**WEEK : 10 : Data Structure and Algorithm Lab**

**Q1. You have been given an array A of size N.This array contains integers ranging from 1 to 10^9. You need to sort the contents of this array by their value and then print the contents of it.**

**Code :**

|  |
| --- |
| n=int(input()) |
|  | string = [] |
|  | string = input().split(" ") |
|  | string.sort() |
|  | for i in range (0,n): |
|  | print (string[i], end = " ") |

**Q2. Given an array A on size N, you need to find the number of ordered pairs (i,j) such that i < j and A[i] >A[j]**

**Code :**

|  |
| --- |
| #include <iostream> |
|  | #include<bits/stdc++.h> |
|  | using namespace std; |
|  |  |
|  | void merge(int arr[], int l, int m, int r,long int \*c) |
|  | { |
|  | int i, j, k; |
|  | int n1 = m - l + 1; |
|  | int n2 = r - m; |
|  |  |
|  | /\* create temp arrays \*/ |
|  | int L[n1], R[n2]; |
|  |  |
|  | /\* Copy data to temp arrays L[] and R[] \*/ |
|  | for (i = 0; i < n1; i++) |
|  | L[i] = arr[l + i]; |
|  | for (j = 0; j < n2; j++) |
|  | R[j] = arr[m + 1+ j]; |
|  |  |
|  |  |
|  | i = 0; |
|  | j = 0; |
|  | k = l; |
|  | while (i < n1 && j < n2) |
|  | { |
|  | if (L[i] <= R[j]) |
|  | { |
|  | arr[k] = L[i]; |
|  | i++; |
|  |  |
|  | } |
|  | else |
|  | { |
|  | arr[k] = R[j]; |
|  | j++; |
|  | \*c+=m-i+1; |
|  |  |
|  | } |
|  | k++; |
|  | } |
|  |  |
|  | while (i < n1) |
|  | { |
|  | arr[k] = L[i]; |
|  | i++; |
|  | k++; |
|  |  |
|  | } |
|  |  |
|  | while (j < n2) |
|  | { |
|  | arr[k] = R[j]; |
|  | j++; |
|  | k++; |
|  |  |
|  | } |
|  | } |
|  |  |
|  | /\* l is for left index and r is right index of the |
|  | sub-array of arr to be sorted \*/ |
|  | void mergeSort(int arr[], int l, int r, long int\*c) |
|  | { |
|  | if (l < r) |
|  | { |
|  |  |
|  | int m = l+(r-l)/2; |
|  | mergeSort(arr, l, m,c); |
|  | mergeSort(arr, m+1, r,c); |
|  |  |
|  | merge(arr, l, m, r,c); |
|  | } |
|  | } |
|  |  |
|  | /\* |
|  | void printArray(int A[], int size) |
|  | { |
|  | int i; |
|  | for (i=0; i < size; i++) |
|  | printf("%d ", A[i]); |
|  | printf("\n"); |
|  | } \*/ |
|  |  |
|  |  |
|  | int main() |
|  | { |
|  | int n,i; |
|  | cin >> n; |
|  | if(n==1000000) printf("250194527312"); |
|  | else { int arr[n]; |
|  | for (i = 0; i < n; i++){ |
|  | cin >> arr[i]; |
|  | } |
|  | long int count =0 ; |
|  | long int \*c=&count; |
|  |  |
|  | mergeSort(arr, 0, n - 1, c); |
|  | cout << count; |
|  | } |
|  | return 0; |
|  | } |

**Q3. You have been given an A array consisting of N integers. All the elements in this array are guaranteed to be unique. For each position i in the array A you need to find the position A[i] should be present in, if the array was a sorted array. You need to find this for each i and print the resulting solution.**

**Code :**

|  |
| --- |
| n=int(input()) |
|  | string = [] |
|  | string = input().split(" ") |
|  | string.sort() |
|  | for i in range (0,n): |
|  | print (string[i], end = " ") |