S.No: 11	Exp. Name: ***Write a C program to Sort given elements using Merge sort***	Date:2023-04-08

ID: 22K61A0574

Aim:

Write a program to `sort` (`Ascending order`) the given elements using `merge sort` technique.

At the time of execution, the program should print the message on the console as:

```
Enter array size :
```

For example, if the user gives the **input** as:

```
Enter array size : 5
```

Next, the program should print the following message on the console as:

```
Enter 5 elements :
```

if the user gives the **input** as:

```
Enter 5 elements : 34 67 12 45 22
```

then the program should **print** the result as:

```
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67
```

Note: Do use the **printf()** function with a **newline** character (`\n`).

Source Code:

MergeSortMain.c

```
#include <stdio.h>
#include "MergeSortFunctions.c"
void main() {
   int arr[15], i, n;
   printf("Enter array size : ");
   scanf("%d", &n);
   printf("Enter %d elements : ", n);
   for (i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }
   printf("Before sorting the elements are : ");
   display(arr, n);
   splitAndMerge(arr, 0, n - 1);
   printf("After sorting the elements are : ");
   display(arr, n);
}
```

MergeSortFunctions.c

```
void display(int arr[15], int n) {
    int i,b[15];
    for(i=0;i<n;i++)
    {
```

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

    }
    printf("\n");
}
void merge(int arr[15], int low, int mid, int high) {
   int b[15];
   int i = low,j = mid+1,k = low;
   while(i <= mid && j <= high)
   {
      if(arr[i] <=arr[j])
      {
         b[k] = arr[i];
         i++;
         k++;
      }
      else
      {
         b[k] = arr[j];
         j++;
         k++;
      }
   }
   while(i <= mid)
   {
      b[k] = arr[i];
      k++;
      i++;
   }
   while(j <= high)
   {
      b[k] = arr[j];
      k++;
      j++;
   }
   for(i = low;i <= high;i++)
   {
      arr[i] = b[i];
   }
}
void splitAndMerge(int arr[15], int low, int high) {
   int mid;
   if(low < high)
   {
      mid=(low+high)/2;
      splitAndMerge(arr,low,mid);
      splitAndMerge(arr,mid+1,high);
      merge(arr,low,mid,high);
   }

}
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output
Enter array size : 5
Enter 5 elements : 34 67 12 45 22
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Test Case - 2
User Output
Enter array size : 8
Enter 8 elements : 77 55 22 44 99 33 11 66
Before sorting the elements are : 77 55 22 44 99 33 11 66
After sorting the elements are : 11 22 33 44 55 66 77 99

Test Case - 3
User Output
Enter array size : 5
Enter 5 elements : -32 -45 -67 -46 -14
Before sorting the elements are : -32 -45 -67 -46 -14
After sorting the elements are : -67 -46 -45 -32 -14