Count Inversions

Difficulty: **Medium** Accuracy: **16.93%** Submissions: **707K+** Points: **4**

Given an array of integers **arr[]**. You have to find the **Inversion Count** of the array.

Note : Inversion count is the number of pairs of elements (i, j) such that i < j and arr[i] > arr[j].

Examples:

Input: arr[] = [2, 4, 1, 3, 5]
Output: 3
Explanation: The sequence 2, 4, 1, 3, 5 has three inversions (2, 1), (4, 1), (4, 3).

Input: arr[] = [2, 3, 4, 5, 6]
Output: 0
Explanation: As the sequence is already sorted so there is no inversion count.

C++ (12) Start Timer

```cpp
class Solution {
  public:
    int merge (vector <int> &arr , int l, int m , int r)
    {
        int x=m-l+1;
        int y=r-m;

        vector <int> left(x) , right(y);
        for(int i=0;i<x;i++){left[i]=arr[i+l];}
        for(int j=0;j<y;j++){right[j]=arr[m+1+j];}

        int i=0,j=0,k=l,count=0;
        while(i<x&&j<y)
        {
            if(left[i]<=right[j]){arr[k++]=left[i++];}
            else{
                arr[k++]=right[j++];
                count=count+x-i;
            }

        }
        while(i<x){arr[k++]=left[i++];}
        while(j<y){arr[k++]=right[j++];}
        return count;
    }
    int merge_sort(vector <int> &arr ,int l , int r)
    {
        int count=0;
```

Custom Input Compile & Run Sub

Overlapping Intervals

Difficulty: **Medium** Accuracy: **57.41%** Submissions: **113K+** Points: **4**

Average Time: **20m**

Given an array of Intervals **arr[][]**, where **arr[i] = [start$_i$, end$_i$].** The task is to merge all of the **overlapping Intervals**.

Examples:

Input: arr[][] = [[1, 3], [2, 4], [6, 8], [9, 10]]
Output: [[1, 4], [6, 8], [9, 10]]
Explanation: In the given intervals we have only two overlapping intervals here, [1, 3] and [2, 4] which on merging will become [1, 4]. Therefore we will return [[1, 4], [6, 8], [9, 10]].

Input: arr[][] = [[6, 8], [1, 9], [2, 4], [4, 7]]
Output: [[1, 9]]
Explanation: In the given intervals all the intervals overlap with the interval [1, 9]. Therefore we will return [1, 9].

C++ (12) Start Timer

```cpp
class Solution {
  public:
    vector<vector<int>> mergeOverlap(vector<vector<int>>& ar
        // Code here
        sort(arr.begin(),arr.end());
        vector<vector<int>> ans;

        ans.push_back(arr[0]);

        for(int i=1;i<arr.size();i++)
        {
            vector <int> &last = ans.back();
            vector <int> &current = arr[i];

            if(current[0]<=last[1])
            {
                last[1]=max(last[1],current[1]);
            }
            else{ans.push_back(current);}
        }

        return ans;
    }
};
```