SAVEETHA SCHOOL OF ENGINEERING
SIMATS, CHENNAI -602105

CSA0695-DESIGN ANALYSIS OF ALGORITHMS FOR OPEN ADDRESSING TECHNIQUES

Project name : Count of good triplets in an array

SUBMITTED TO

Dr . R. DHANALAKSHMI

SUBMITTED BY

K.HARSHA VARDHAN(192211503)

Count Good Triplets in an Array

This problem involves analyzing two arrays, nums1 and nums2, which are permutations of the same set of numbers. The goal is to identify "good triplets," which are sets of three distinct numbers appearing in the same increasing order within both arrays.

P

Abstract

This project is to determine the number of good triplets in two given permutations of integers. A good triplet is defined as a set of three distinct values found in both arrays in increasing order by their indices.

Introduction

Given two 0-indexed arrays nums1 and nums2, each of length n, where both arrays are permutations of the set [0, 1, ..., n-1], we are interested in counting the number of "good triplets". A triplet (x, y, z) is defined as good if it satisfies the following conditions: 1. Distinct Values: The indices x, y, and z must be distinct. 2. Increasing Order in nums1: The values at these indices must appear in increasing order in nums1. Specifically, if v1, v2, and v3 are the values at indices x, y, and z, respectively, then pos1[v1] < pos1[v2] < pos1[v3] where pos1[v] denotes the index of value v in nums1. 3. Increasing Order in nums2: Similarly, the values at these indices must also appear in increasing order in nums2, which means pos2[v1] < pos2[v2] < pos2[v3] where pos2[v] denotes the index of value v in nums2. The goal is to return the total number of such good triplets

Approach

1 Iteration

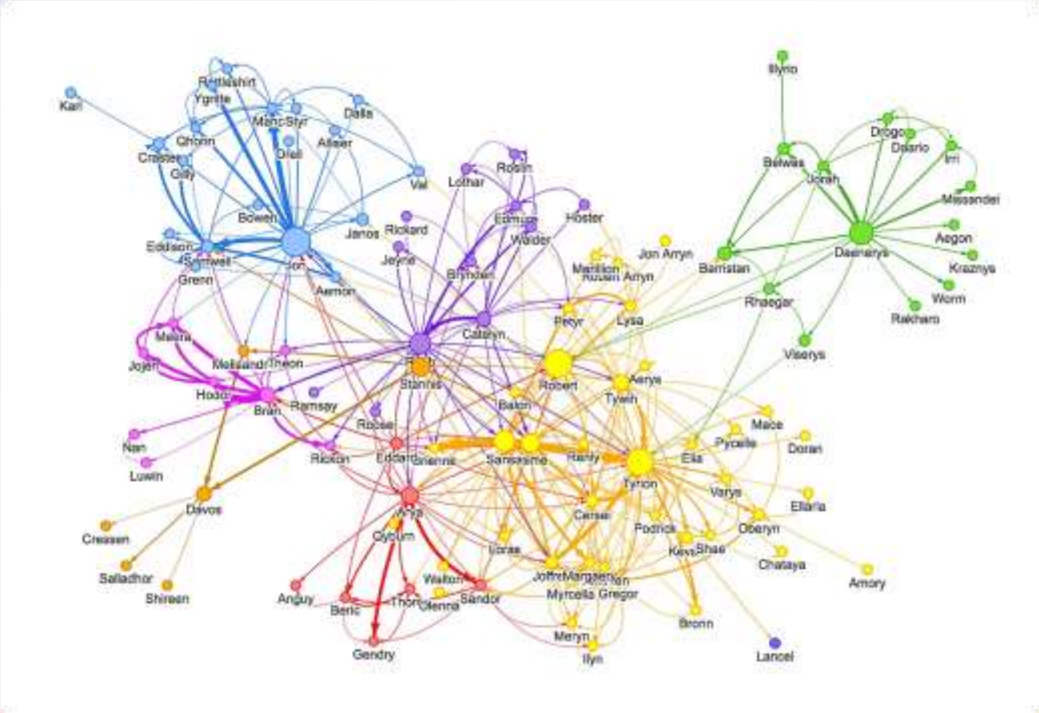
The solution involves iterating through the arrays to identify potential triplets.

2 Comparison

For each triplet, compare its positions in both arrays to determine if the order is consistent.

