

# Rajalakshmi Engineering College

Name: Prajeet V  
Email: 240701389@rajalakshmi.edu.in  
Roll no: 240701389  
Phone: 9363389322  
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\_CY\_Updated

Attempt : 1  
Total Mark : 30  
Marks Obtained : 30

### Section 1 : Coding

#### 1. Problem Statement

Aryan is participating in a coding competition where he needs to sort a list of numbers using an efficient sorting algorithm. He decides to use Merge Sort, a divide-and-conquer algorithm, to achieve this. Given a list of  $n$  elements, Aryan must implement merge sort to arrange the numbers in ascending order.

Help Aryan by implementing the merge sort algorithm to correctly sort the given list of numbers.

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

The first line of input contains an integer  $n$ , the number of elements in the list.

The second line contains  $n$  space-separated integers representing the elements

of the list.

### **Output Format**

The output prints the sorted list of numbers in ascending order, separated by a space.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

80 40 20 50 30

Output: 20 30 40 50 80

### **Answer**

```
// You are using GCC
```

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

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

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

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

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

```
    int L[n1], R[n2];
```

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

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

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

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

```
    int i = 0, j = 0, k = left;
```

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

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

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

```
            i++;
```

```
        } else {
```

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

```
            j++;
```

```
        }
```

```
        k++;
```

```
}
```

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

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

```
void mergeSort(int 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);  
  
    int arr[n];  
    for (int i = 0; i < n; i++) {  
        scanf("%d", &arr[i]);  
    }  
  
    mergeSort(arr, 0, n - 1);  
  
    for (int i = 0; i < n; i++) {  
        printf("%d ", arr[i]);  
    }  
    printf("\n");  
}
```

```
    return 0;  
}
```

**Status :** Correct

**Marks :** 10/10

## 2. Problem Statement

Marie, the teacher, wants her students to implement the ascending order of numbers while also exploring the concept of prime numbers.

Students need to write a program that sorts an array of integers using the merge sort algorithm while counting and returning the number of prime integers in the array. Help them to complete the program.

### **Input Format**

The first line of input consists of an integer N, representing the number of array elements.

The second line consists of N space-separated integers, representing the array elements.

### **Output Format**

The first line of output prints the sorted array of integers in ascending order.

The second line prints the number of prime integers in the array.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 7

5 3 6 8 9 7 4

Output: Sorted array: 3 4 5 6 7 8 9

Number of prime integers: 3

### **Answer**

```
// You are using GCC
```

```

#include <stdio.h>
#include <stdbool.h>

void merge(int arr[], int left, int mid, int right) {
    int n1 = mid - left + 1;
    int n2 = right - mid;

    int L[n1], R[n2];

    for (int i = 0; i < n1; i++)
        L[i] = arr[left + i];
    for (int j = 0; j < n2; j++)
        R[j] = arr[mid + 1 + j];

    int i = 0, j = 0, k = left;
    while (i < n1 && j < n2) {
        if (L[i] <= R[j]) {
            arr[k] = L[i];
            i++;
        } else {
            arr[k] = R[j];
            j++;
        }
        k++;
    }

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

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

void mergeSort(int 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);
    }
}
```

```
bool isPrime(int num) {
    if (num <= 1)
        return false;
    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0)
            return false;
    }
    return true;
}
```

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

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

    mergeSort(arr, 0, n - 1);

    printf("Sorted array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    int primeCount = 0;
    for (int i = 0; i < n; i++) {
        if (isPrime(arr[i])) {
            primeCount++;
        }
    }
}
```

```
printf("Number of prime integers: %d\n", primeCount);  
return 0;  
}
```

**Status :** Correct

**Marks :** 10/10

### 3. Problem Statement

Reshma is passionate about sorting algorithms and has recently learned about the merge sort algorithm. She wants to implement a program that utilizes the merge sort algorithm to sort an array of integers, both positive and negative, in ascending order.

Help her in implementing the program.

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

The first line of input consists of an integer N, representing the number of elements in the array.

The second line of input consists of N space-separated integers, representing the elements of the array.

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

The output prints N space-separated integers, representing the array elements sorted in ascending order.

Refer to the sample output for formatting specifications.

#### ***Sample Test Case***

Input: 9

5 -3 0 12 7 -8 2 1 6

Output: -8 -3 0 1 2 5 6 7 12

#### ***Answer***

// You are using GCC

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

```
void merge(int arr[], int left, int mid, int right) {  
    int n1 = mid - left + 1;  
    int n2 = right - mid;
```

```
    int L[n1], R[n2];
```

```
    for (int i = 0; i < n1; i++)  
        L[i] = arr[left + i];  
    for (int j = 0; j < n2; j++)  
        R[j] = arr[mid + 1 + j];
```

```
    int i = 0, j = 0, k = left;  
    while (i < n1 && j < n2) {  
        if (L[i] <= R[j]) {  
            arr[k] = L[i];  
            i++;  
        } else {  
            arr[k] = R[j];  
            j++;  
        }  
        k++;  
    }
```

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

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

```
void mergeSort(int 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);

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

    mergeSort(arr, 0, N - 1);

    for (int i = 0; i < N; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}
```

**Status :** Correct

**Marks :** 10/10