (might work when current element is negative).
3. Current element might be a starting position for maximum product sub array