MERGE SORT

CODE:

```
#include <bits/stdc++.h>
using namespace std;

void merge(int array[], int const left, int const mid,
           int const right)
{
    int const subArrayOne = mid - left + 1;
    int const subArrayTwo = right - mid;

    // Create temp arrays
    auto *leftArray = new int[subArrayOne],
         *rightArray = new int[subArrayTwo];

    // Copy data to temp arrays leftArray[] and rightArray[]
    for (auto i = 0; i < subArrayOne; i++)
        leftArray[i] = array[left + i];
    for (auto j = 0; j < subArrayTwo; j++)
        rightArray[j] = array[mid + 1 + j];

    auto indexOfSubArrayOne = 0, indexOfSubArrayTwo = 0;
    int indexOfMergedArray = left;

    // Merge the temp arrays back into array[left..right]
    while (indexOfSubArrayOne < subArrayOne
        && indexOfSubArrayTwo < subArrayTwo) {
        if (leftArray[indexOfSubArrayOne]
            <= rightArray[indexOfSubArrayTwo]) {
            array[indexOfMergedArray]
                = leftArray[indexOfSubArrayOne];
            indexOfSubArrayOne++;
        }
```

```
            else {
                array[indexOfMergedArray]
                    = rightArray[indexOfSubArrayTwo];
                indexOfSubArrayTwo++;
            }
            indexOfMergedArray++;
        }

        // Copy the remaining elements of
        // left[], if there are any
        while (indexOfSubArrayOne < subArrayOne) {
            array[indexOfMergedArray]
                = leftArray[indexOfSubArrayOne];
            indexOfSubArrayOne++;
            indexOfMergedArray++;
        }

        // Copy the remaining elements of
        // right[], if there are any
        while (indexOfSubArrayTwo < subArrayTwo) {
            array[indexOfMergedArray]
                = rightArray[indexOfSubArrayTwo];
            indexOfSubArrayTwo++;
            indexOfMergedArray++;
        }
        delete[] leftArray;
        delete[] rightArray;
}

void mergeSort(int array[], int const begin, int const end)
{
        if (begin >= end)
            return;

        int mid = begin + (end - begin) / 2;
```

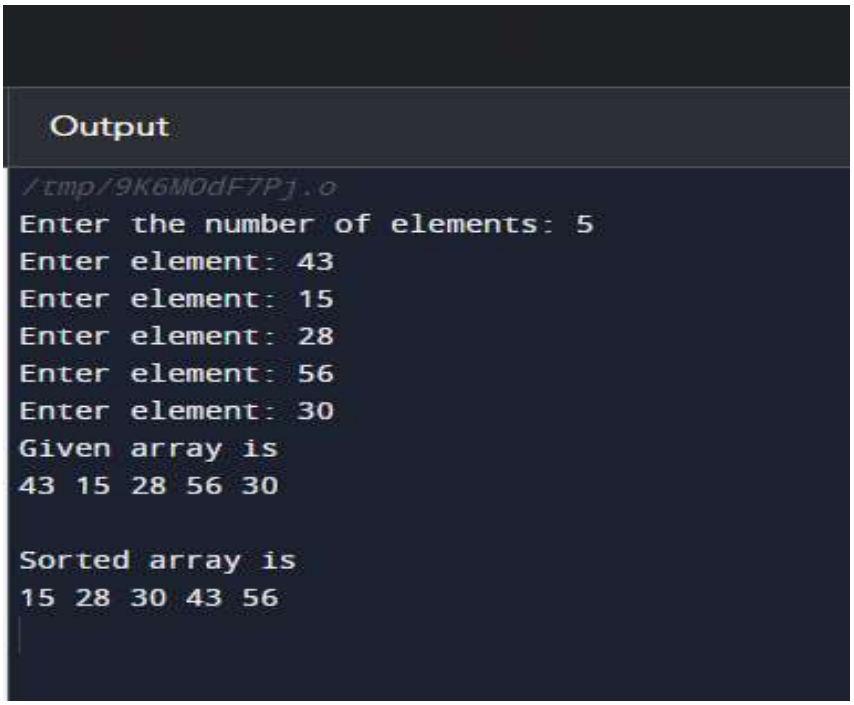
```
        mergeSort(array, begin, mid);
        mergeSort(array, mid + 1, end);
        merge(array, begin, mid, end);
}

// Driver code
int main()
{
   int n;
   cout<<"Enter the number of elements: ";
   cin>>n;
   int arr[n];
   for (int i=0; i<n; i++){
      cout<<"Enter element: ";
      cin>>arr[i];
   }
   int size;
        cout << "Given array is \n";
        for (int i = 0; i < n; i++)
                cout << arr[i] << " ";
        cout << endl;

        mergeSort(arr, 0, n - 1);

        cout << "\nSorted array is \n";
        for (int i = 0; i < n; i++)
                cout << arr[i] << " ";
        cout << endl;
        return 0;
}
```

OUTPUT:


Output
/tmp/9K6MOdF7Pj.o
Enter the number of elements: 5
Enter element: 43
Enter element: 15
Enter element: 28
Enter element: 56
Enter element: 30
Given array is
43 15 28 56 30
Sorted array is
15 28 30 43 56