Find number of pairs such that x^y > y^x

Given two arrays X[] and Y[] of positive integers, find number of pairs such that **x^y > y^x** where x is an element from X[] and y is an element from Y[].

Examples:

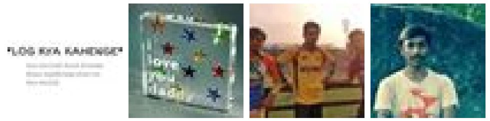
```
Input: X[] = {2, 1, 6}, Y = {1, 5}
Output: 3
// There are total 3 pairs where pow(x, y) is greater than pow(y, x)
// Pairs are (2, 1), (2, 5) and (6, 1)


Input: X[] = {10, 19, 18}, Y[] = {11, 15, 9};
Output: 2
// There are total 2 pairs where pow(x, y) is greater than pow(y, x)
// Pairs are (10, 11) and (10, 15)
```

The **brute force solution** is to consider each element of X[] and Y[], and check whether the given condition satisfies or not. Time Complexity of this solution is **O(m*n)** where m and n are sizes of given arrays.

Following is C++ code based on brute force solution.

```
int countPairsBruteForce(int X[], int Y[], int m, int n)
{
    int ans = 0;
    for (int i = 0; i < m; i++)
       for (int j = 0; j < n; j++)
          if (pow(X[i], Y[j]) > pow(Y[j], X[i]))
```

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

```
                ans++;
    return ans;
}
```

Efficient Solution:
The problem can be solved in **O(nLogn + mLogn)** time. The trick here is, if y > x then x^y > y^x with some exceptions. Following are simple steps based on this trick.

1) Sort array Y[].
2) For every x in X[], find the index idx of smallest number greater than x (also called ceil of x) in Y[] using binary search or we can use the inbuilt function upper_bound() in algorithm library.
3) All the numbers after idx satisfy the relation so just add (n-idx) to the count.

Base Cases and Exceptions:
Following are exceptions for x from X[] and y from Y[]
If x = 0, then the count of pairs for this x is 0.
If x = 1, then the count of pairs for this x is equal to count of 0s in Y[].
The following cases must be handled separately as they don't follow the general rule that x smaller than y means x^y is greater than y^x.
a) x = 2, y = 3 or 4
b) x = 3, y = 2
Note that the case where x = 4 and y = 2 is not there

Following diagram shows all exceptions in tabular form. The value 1 indicates that the corresponding (x, y) form a valid pair.

Y

X

	0	1	2	3	4
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	1
4	1	1	0	0	0

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

Following is C++ implementation. In the following implementation, we pre-process the Y array and count 0, 1, 2, 3 and 4 in it, so that we can handle all exceptions in constant time. The array NoOfY[] is used to store the counts.

```
#include<iostream>
#include<algorithm>
using namespace std;

// This function return count of pairs with x as one element
// of the pair. It mainly looks for all values in Y[] where
// x ^ Y[i] > Y[i] ^ x
int count(int x, int Y[], int n, int NoOfY[])
{
    // If x is 0, then there cannot be any value in Y such that
    // x^Y[i] > Y[i]^x
    if (x == 0) return 0;

    // If x is 1, then the number of pais is equal to number of
    // zeroes in Y[]
    if (x == 1) return NoOfY[0];

    // Find number of elements in Y[] with values greater than x
    // upper_bound() gets address of first greater element in Y[0..n-1
    int* idx = upper_bound(Y, Y + n, x);
    int ans = (Y + n) - idx;

    // If we have reached here, then x must be greater than 1,
    // increase number of pairs for y=0 and y=1
    ans += (NoOfY[0] + NoOfY[1]);

    // Decrease number of pairs for x=2 and (y=4 or y=3)
    if (x == 2)  ans -= (NoOfY[3] + NoOfY[4]);

    // Increase number of pairs for x=3 and y=2
    if (x == 3)  ans += NoOfY[2];

    return ans;
}

// The main function that returns count of pairs (x, y) such that
// x belongs to X[], y belongs to Y[] and x^y > y^x
int countPairs(int X[], int Y[], int m, int n)
{
    // To store counts of 0, 1, 2, 3 and 4 in array Y
    int NoOfY[5] = {0};
    for (int i = 0; i < n; i++)
```

```
        if (Y[i] < 5)
            NoOfY[Y[i]]++;

    // Sort Y[] so that we can do binary search in it
    sort(Y, Y + n);

    int total_pairs = 0; // Initialize result

    // Take every element of X and count pairs with it
    for (int i=0; i<m; i++)
        total_pairs += count(X[i], Y, n, NoOfY);

    return total_pairs;
}

// Driver program to test above functions
int main()
{
    int X[] = {2, 1, 6};
    int Y[] = {1, 5};

    int m = sizeof(X)/sizeof(X[0]);
    int n = sizeof(Y)/sizeof(Y[0]);

    cout << "Total pairs = " << countPairs(X, Y, m, n);

    return 0;
}
```

Output:

```
Total pairs = 3
```

Time Complexity : Let m and n be the sizes of arrays X[] and Y[] respectively. The sort step takes O(nLogn) time. Then every element of X[] is searched in Y[] using binary search. This step takes O(mLogn) time. Overall time complexity is O(nLogn + mLogn).