3 Counting

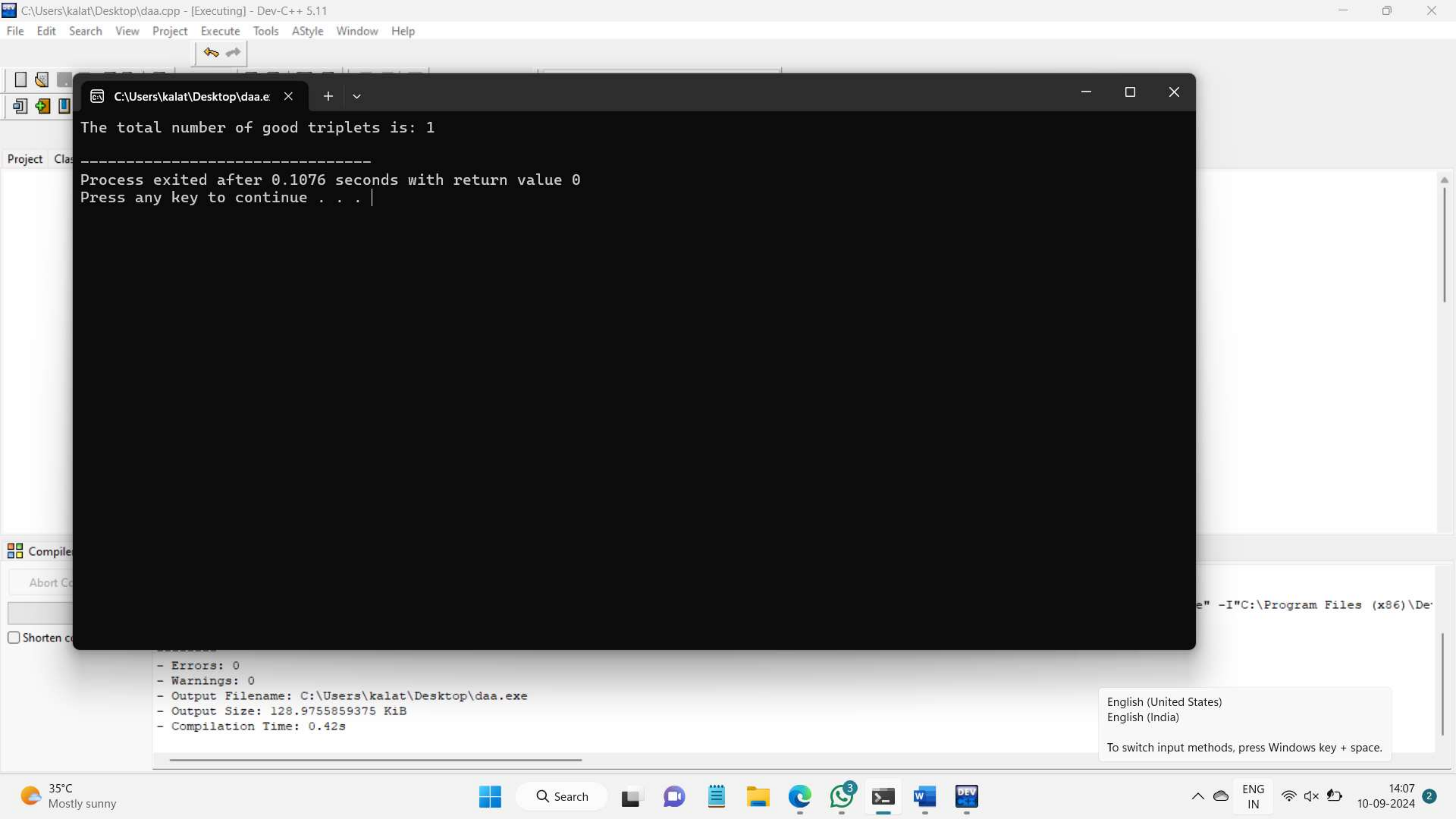
Increment the count of good triplets whenever the order is consistent in both arrays.



Example

nums1	[2, 0, 1, 3]
nums2	[0, 1, 3, 2]

In this example, the triplet (0, 1, 3) is a good triplet because it appears in increasing order in both arrays.



```
The total number of good triplets is: 1

--------------------------------
Process exited after 0.1076 seconds with return value 0
Press any key to continue . . .
```

```
e" -I"C:\Program Files (x86)\De
```

```
--------
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\kalat\Desktop\daa.exe
- Output Size: 128.9755859375 KiB
- Compilation Time: 0.42s
```

TIME COMPLEXITY

- **Best Case**: O(n log n) (This is typically when the data structures used for counting are efficient and perform well.)
- **Average Case**: O(n log n) (Average case assumes that the Fenwick Tree operations are performed in logarithmic time on average.)
- **Worst Case**: O(n log n) (This occurs when the data structure operations are at their maximum complexity, but in practice, the logarithmic factor dominates.)

Future scope:

The feature scope for the "count good triplets" problem involves developing an efficient algorithm to count the number of triplets (x, y, z) in two permutations nums1 and nums2 of length n, such that the positions of these values in both arrays are in strictly increasing order. The solution should map positions of elements in both arrays, transform nums2 based on nums1's indices, and then count increasing triplets in the transformed array using data structures like Fenwick Trees or Segment Trees for optimal performance, targeting a time complexity of O(n log n).

conclusion

The final result is the count of good triplets found in the arrays.

1

Triplet Found

The triplet satisfies the order requirement in both arrays.

2

Count Increment

Increase the count by 1.

3

Continue Iteration

Repeat the process for all possible triplets.