

# Rajalakshmi Engineering College

Name: RANJITHAM R  
Email: 240801268@rajalakshmi.edu.in  
Roll no: 240801268  
Phone: 9514729237  
Branch: REC  
Department: I ECE AF  
Batch: 2028  
Degree: B.E - ECE

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### 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.

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.

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).

### **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.

### **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>
```

```
#include <stdio.h>
```

```
void merge(int arr[], int left[], int right[], int left_size, int right_size) {  
    int i = 0, j = 0, k = 0;
```

```
    // Merge elements into arr[]  
    while (i < left_size && j < right_size) {  
        if (left[i] <= right[j]) {  
            arr[k++] = left[i++];  
        } else {  
            arr[k++] = right[j++];  
        }  
    }  
}
```

```
    // Copy any remaining elements  
    while (i < left_size) {  
        arr[k++] = left[i++];  
    }
```

```
    while (j < right_size) {
```

```

        arr[k++] = right[j++];
    }
}

void mergeSort(int arr[], int n) {
    if (n < 2) return;

    int mid = n / 2;
    int left[20], right[20]; // Assuming max size is 20

    // Copy data to left and right subarrays
    for (int i = 0; i < mid; i++) {
        left[i] = arr[i];
    }
    for (int i = mid; i < n; i++) {
        right[i - mid] = arr[i];
    }

    mergeSort(left, mid);
    mergeSort(right, n - mid);
    merge(arr, left, right, mid, n - mid);
}

```

```

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

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: RANJITHAM R  
Email: 240801268@rajalakshmi.edu.in  
Roll no: 240801268  
Phone: 9514729237  
Branch: REC  
Department: I ECE AF  
Batch: 2028  
Degree: B.E - ECE

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 6\_COD\_Question 2

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Nandhini asked her students to arrange a set of numbers in ascending order. She asked the students to arrange the elements using insertion sort, which involves taking each element and placing it in its appropriate position within the sorted portion of the array.

Assist them in the task.

##### ***Input Format***

The first line of input consists of the value of n, representing the number of array elements.

The second line consists of n elements, separated by a space.

##### ***Output Format***

The output prints the sorted array, separated by a space.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

67 28 92 37 59

Output: 28 37 59 67 92

### **Answer**

```
#include <stdio.h>
```

```
#include <stdio.h>
```

```
// Function to sort the array using insertion sort
```

```
void insertionSort(int arr[], int n) {
```

```
    int i, key, j;
```

```
    for (i = 1; i < n; i++) {
```

```
        key = arr[i];
```

```
        j = i - 1;
```

```
        // Move elements that are greater than key one position ahead
```

```
        while (j >= 0 && arr[j] > key) {
```

```
            arr[j + 1] = arr[j];
```

```
            j--;
```

```
        }
```

```
        arr[j + 1] = key;
```

```
    }
```

```
}
```

```
// Function to print the array
```

```
void printArray(int arr[], int n) {
```

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

```
        printf("%d ", arr[i]);
```

```
    printf("\n");
```

```
}
```

```
int main() {
```

```
    int n;
```

```
scanf("%d", &n);
int arr[n];
for (int i = 0; i < n; i++) {
    scanf("%d", &arr[i]);
}

insertionSort(arr, n);
printArray(arr, n);
return 0;
}
```

**Status :** Correct

**Marks : 10/10**

# Rajalakshmi Engineering College

Name: RANJITHAM R  
Email: 240801268@rajalakshmi.edu.in  
Roll no: 240801268  
Phone: 9514729237  
Branch: REC  
Department: I ECE AF  
Batch: 2028  
Degree: B.E - ECE

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 6\_COD\_Question 3

Attempt : 1  
Total Mark : 10  
Marks Obtained : 0

#### Section 1 : Coding

##### 1. Problem Statement

You are the lead developer of a text-processing application that assists writers in organizing their thoughts. One crucial feature is a character-sorting service that helps users highlight the most critical elements of their text.

To achieve this, you decide to enhance the service to sort characters in descending order using the Quick-Sort algorithm. Implement the algorithm to efficiently rearrange the characters, ensuring that it is sorted in descending order.

##### ***Input Format***

The first line of the input consists of a positive integer value N, representing the number of characters to be sorted.



The second line of input consists of N space-separated lowercase alphabetical characters.

### **Output Format**

The output displays the set of alphabetical characters, sorted in descending order.

Refer to the sample output for the formatting specifications.

### **Sample Test Case**

Input: 5

a d g j k

Output: k j g d a

### **Answer**

```
#include <stdio.h>
```

```
#include <string.h>
```

```
// Function to swap two characters
```

```
void swap(char *a, char *b) {
```

```
    char temp = *a;
```

```
    *a = *b;
```

```
    *b = temp;
```

```
}
```

```
// Partition function for Quick Sort (descending order)
```

```
int partition(char arr[], int low, int high) {
```

```
    char pivot = arr[high];
```

```
    int i = low - 1;
```

```
    for (int j = low; j < high; j++) {
```

```
        if (arr[j] > pivot) { // Sorting in descending order
```

```
            i++;
```

```
            swap(&arr[i], &arr[j]);
```

```
        }
```

```
    }
```

```

        swap(&arr[i + 1], &arr[high]);
        return (i + 1);
    }

// Quick Sort function
void quickSort(char arr[], int low, int high) {
    if (low < high) {
        int pivotIndex = partition(arr, low, high);
        quickSort(arr, low, pivotIndex - 1);
        quickSort(arr, pivotIndex + 1, high);
    }
}

// Function to display the sorted array correctly
void displayArray(char arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%c", arr[i]); // Print character
        if (i != n - 1) {
            printf(" "); // Print space only if it's not the last element
        }
    }
    printf("\n");
}

int main() {
    int n;
    scanf("%d", &n);

    char characters[n];

    for (int i = 0; i < n; i++) {
        char input;
        scanf(" %c", &input);
        characters[i] = input;
    }

    quicksort(characters, 0, n - 1);

    for (int i = 0; i < n; i++) {
        printf("%c ", characters[i]);
    }
}

