Rajalakshmi Engineering College

*241901104*

*241901104*

Name: Shafraz.R

*241901104*

*241901104*

Email: 241901104@rajalakshmi.edu.in

Roll no: 241901104

Phone: 6380360049

Branch: REC

Department: l CSE (CS) FB

Batch: 2028

Degree: B.E - CSE (CS)

NeoColab\_REC\_CS23231\_DATA STRUCTURES

*241901104*

*241901104*

REC\_DS using C\_Week 6\_COD\_Question 1

Attempt : 1

Total Mark : 10

Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John and Mary are collaborating on a project that involves data analysis. They each have a set of age data, one sorted in ascending order and the other in descending order. However, their analysis requires the data to be in ascending order.

*241901104*

*241901104*

*241901104*

*241901104*

Write a program to help them merge the two sets of age data into a single sorted array in ascending order using merge sort.

# Input Format

The first line of input consists of an integer N, representing the number of age values in each dataset.

*241901104*

*241901104*

*241901104*

*241901104*

The second line consists of N space-separated integers, representing the ages of participants in John's dataset (in ascending order).

The third line consists of N space-separated integers, representing the ages of participants in Mary's dataset (in descending order).

*241901104*

*241901104*

*241901104*

*241901104*

# Output Format

The output prints a single line containing space-separated integers, which represents the merged dataset of ages sorted in ascending order.

Refer to the sample output for formatting specifications.

*241901104*

*241901104*

*241901104*

*241901104*

# Sample Test Case

Input: 5

1 3 5 7 9

10 8 6 4 2

Output: 1 2 3 4 5 6 7 8 9 10

# Answer

#include <stdio.h>

void merge(int arr[], int left[], int right[], int left\_size, int right\_size) { int i = 0, j = 0, k = 0;

*241901104*

*241901104*

while (i < left\_size && j < right\_size) {

if (left[i] <= right[j]) { arr[k] = left[i]; i++; } else { arr[k] = right[j]; j++;

*241901104*

*241901104*

} k++;

}

while (i < left\_size) { arr[k] = left[i]; i++; k++;

*241901104*

*241901104*

*241901104*

*241901104*

}

while (j < right\_size) { arr[k] = right[j]; j++; k++;

*241901104*

*241901104*

*241901104*

*241901104*

}

}

void mergeSort(int arr[], int size) {

if (size > 1) { int mid = size / 2; int left[mid];

*241901104*

*241901104*

int right[size - mid];

for (int i = 0; i < mid; i++) { left[i] = arr[i];

*241901104*

*241901104*

}

for (int i = mid; i < size; i++) {

right[i - mid] = arr[i];

}

mergeSort(left, mid); mergeSort(right, size - mid);

merge(arr, left, right, mid, size - mid);

*241901104*

*241901104*

}

}

int main() { int n, m; scanf("%d", &n); int arr1[n], arr2[n]; for (int i = 0; i < n; i++) {

*241901104*

*241901104*

scanf("%d", &arr1[i]);

}

for (int i = 0; i < n; i++) {

scanf("%d", &arr2[i]);

*241901104*

*241901104*

}

int merged[n + n]; mergeSort(arr1, n); mergeSort(arr2, n); merge(merged, arr1, arr2, n, n); for (int i = 0; i < n + n; i++) { printf("%d ", merged[i]);

*241901104*

*241901104*

}

return 0;

}

*241901104*

*241901104*

*241901104*

*241901104*

# Status : CorrectMarks : 10/10

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*

*241901104*