**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **29/05/20** | | | | | **Name:** | **SARANG VK** | |
| **Sem & Sec** | **8th B** | | | | | **USN:** | **4AL16CS085** | |
| **Online Test Summary** | | | | | | | | |
| **Subject** | | **BDA** | | | | | | |
| **Max. Marks** | | **30** | | **Score** | | | **14** | |
| **Certification Course Summary** | | | | | | | | |
| **Course** | **HADOOP** | | | | | | | |
| **Certificate Provider** | | | **GREATLEARNING** | | **Duration** | | | **34 MINUTES** |
| **Coding Challenges** | | | | | | | | |
| **Problem Statement:** Given an array arr[] of size N and an integer K. The task is to find the last remaining element in the array after reducing the array. | | | | | | | | |
| **Status:COMPLETED** | | | | | | | | |
| **Uploaded the report in Github** | | | | | **YES** | | | |
| **If yes Repository name** | | | | | **alvas-education-foundation/sarang\_vk** | | | |
| **Uploaded the report in slack** | | | | | **YES** | | | |

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)

Coding was given n it was uploaded for github and slack

PROGRAM1

void**moreThanNdK(intarr[], int n, int k)**

**{**

**// k must be greater than 1 to get some output**

**if (k < 2)**

**return;**

**/\* Step 1: Create a temporary array (contains element**

**and count) of size k-1. Initialize count of all**

**elements as 0 \*/**

**structeleCount temp[k-1];**

**for (inti=0; i<k-1; i++)**

**temp[i].c = 0;**

**/\* Step 2: Process all elements of input array \*/**

**for (inti = 0; i< n; i++)**

**{**

**int j;**

**/\* If arr[i] is already present in**

**the element count array, then increment its count \*/**

**for (j=0; j<k-1; j++)**

**{**

**if (temp[j].e == arr[i])**

**{**

**temp[j].c += 1;**

**break;**

**}**

**}**

**/\* If arr[i] is not present in temp[] \*/**

**if (j == k-1)**

**{**

**int l;**

**/\* If there is position available in temp[], then place**

**arr[i] in the first available position and set count as 1\*/**

**for (l=0; l<k-1; l++)**

**{**

**if (temp[l].c == 0)**

**{**

**temp[l].e = arr[i];**

**temp[l].c = 1;**

**break;**

**}**

**}**

**/\* If all the position in the temp[] are filled, then**

**decrease count of every element by 1 \*/**

**if (l == k-1)**

**for (l=0; l<k; l++)**

**temp[l].c -= 1;**

**}**

**}**

**/\*Step 3: Check actual counts of potential candidates in temp[]\*/**

**for (inti=0; i<k-1; i++)**

**{**

**// Calculate actual count of elements**

**int ac = 0; // actual count**

**for (int j=0; j<n; j++)**

**if (arr[j] == temp[i].e)**

**ac++;**

**// If actual count is more than n/k, then print it**

**if (ac > n/k)**

**cout<< "Number:" << temp[i].e**

**<< " Count:" << ac <<endl;**

**}**

**}**



