3.Hard\12.Maximum_product_subarray.cpp

```cpp
/*QUESTION:

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

Example:

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

APPROACH:

To find the subarray with the largest product, we iterate through the array while keeping track of the current product. We maintain two variables: `ans` to store the maximum product found so far and `prdct` to store the current product. Since negative numbers can change the sign and potentially result in a larger product, we run the loop twice, once from left to right and once from right to left.

CODE:*/

int maxProduct(vector<int>& nums) {
    int ans = INT_MIN;
    int prdct = 1;

    // Iterate from left to right
    for (int i = 0; i < nums.size(); i++) {
        prdct = prdct * nums[i];
        ans = max(ans, prdct);
        if (prdct == 0)
            prdct = 1;
    }

    prdct = 1;

    // Iterate from right to left
    for (int i = nums.size() - 1; i >= 0; i--) {
        prdct = prdct * nums[i];
        ans = max(ans, prdct);
        if (prdct == 0)
            prdct = 1;
    }

    return ans;
}

/*
TIME COMPLEXITY: O(N), where N is the size of the input array.
SPACE COMPLEXITY: O(1).
*/

```