// C++ program for Merge Sort

#include<iostream>

using namespace std;

// Merges two subarrays of arr[].

// First subarray is arr[l..m]

// Second subarray is arr[m+1..r]

void merge(int arr[], int l, int m, int r)

{

int n1 = m - l + 1;

int n2 = r - m;

// Create temp arrays

int L[n1], R[n2];

// Copy data to temp arrays L[] and R[]

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

L[i] = arr[l + i];

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

R[j] = arr[m + 1 + j];

// Merge the temp arrays back into arr[l..r]

// Initial index of first subarray

int i = 0;

// Initial index of second subarray

int j = 0;

// Initial index of merged subarray

int k = l;

while (i < n1 && j < n2)

{

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

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

// Copy the remaining elements of

// L[], if there are any

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

// Copy the remaining elements of

// R[], if there are any

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

// l is for left index and r is

// right index of the sub-array

// of arr to be sorted \*/

void mergeSort(int arr[], int l, int r)

{

if (l < r)

{

// Same as (l+r)/2, but avoids

// overflow for large l and h

int m = l + (r - l) / 2;

// Sort first and second halves

mergeSort(arr, l, m);

mergeSort(arr, m + 1, r);

merge(arr, l, m, r);

}

}

// UTILITY FUNCTIONS

// Function to print an array

void printArray(int A[], int size)

{

for(int i = 0; i < size; i++)

cout << A[i] << " ";

}

// Driver code

int main()

{

int arr[] = { 12, 11, 13, 5, 6, 7 };

int arr\_size = sizeof(arr) / sizeof(arr[0]);

cout << "Given array is \n";

printArray(arr, arr\_size);

mergeSort(arr, 0, arr\_size - 1);

cout << "\nSorted array is \n";

printArray(arr, arr\_size);

return 0;

}

// This code is contributed by Mayank Tyagi