**Count inversions in an array**

**Problem Statement:** Given an array of N integers, count the inversion of the array (using merge-sort).

What is an inversion of an array? Definition: for all i & j < size of array, if i < j then you have to find pair (A[i],A[j]) such that A[j] < A[i].

```
from os import *
from sys import *
from collections import *
from math import *
import math

def merge(a,low,mid,high):
    left=low
    right=mid+1
    temp=[]
    cnt=0
    while(left<=mid and right<=high):
        if a[left]<=a[right]:
            temp.append(a[left])
            left+=1
        else:
            temp.append(a[right])
            cnt+=(mid-left+1)
            right+=1

    while(left<=mid):
        temp.append(a[left])
```

```
        left+=1
    while right<=high:
        temp.append(a[right])
        right+=1
    for i in range(low,high+1):
        arr[i]=temp[i-low]
    return cnt

def mergesort(a,low,high):
    cnt=0
    if low>=high:
        return cnt
    mid=math.floor((low+high)/2)
    cnt+=mergesort(a,low,mid)
    cnt+=mergesort(a,mid+1,high)
    cnt+=merge(a,low,mid,high)
    return cnt

def getInversions(arr, n) :
    n=len(arr)
    return mergesort(arr,0,n-1)
```

```
# Taking inpit using fast I/O.
def takeInput() :
    n = int(input())
    arr = list(map(int, stdin.readline().strip().split(" ")))
    return arr, n

# Main.
arr, n = takeInput()
print(getInversions(arr, n))
```