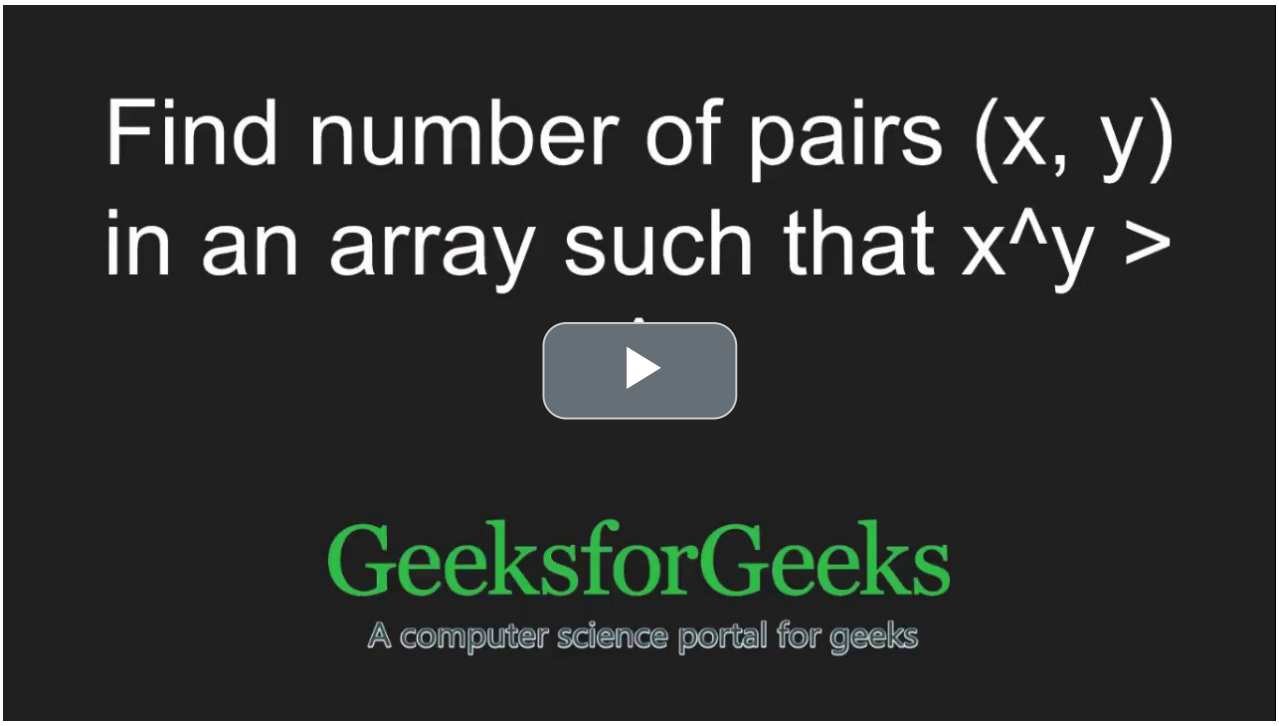
Find number of pair… that x^y > y^x

Difficulty Level : Hard • Last Updated : 13 Aug, 2022

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

Examples:



Array Matrix Strings Hashing Linked List Stack Queue Binary Tree Binary Search Tr

Input: *X[] = {2, 1, 6}, Y = {1, 5}*

Output: *3*

Explanation: *There are total 3 pairs where pow(x, y) is greater than pow(y, x)*
Pairs are (2, 1), (2, 5) and (6, 1)

```
long long countPairsBruteForce(long long X[], long long Y[],
                               long long m, long long n)
{
    long long 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]))
                ans++;
    return ans;
}
```

Java

```
public static long countPairsBruteForce(long X[], long Y[],
                                        int m, int n)
{
    long ans = 0;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            if (Math.pow(X[i], Y[j]) > Math.pow(Y[j], X[i]))
                ans++;
    return ans;
}
```

Python3

```
def countPairsBruteForce(X, Y, m, n):
    ans = 0
    for i in range(m):
```

```
public static int countPairsBrut

{
    int ans = 0;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            if (Math.Pow(X[i], Y[j]) > Math.Pow(Y[j], X[i]))
                ans++;

    return ans;
}
```

Javascript

```
function countPairsBruteForce(X, Y, m, n){
    let ans = 0;
    for(let i=0; i<m; i++ ){
        for(let j=0;j<n;j++){
            if ((Math.pow(X[i], Y[j]) > Math.pow(Y[j], X[i]))){
                ans += 1;
             }
        }
    }
    return ans;
}
```

Time Complexity: O(M*N) where **M** and **N** are sizes of given arrays.

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.

- Sort array Y[].
- For every x in X[], find the index idx of the 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.

- If x = 1, then the count of pairs for t
- If x>1, then we also need to add co
- x smaller than y means x^y is great
  1. x = 2, y = 3 or 4
  2. 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.

	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

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.