```

```
} return 0;
```

**Status : Wrong**

**Marks : 0/10**

# Rajalakshmi Engineering College

Name: RANJITHAM R  
Email: 240801268@rajalakshmi.edu.in  
Roll no: 240801268  
Phone: 9514729237  
Branch: REC  
Department: I ECE AF  
Batch: 2028  
Degree: B.E - ECE

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 6\_COD\_Question 4

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Kavya, a software developer, is analyzing data trends. She has a list of integers and wants to identify the  $n$ th largest number in the list after sorting the array using QuickSort.

To optimize performance, Kavya is required to use QuickSort to sort the list before finding the  $n$ th largest number.

##### ***Input Format***

The first line of input consists of an integer  $n$ , representing the size of the array.

The second line consists of  $n$  space-separated integers, representing the elements of the array `nums`.

The third line consists of an integer  $k$ , representing the position of the largest

number you need to print after sorting the array.

### **Output Format**

The output prints the k-th largest number in the sorted array (sorted in ascending order).

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 6

-1 0 1 2 -1 -4

3

Output: 0

### **Answer**

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
int partition(int arr[], int low, int high) {
```

```
    int pivot = arr[high];
```

```
    int i = low - 1;
```

```
    for (int j = low; j <= high; j++) {
```

```
        if (arr[j] < pivot) {
```

```
            i++;
```

```
            int temp = arr[i];
```

```
            arr[i] = arr[j];
```

```
            arr[j] = temp;
```

```
        }
```

```
    }
```

```
    int temp = arr[i + 1];
```

```
    arr[i + 1] = arr[high];
```

```
    arr[high] = temp;
```

```
    return i + 1;
```

```
}
```

```
void quickSort(int arr[], int low, int high) {
```

```
    if (low < high) {
```

```
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

void findNthLargest(int* nums, int n, int k) {
    quickSort(nums, 0, n - 1);
    printf("%d\n", nums[n - k]);
}

int main() {
    int n, k;
    scanf("%d", &n);
    int* nums = (int*)malloc(n * sizeof(int));
    for (int i = 0; i < n; i++) {
        scanf("%d", &nums[i]);
    }
    scanf("%d", &k);
    findNthLargest(nums, n, k);
    free(nums);
    return 0;
}
```

**Status :** Correct

**Marks :** 10/10

# Rajalakshmi Engineering College

Name: RANJITHAM R  
Email: 240801268@rajalakshmi.edu.in  
Roll no: 240801268  
Phone: 9514729237  
Branch: REC  
Department: I ECE AF  
Batch: 2028  
Degree: B.E - ECE

Scan to verify results



## NeoColab\_REC\_CS23231\_DATA STRUCTURES

### REC\_DS using C\_Week 6\_COD\_Question 5

Attempt : 1  
Total Mark : 10  
Marks Obtained : 10

#### Section 1 : Coding

##### 1. Problem Statement

Jose has an array of N fractional values, represented as double-point numbers. He needs to sort these fractions in increasing order and seeks your help.

Write a program to help Jose sort the array using the merge sort algorithm.

##### ***Input Format***

The first line of input consists of an integer N, representing the number of fractions to be sorted.

The second line consists of N double-point numbers, separated by spaces, representing the fractions array.

##### ***Output Format***

The output prints N double-point numbers, sorted in increasing order, and rounded to three decimal places.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 4

0.123 0.543 0.321 0.789

Output: 0.123 0.321 0.543 0.789

### **Answer**

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <stdio.h>
```

```
// Merge function for merge sort
```

```
void merge(double arr[], int left, int mid, int right) {
```

```
    int i, j, k;
```

```
    int n1 = mid - left + 1;
```

```
    int n2 = right - mid;
```

```
    double L[10], R[10]; // Temporary arrays ( $N \leq 10$ )
```

```
    for (i = 0; i < n1; i++)
```

```
        L[i] = arr[left + i];
```

```
    for (j = 0; j < n2; j++)
```

```
        R[j] = arr[mid + 1 + j];
```

```
    i = 0;
```

```
    j = 0;
```

```
    k = left;
```

```
    while (i < n1 && j < n2) {
```

```
        if (L[i] <= R[j])
```

```
            arr[k++] = L[i++];
```

```
        else
```

```
            arr[k++] = R[j++];
```

```
    }
```



```

while (i < n1)
    arr[k++] = L[i++];

while (j < n2)
    arr[k++] = R[j++];
}

// Merge sort function
void mergeSort(double arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;

        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);

        merge(arr, left, mid, right);
    }
}

int main() {
    int n;
    scanf("%d", &n);
    double fractions[n];
    for (int i = 0; i < n; i++) {
        scanf("%lf", &fractions[i]);
    }
    mergeSort(fractions, 0, n - 1);
    for (int i = 0; i < n; i++) {
        printf("%.3f ", fractions[i]);
    }
    return 0;
}

```

**Status :** Correct

**Marks :** 10/10