

# Rajalakshmi Engineering College

Name: pooja D  
Email: 240701385@rajalakshmi.edu.in  
Roll no: 240701385  
Phone: 9677250159  
Branch: REC  
Department: I CSE FD  
Batch: 2028  
Degree: B.E - CSE

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 : 10

#### 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>
```

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

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

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

```
    *b = temp;
```

```
}
```

```
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) {
```

```
            i++;
```

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

```
        }
```

```
    }
```

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

```
    return i + 1;
```

```
}
```

```
void quicksort(char arr[], int low, int high) {
    if (low < high) {
        int pi = partition(arr, low, high);
        quicksort(arr, low, pi - 1);
        quicksort(arr, pi + 1, high);
    }
}

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 :** Correct

**Marks :** 10/10