GeeksforGeeks
A computer science portal for geeks

GeeksQuiz

Login

Home Algorithms DS GATE Interview Corner Q&A C C++ Java Books Contribute Ask a Q About

Array Bit Magic C/C++ Articles GFacts Linked List MCQ Misc Output String Tree Graph

Maximum Product Subarray

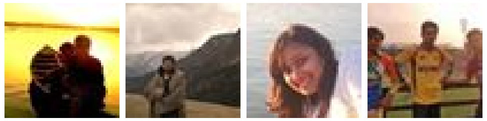
Given an array that contains both positive and negative integers, find the product of the maximum product subarray. Expected Time complexity is O(n) and only O(1) extra space can be used.

Examples:

```
Input: arr[] = {6, -3, -10, 0, 2}
Output:   180  // The subarray is {6, -3, -10}

Input: arr[] = {-1, -3, -10, 0, 60}
Output:   60  // The subarray is {60}

Input: arr[] = {-2, -3, 0, -2, -40}
Output:   80  // The subarray is {-2, -40}
```

The following solution assumes that the given input array always has a positive ouput. The solution works for all cases mentioned above. It doesn't work for arrays like {0, 0, -20, 0}, {0, 0, 0}.. etc. The solution can be easily modified to handle this case.
It is similar to Largest Sum Contiguous Subarray problem. The only thing to note here is, maximum product can also be obtained by minimum (negative) product ending with the previous element multiplied by this element. For example, in array {12, 2, -3, -5, -6, -2}, when we are at element -2, the maximum product is multiplication of, minimum product ending with -6 and -2.

```
#include <stdio.h>

// Utility functions to get minimum of two integers
int min (int x, int y) {return x < y? x : y; }

// Utility functions to get maximum of two integers
```



Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

```
int max (int x, int y) {return x > y? x : y; }
```

```
/* Returns the product of max product subarray.  Assumes that the
   given array always has a subarray with product more than 1 */
int maxSubarrayProduct(int arr[], int n)
{
    // max positive product ending at the current position
    int max_ending_here = 1;

    // min negative product ending at the current position
    int min_ending_here = 1;

    // Initialize overall max product
    int max_so_far = 1;

    /* Traverse throught the array. Following values are maintained af
       max_ending_here is always 1 or some positive product ending wit
       min_ending_here is always 1 or some negative product ending wit
    for (int i = 0; i < n; i++)
    {
        /* If this element is positive, update max_ending_here. Update
           min_ending_here only if min_ending_here is negative */
        if (arr[i] > 0)
        {
            max_ending_here = max_ending_here*arr[i];
            min_ending_here = min (min_ending_here * arr[i], 1);
        }

        /* If this element is 0, then the maximum product cannot
           end here, make both max_ending_here and min_ending_here 0
           Assumption: Output is alway greater than or equal to 1. */
        else if (arr[i] == 0)
        {
            max_ending_here = 1;
            min_ending_here = 1;
        }

        /* If element is negative. This is tricky
           max_ending_here can either be 1 or positive. min_ending_her
           or negative.
           next min_ending_here will always be prev. max_ending_here *
           next max_ending_here will be 1 if prev min_ending_here is 1
           next max_ending_here will be prev min_ending_here * arr[i]
        else
        {
            int temp = max_ending_here;
            max_ending_here = max (min_ending_here * arr[i], 1);
```

Geometric Algorithms

Popular Posts

All permutations of a given string
Memory Layout of C Programs
Understanding “extern” keyword in C
Median of two sorted arrays
Tree traversal without recursion and without stack!
Structure Member Alignment, Padding and Data Packing
Intersection point of two Linked Lists
Lowest Common Ancestor in a BST.
Check if a binary tree is BST or not
Sorted Linked List to Balanced BST

```
            min_ending_here = temp * arr[i];
        }

        // update max_so_far, if needed
        if (max_so_far <  max_ending_here)
          max_so_far  =  max_ending_here;
    }

    return max_so_far;
}

// Driver Program to test above function
int main()
{
    int arr[] = {1, -2, -3, 0, 7, -8, -2};
    int n = sizeof(arr)/sizeof(arr[0]);
    printf("Maximum Sub array product is %d", maxSubarrayProduct(arr, 
    return 0;
}
```

Output:

```
Maximum Sub array product is 112
```

Time Complexity: O(n)
Auxiliary Space: O(1)

This article is compiled by **Dheeraj Jain** and reviewed by GeeksforGeeks team. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Related Tpoics:

- Remove minimum elements from either side such that 2*min becomes more than max
- Divide and Conquer | Set 6 (Search in a Row-wise and Column-wise Sorted 2D Array)
- Bucket Sort
- Kth smallest element in a row-wise and column-wise sorted 2D array | Set 1
- Find the number of zeroes
- Find if there is a subarray with 0 sum
- Divide and Conquer | Set 5 (Strassen's Matrix Multiplication)
- Count all possible groups of size 2 or 3 that have sum as multiple of 3

7 Tweet 0 0

Writing code in comment? Please use **ideone.com** and share the link here.

705 Subscribe

Recent Comments

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 4 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 8 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 32 minutes ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 34 minutes ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 59 minutes ago

newCoder3006 Code without using while loop. We can do it...

Find subarray with given sum · 1 hour ago