Counting Inversions using Set in C++ STL

Inversion Count for an array indicates – how far (or close) the array is from being sorted. If array is already sorted then inversion count is 0. If array is sorted in reverse order that inversion count is the maximum.

```
 Two elements a[i] and a[j] form an inversion if
  a[i] > a[j] and i < j. For simplicity, we may
  assume that all elements are unique.

  Example:
  Input:  arr[] = {8, 4, 2, 1}
  Output: 6
  Given array has six inversions (8,4), (4,2),
  (8,2), (8,1), (4,1), (2,1).
```

We have already discussed below approaches.

1) Naive and Merge Sort based approaches.

2) AVL Tree based approach.

In this post an easy implementation of approach 2 using Set in C++ STL is discussed.

```
1) Create an empty Set in C++ STL (Note that a Set in C++ STL is
   implemented using Self-Balancing Binary Search Tree). And insert
   first element of array into the set.

2) Initialize inversion count as 0.

3) Iterate from 1 to n-1 and do following for every element in arr[i]
     a) Insert arr[i] into the set.
     b) Find the first element greater than arr[i] in set
        using upper_bound() defined Set STL.
     c) Find distance of above found element from last element in set
        and add this distance to inversion count.

4) Return inversion count.
```

```
// A STL Set based approach for inversion count
```

```
#include<bits/stdc++.h>
using namespace std;

// Returns inversion count in arr[0..n-1]
int getInvCount(int arr[],int n)
{
    // Create an empty set and insert first element in it
    set<int> set1;
    set1.insert(arr[0]);

    int invcount = 0; // Initialize result

    set<int>::iterator itset1; // Iterator for the set

    // Traverse all elements starting from second
    for (int i=1; i<n; i++)
    {
        // Insert arr[i] in set (Note that set maintains
        // sorted order)
        set1.insert(arr[i]);

        // Set the iterator to first greater element than arr[i]
        // in set (Note that set stores arr[0],.., arr[i-1]
        itset1 = set1.upper_bound(arr[i]);

        // Get distance of first greater element from end
        // and this distance is count of greater elements
        // on left side of arr[i]
        invcount += distance(itset1, set1.end());
    }

    return invcount;
}

// Driver program to test above
int main()
{
    int arr[] = {8, 4, 2, 1};
    int n = sizeof(arr)/sizeof(int);
    cout << "Number of inversions count are : "
         << getInvCount(arr,n);
    return 0;
}
```

Run on IDE

Output:

```
Number of inversions count are : 6
```

Note that the worst case time complexity of above implementation is O(n^2) as distance function in STL takes O(n) time worst case, but this implementation is much simpler than other implementations and would take much less time than Niave method on average.

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

Practice Similar Questions On: STL CPP

C++ STL

Improve this Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Recommended Posts:

Set in C++ Standard Template Library (STL)
Commonly Asked C++ Interview Questions | Set 2
Pair in C++ Standard Template Library (STL)
Insertion and Deletion in STL Set C++
How to implement Min Heap using STL?
alignof operator in C++
File Handling through C++ Classes
tellg() function in C++ with example
Sorting of Vector of Tuple in C++ (Descending Order)
Sorting of Vector of Tuple in C++ (Ascending Order)

Logged in as **harpreet_singh**(Logout)

3 Average Difficulty : **3/5.0**
Based on **15** vote(s)

Basic Easy Medium Hard Expert

□ Add to TODO List

□ Mark as DONE

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

Share this post!

A computer science portal for geeks

710-B, Advant Navis Business Park,
Sector-142, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY
About Us
Careers
Privacy Policy
Contact Us

LEARN
Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE
Company-wise
Topic-wise
Contests
Subjective Questions

CONTRIBUTE
Write an Article
Write Interview Experience
Internships
Videos