Count Inversions

Difficulty: **Medium** Accuracy: **16.93%** Submissions: **607K+** Points: **4**

Given an array of integers **arr[]**. Find the **Inversion Count** in the array.
Two elements arr[i] and arr[j] form an inversion if **arr[i] > arr[j]** and **i < j**.

> ***Inversion Count***: For an array, inversion count indicates how far (or close) the array is from being sorted. If the array is already sorted then the inversion count is 0.
> If an array is sorted in the reverse order then the inversion count is the maximum.

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

Java (1.8) Start Timer

```
1  // } Driver Code Ends
19
20 class Solution {
21     static int countAndMerge(int[] arr, int l, int m, int r) {
22         int n1 = m - l + 1, n2 = r - m;
23         int[] left = new int[n1];
24         int[] right = new int[n2];
25         for (int i = 0; i < n1; i++)
26             left[i] = arr[i + l];
27         for (int j = 0; j < n2; j++)
28             right[j] = arr[m + 1 + j];
29         int res = 0;
30         int i = 0, j = 0, k = l;
31         while (i < n1 && j < n2) {
32             if (left[i] <= right[j])
33                 arr[k++] = left[i++];
34             else {
35                 arr[k++] = right[j++];
36                 res += (n1 - i);
37             }
38         }
39         while (i < n1)
40             arr[k++] = left[i++];
41         while (j < n2)
42             arr[k++] = right[j++];
43         return res;
44     }
```

```
45      static int countInv(int[] arr, int l, int r) {
46          int res = 0;
47          if (l < r) {
48              int m = (r + l) / 2;
49              res += countInv(arr, l, m);
50              res += countInv(arr, m + 1, r);
51              res += countAndMerge(arr, l, m, r);
52          }
53          return res;
54      }
55      static int inversionCount(int[] arr) {
56          return countInv(arr, 0, arr.length - 1);
57      }
58  }
```