**Report**

**Abstract:** we need to count the number of inversions in a given array. Where inversion means, number of pairs in a given array where sorting is done on those set of pair

**Problem Description:**

* Here we need to count the total number of inversions
* Example: 2,4,1,3,5
* If we perform merge sort on that particular array

2 ,4 --->the sub-array is sorted

1,3,5--->the sub-array is sorted

when two sub-arrays are merged

1 comes before 2--->inversion

1 comes before 4--->inversion

3 comes before 4--->inversion

total number of inversions =3

* So while merging left side part and right side part if right side part enters the main array, we encounter an inversion

**Algorithm:**

Mergesort\_and\_count it(array)

Mid=array.length/2

If(array.length<=1)

Return 0

For i=0 till mid

Left array[i]=array[i]

For j=0 till array.length-mid

Right array[j]=array[mid+j]

Leftside=mergesort\_and\_count it(leftarray)

rightside=mergesort\_and\_count it(rightarray)

count=merge(leftarray,rightarray,array2)

for k=0 till array.length

array[k]=array2[k]

totalcount=leftside+rightside+count

return totalcount

merge(leftsidearray,rightsidearray,array2)

i=0,j=0,count=0

for k=0 till leftsidearray.length+rightsidearray.length

if i<leftsidearray.length && j<rightsidearray.length

if leftsidearray[i]<=rightsidearray[j]

array2[k]=leftsidearray[i]

i=i+1

else

array2[k]=rightsidearray[j]

j=j+1

count=count+leftsidearray.length-i

else

break

if i==leftside.length

for v=j till rightsidearray.length

array2[k]=rightsidearray[v]

k=k+1

else

for v=I till leftsidearray.length

array2[k]=leftsidearray[v]

k=k+1

return count

**Implementation Details**:(implemented in java)

* Since given input is in file, we need to read the entire file and store an array
* To read from file we use

1. File reader
2. Buffered reader

* Implement mergesort algorithm and count the inversion each time where right side part comes before left side part in main array
* Since infinity (sentinel value) is not programmatically implemented. We need to check out for left out values in subarray and place in main array
* For each division and conquer method we perform and calculate inversion count value and sum up for entire array
* Since we are dealing with huge data, inversion count variable is “long” data type