Count Inversion Problem

arr = {6, 3, 5, 2, 7}

1, 1, 1, 1, 1 = $\boxed{5}$

Pairs (arr[i], arr[j])

1. i < j
2. arr[i] > arr[j]

Brute Force Approach

Check all the possible pairs

```
for(i=0 to n){
    for(j=i+1; j<n; j++){
        if(arr[i]>arr[j])
            Invcount++
}}
```

T.C = O(n^2)

$\boxed{\text{ANS}=5}$

Optimal Approach (Merge Sort)

arr = {1, 3, 5, 10, 2, 6, 8, 9}

1	3	5	10

2	6	8	9

1	2	3	5	6	8	9	10

Inversion Counts

if(arr[i] <= arr[j])
    swap = arr[i]
else → arr[i] > arr[j]
    InvCount += (mid-i+1)

InvCount = $\boxed{0+3+1+1+1}$ → ⑥

Changes to be done in the Merge logic
In Merge step
  Use a variable InvCount = 0
  In else part → InvCount += (mid-i+1)
      return InvCount

In the MergeSort — main part.
  ① — LeftCount
  ② — RightCount

$\boxed{\text{Total} = LC + RC + InvCount}$

T.C = $O(n \log n)$
S.C = $O(n)$