**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **02-06-2020** | | | | | **Name:** | **Huda Sultana** | |
| **Sem & Sec** | **8 A** | | | | | **USN:** | **4AL16CS039** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **-** | | | | | | |
| **Max. Marks** | | **-** | | **Score** | | | **-** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **Step into Robotic Process Automation.** | | | | | | | |
| **Certificate Provider** | | | **UIPath** | | **Duration** | | | **180 mins** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement:**   1. **Given an array of positive integers.** **Write a C Program to find inversion count of array.** | | | | | | | | |
| **Status: Solved** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **Yes** | | | |
| **If yes Repository name** | | | | | **Hudasulltana/online\_coding** | | | |
| **Uploaded the report in slack** | | | | | **Yes** | | | |

Online Test Details: (Attach the snapshot and briefly write the report for the same)

NOT CONDUCTED

Certification Course Details: (Attach the snapshot and briefly write the report for the same)



Coding Challenges Details: (Attach the snapshot and briefly write the report for the same)

**PROGRAM 1 .**

**// Given an array of positive integers.** **Write a C Program to find inversion count of array.**

**#include<stdlib.h>**

**#include<stdio.h>**

**long count = 0;**

**void merge(int a[],int low,int mid,int high)**

**{**

**int l = mid-low+1,r=high-mid,\*p,\*q,i,j,k=low;**

**p = (int \*)malloc(l\*sizeof(int));**

**q = (int \*)malloc(r\*sizeof(int));**

**for(i=0;i<l;i++) p[i] = a[low+i];**

**for(i=0;i<r;i++) q[i] = a[mid+1+i];**

**i=0;**

**j=0;**

**while(i<l && j<r)**

**{**

**if(p[i] <= q[j])**

**{**

**a[k++] = p[i++];**

**}**

**else**

**{**

**a[k++] = q[j++];**

**count = count+l-i;**

**}**

**}**

**while(i<l)**

**{**

**a[k++] = p[i++];**

**//count++;**

**}**

**while(j<r)**

**{**

**a[k++] = q[j++];**

**}**

**free(p);**

**free(q);**

**return ;**

**}**

**void mergesort(int a[],int l,int h)**

**{**

**int mid = (l+h)/2;**

**if(l<h)**

**{**

**mergesort(a,l,mid);**

**mergesort(a,mid+1,h);**

**merge(a,l,mid,h);**

**}**

**return;**

**}**

**long inversion\_count(int a[],int n)**

**{**

**count=0;**

**mergesort(a,0,n-1);**

**return count;**

**}**

**int main() {**

**int t,n,\*a,i;**

**scanf("%d",&t);**

**while(t--)**

**{**

**scanf("%d",&n);**

**a = (int \*)malloc(n\*sizeof(int));**

**for(i=0;i<n;i++) scanf("%d",&a[i]);**

**printf("%li\n",inversion\_count(a,n));**

**free(a);**

**}**

**return 0;**

**}**