

ಭ Premium

Back to Contest (/contest/biweekly-contest-72/)





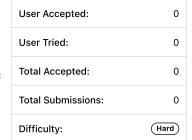
5999. Count Good Triplets in an Array

You are given two 0-indexed arrays nums1 and nums2 of length n, both of which are permutations of [0, 1, ..., n - 1].

My Submissions (/contest/biweekly-contest-72/problems/count-good-triplets-in-an-array/submissions/)

A good triplet is a set of 3 distinct values which are present in increasing order by position both in nums1 and nums 2. In other words, if we consider $pos1_v$ as the index of the value v in nums 1 and $pos2_v$ as the index of the value v in nums2, then a good triplet will be a set (x, y, z) where $0 \le x, y, z \le n - 1$, such that $pos1_x < x$ $pos1_v < pos1_z$ and $pos2_x < pos2_v < pos2_z$.

Return the total number of good triplets.



Example 1:

```
Input: nums1 = [2,0,1,3], nums2 = [0,1,2,3]
Output: 1
Explanation:
There are 4 triplets (x,y,z) such that pos1_x < pos1_y < pos1_z. They are (2,0,1), (2,0,3), (2,1,3), and (0,1,3).
Out of those triplets, only the triplet (0,1,3) satisfies pos2_x < pos2_y < pos2_z. Hence, there is only 1 good triplet.
```

Example 2:

```
Input: nums1 = [4,0,1,3,2], nums2 = [4,1,0,2,3]
Output: 4
Explanation: The 4 good triplets are (4,0,3), (4,0,2), (4,1,3), and (4,1,2).
```

Constraints:

- n == nums1.length == nums2.length
- $3 \le n \le 10^5$
- 0 <= nums1[i], nums2[i] <= n 1
- nums1 and nums2 are permutations of [0, 1, ..., n-1].

```
JavaScript
                                                                                                                        2 *
                                                                                                                    क
1 ▼ function Fenwick(n) {
2
        let tree = Array(n).fill(0);
3
        return { query, update }
4
        function query(i) {
5
            let sum = 0;
 6
            i++;
7
            while (i > 0) {
8
                 sum += tree[i];
9
                 i -= i & -i;
10
            }
            return sum;
11
12
13 •
        function update(i, v) {
14
            i++;
15
            while (i < n) {
16
                 tree[i] += v;
17
                 i += i & -i;
18
            }
19
        }
20
    }
21
    const goodTriplets = (a, b) \Rightarrow \{
22 •
23
        let n = a.length, m = new Map(), res = 0;
24
        for (let i = 0; i < n; i++) m.set(b[i], i);
25
        let fen = new Fenwick(n + 3);
26 •
        for (let i = 0; i < n; i++) {
27
           let pos = m.get(a[i]);
```

```
28     let l = fen.query(pos), r = (n - 1 - pos) - (fen.query(n - 1) - fen.query(pos));
29     res += l * r;
30     fen.update(pos, 1);
31     }
32     return res;
33 };
```

 $\ \square$ Custom Testcase

Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/644649365/) @

More Details > (/submissions/detail/644649365/)

Share your acceptance!

Copyright © 2022 LeetCode

Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Online Interview (/interview/) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)

United States (/region)