Below is the implementation of the above approach:

```
// C++ program to finds the number of pairs (x, y)
// in an array such that x^y > y^x
```

```
{
    // If x is 0, then there can
    // that x^Y[i] > Y[i]^x
    if (x == 0)
        return 0;

    // If x is 1, then the numbe
    // 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;
}

// Function to return 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
```

```
// Driver program
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;
}
```

Java

```
// Java program to finds number of pairs (x, y)
// in an array such that x^y > y^x

import java.util.Arrays;

class Test {
    // Function to 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
    static 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 pairs 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 getting upperbound of x with
        // binary search
        int idx = Arrays.binarySearch(Y, x);
        int ans;
        if (idx < 0) {
```

```
            ans = Y.length - idx
        }

        // If we have reached he
        // than 1, increase numb
        ans += (NoOfY[0] + NoOfY

        // 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;
    }

    // Function to returns count of pairs (x, y) such that
    // x belongs to X[], y belongs to Y[] and x^y > y^x
    static long countPairs(int X[], int Y[], int m, int n)
    {
        // To store counts of 0, 1, 2, 3 and 4 in array Y
        int NoOfY[] = new int[5];
        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
        Arrays.sort(Y);

        long 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 method
    public static void main(String args[])
    {
        int X[] = { 2, 1, 6 };
        int Y[] = { 1, 5 };
```

Python3

```
# Python3 program to find the nu
# of pairs (x, y) in an array
# such that x^y > y^x
import bisect

# Function to 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


def count(x, Y, n, 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 pairs
    # 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, bisect.bisect_right
    # gets address of first greater element
    # in Y[0..n-1]
    idx = bisect.bisect_right(Y, x)
    ans = 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:
```

```
def count_pairs(X, Y, m, n):

    # To store counts of 0, 1, 2
    # and 4 in array Y
    NoOfY = [0] * 5
    for i in range(n):
        if Y[i] < 5:
            NoOfY[Y[i]] += 1

    # Sort Y so that we can do binary search in it
    Y.sort()
    total_pairs = 0  # Initialize result

    # Take every element of X and
    # count pairs with it
    for x in X:
        total_pairs += count(x, Y, n, NoOfY)

    return total_pairs


# Driver Code
if __name__ == '__main__':

    X = [2, 1, 6]
    Y = [1, 5]
    print("Total pairs = ",
          count_pairs(X, Y, len(X), len(Y)))

# This code is contributed by shaswatd673
```

C#

```
// C# program to finds number of pairs (x, y)
// in an array such that x^y > y^x
using System;

class GFG {

    // Function to return count of pairs
    // with x as one element of the pair.
    // It mainly looks for all values in Y[]
```

```
        // If x is 1, then the n
        // is equal to number of
        if (x == 1)
            return NoOfY[0];

        // Find number of elemen
        // values greater than x getting
        // upperbound of x with binary search
        int idx = Array.BinarySearch(Y, x);
        int ans;
        if (idx < 0) {
            idx = Math.Abs(idx + 1);
            ans = Y.Length - idx;
        }

        else {
            while (idx < n && Y[idx] == x) {
                idx++;
            }
            ans = Y.Length - 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;
    }

    // Function to that returns count
    // of pairs (x, y) such that x belongs
    // to X[], y belongs to Y[] and x^y > y^x
    static int countPairs(int[] X, int[] Y, int m, int n)
    {
        // To store counts of 0, 1, 2, 3 and 4 in array Y
```

```
        int total_pairs = 0; //

        // Take every element of
        for (int i = 0; i < m; i
            total_pairs += count

        return total_pairs;
    }

    // Driver method
    public static void Main()
    {
        int[] X = { 2, 1, 6 };
        int[] Y = { 1, 5 };

        Console.Write(
            "Total pairs = "
            + countPairs(X, Y, X.Length, Y.Length));
    }
}

// This code is contributed by Sam007
```

Javascript

```
<script>

// JavaScript program to finds number of pairs (x, y)
// in an array such that x^y > y^x

// Iterative function to implement Binary Search
 function  binarySearch(arr, x) {

    let start=0, end=arr.length-1;

    // Iterate while start not meets end
    while (start<=end){

        // Find the mid index
        let mid=parseInt((start + end)/2);

        // If element is present at mid, return True
        if (arr[mid]===x) return mid;
```

```
    return -1;
}
    // Function to return count
    // element of the pair. It m
    // in Y where x ^ Y[i] > Y[i
    function count(x , Y , n , N
        // 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 pairs 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 getting upperbound of x with
        // binary search
        var idx = binarySearch(Y, x);
        var ans;
        if (idx < 0) {
            idx = Math.abs(idx + 1);
            ans = Y.length - idx;
        } else {
            while (idx < n && Y[idx] == x) {
                idx++;
            }
            ans = Y.length - 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;
    }
```

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

        // Sort Y so that we can
        Y.sort((a,b)=>a-b);

        var total_pairs = 0; //

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

        return total_pairs;
    }

    // Driver method

        var X = [ 2, 1, 6 ];
        var Y = [ 1, 5 ];

        document.write("Total pairs = " +
        countPairs(X, Y, X.length, Y.length));

// This code contributed by umadevi9616

</script>
```

Output

```
Total pairs = 3
```

Time Complexity: O(nLogn + mLogn), where m and n are 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.
Auxiliary Space: O(1)

https://www.youtube.com/watch?v=chYKJGPNEvg

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

Previous Next

RECOMMENDED ARTICLES

Page : 1 2 3

01 Find number of pairs in an array such that their XOR is 0
12, Jan 18

02 Find the maximum cost of an array of pairs choosing at most K

05 Check if an array of pairs can be sorted by swapping pairs with different first elements
29, Apr 21

19, Jan 22

04 Probability of obtaining pairs two arrays such that element the first array is smaller than of the second array
14, Jun 21

not contain any pair with sum K
12, Jan 21

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Hard

Easy | Normal | Medium | Hard | Expert

Improved By : Devarshi_Singh, shaswatd673, SHUBHAMSINGH10, drexor, RohitOberoi, Rajput-Ji, oldMessi, rohitsingh07052, umadevi9616, simranarora5sos, sachinvinod1904, meetmandhane

Article Tags : FactSet, Sorting

Practice Tags : FactSet, Sorting

Improve Article | Report Issue

Start Your Coding Journey

A-143, 9th Flo
Sector-136, N

feedback@geeksforgeeks.org