# Python program for implementation of MergeSort

# Merges two subarrays of arr[].

# First subarray is arr[l..m]

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

def merge(arr, l, m, r):

n1 = m - l + 1

n2 = r - m

# create temp arrays

L = [0] \* (n1)

R = [0] \* (n2)

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

for i in range(0, n1):

L[i] = arr[l + i]

for j in range(0, n2):

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

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

i = 0 # Initial index of first subarray

j = 0 # Initial index of second subarray

k = l # Initial index of merged subarray

while i < n1 and j < n2:

if L[i] <= R[j]:

arr[k] = L[i]

i += 1

else:

arr[k] = R[j]

j += 1

k += 1

# Copy the remaining elements of L[], if there

# are any

while i < n1:

arr[k] = L[i]

i += 1

k += 1

# Copy the remaining elements of R[], if there

# are any

while j < n2:

arr[k] = R[j]

j += 1

k += 1

# l is for left index and r is right index of the

# sub-array of arr to be sorted

def mergeSort(arr, l, r):

if l < r:

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

# large l and h

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

# Sort first and second halves

mergeSort(arr, l, m)

mergeSort(arr, m+1, r)

merge(arr, l, m, r)

# Driver code to test above

arr = [12, 11, 13, 5, 6, 7]

n = len(arr)

print("Given array is")

for i in range(n):

print("%d" % arr[i],end=" ")

mergeSort(arr, 0, n-1)

print("\n\nSorted array is")

for i in range(n):

print("%d" % arr[i],end=" ")

# This code is contributed by Mohit Kumra