This article is contributed by **Shubham Mittal**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recent Comments

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 3 minutes ago

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

Root to leaf path sum equal to a given number · 28 minutes ago

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

Count trailing zeroes in factorial of a number · 30 minutes ago

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

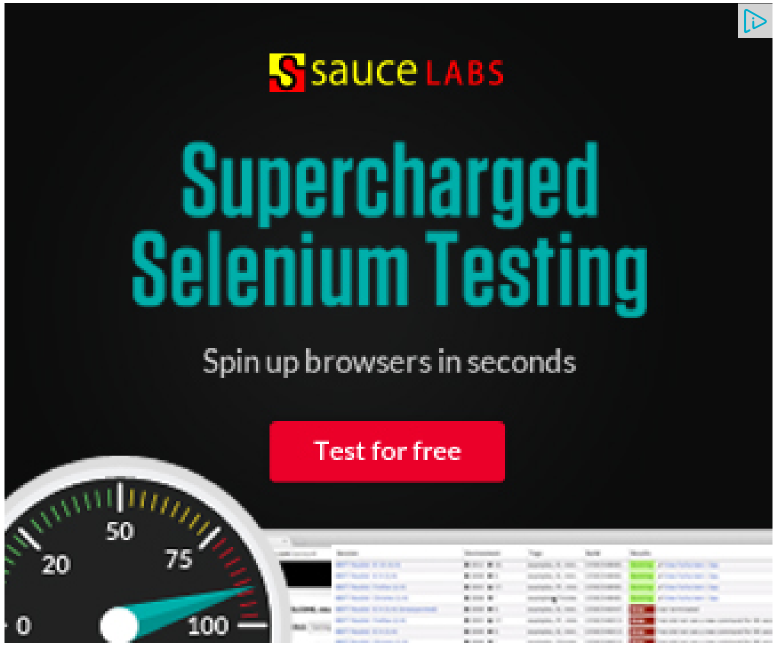
Find subarray with given sum · 54 minutes ago

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

Find subarray with given sum · 1 hour ago

Sanjay Agarwal You can also use the this method:...

Count trailing zeroes in factorial of a number · 1 hour ago



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

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

15 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

zzer • a month ago

excellent for Chaudhary.
step1: sort the array Y
step2: for every element v in X:
if v==1
continue;
if v== 2
count += size of Y
else
if 1 exists in Y:
count+=1
count += num of elements in Y which is bigger than v: binary search---logm
time: O(mlogm+nlogm),
or else we can sort X and handle every element in Y time is O(nlogn+mlogn)

• Reply • Share ›

Ankit Chaudhary • 4 months ago

x^y = y^x
take natural log on both sides.
=> yln(x)=xln(y)
=> ln(x)/x = ln(y)/y

now differentiate ln(x)/x wrt x and compare it with zero.

=> d/dx(ln(x)/x)
=> 1/x^2 - ln(x)/x^2
=> (1-ln(x))/x^2

for all real x, x^2 >=0
therefore : (1-ln(x)) >=0 for x<=e (~ 2.71)

1-ln(x) < 0 for x>e
So ln(x)/x is increasing in range <=e, i.e. for integers, its increasing for 1,2 and decreasing else where.

Following are the cases :

see more

3 ^ | v • Reply • Share ›

Krishna ➦ Ankit Chaudhary • 4 months ago
Classic :), absolutely loved it :)

^ | v • Reply • Share ›

Guest ➦ Krishna • 4 months ago
Sound Mathematical Explanation!!!

^ | v • Reply • Share ›

New • 7 months ago
why do you have to decrease the number of pairs for

if (x == 2) ans -= (NoOfY[3] + NoOfY[4]);

The ans can be kept as it is. There is no need to decrease

^ | v • Reply • Share ›

lovey • 7 months ago
we can also assume one exception that if their is a '1' in an array Y, then for e
with '1' from Y.
i.e, (x1,1),(x2,1)..... and so on..

1 ^ | v • Reply • Share ›

new1 • 7 months ago

http://www.codechef.com/proble...

I think this is similar to the codechef question for OCT'13

5 ^ | v • Reply • Share ›

guest • 7 months ago

@Author ,Geeksforgeeks

Please mention following case in explanation of exception

If x is >1 ,then add number of 1's and 0's in Y,
ans += (NoOfY[0] + NoOfY[1]);

It is difficult map as its missing in exception..

1 ^ | v • Reply • Share ›

Tutulive ➦ guest • 7 months ago

Hi,
The author is adding the number of 0s and 1s to the final answer beca
the elements greater than x and assign this value to answer. But , ther
satisfy the required condition (x^y > y^x) . These extra elements are 0 a
3^1 > 1^3 . For this reason , the author is adding number of 0s and 1s t

^ | v • Reply • Share ›

guest ➦ Tutulive • 7 months ago

Yeah..I got it..But i am asking them to add the same in the post
others:)

^ | v • Reply • Share ›

nikita • 7 months ago

hello...can u pls xplain... how u narrowed down to the set of exceptions??

3 ^ | v • Reply • Share ›

blackpearl nikita • 7 months ago

The above question can be converted to (logx/x) > (logy/y) after taking

If you check the graph of (logx/x) vs x, it increases upto somewhere be
From this you can deduce quite easily that for x and y greater than 3, if
else vice versa.

3 • Reply • Share ›

IsAs blackpearl • 6 months ago

if logx/x > logy/y then x^y > y^x

We can compute two arrays corresponding to X[] and Y[] which
NewX[], NewY[]

We can find the pairs using newly computed arrays

Now we can sort one of the arrays i.e say NewY[] and find the u
array. This eliminates the usage of exceptions table which com

2 • Reply • Share ›

e blackpearl • 7 months ago

hello,can someone please explain the following two statements
int* idx = upper_bound(Y, Y + n, x);
int ans = (Y + n) - idx;

• Reply • Share ›

Tutulive e • 7 months ago

Refer to this , http://www.cplusplus.com/refer...

• Reply • Share ›