### Merge Sort

Given an array arr[], its starting position l and its ending position r. Sort the array using merge sort algorithm.  
**Example 1:**

**Input:**

N = 5

arr[] = {4 1 3 9 7}

**Output:**

1 3 4 7 9

**Example 2:**

**Input:**

N = 10

arr[] = {10 9 8 7 6 5 4 3 2 1}

**Output:**

1 2 3 4 5 6 7 8 9 10

**Expected Time Complexity:** O(nlogn)

**Expected Auxiliary Space:** O(n)

**Constraints:**  
1 <= N <= 105  
1 <= arr[i] <= 105

**Company Tags**

[**Paytm**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Paytm) [**Amazon**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Amazon) [**Microsoft**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Microsoft) [**Snapdeal**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Snapdeal) [**Oracle**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Oracle) [**Goldman Sachs**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Goldman%20Sachs) [**Wipro**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Wipro) [**Qualcomm**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Qualcomm) [**Boomerang Commerce**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Boomerang%20Commerce) [**Grofers**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Grofers) [**Target Corporation**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Target%20Corporation) [**Medlife**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Medlife)

//{ Driver Code Starts

import java.util.\*;

class Merge\_Sort

{

//method to print the elements of the array

static void printArray(int arr[])

{

StringBuffer sb=new StringBuffer("");

int n = arr.length;

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

sb.append(arr[i]+" ");

System.out.println(sb.toString());

}

public static void main(String args[])

{

//taking input using Scanner class

Scanner sc = new Scanner(System.in);

//taking testcases

int T = sc.nextInt();

while(T>0)

{

//taking elements count

int n = sc.nextInt();

//creating an object of class Merge\_Sort

Merge\_Sort ms = new Merge\_Sort();

//creating an array of size n

int arr[] = new int[n];

//adding elements to the array

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

arr[i] = sc.nextInt();

Solution g = new Solution();

//calling the method mergeSort

g.mergeSort(arr,0,arr.length-1);

//calling the method printArray

ms.printArray(arr);

T--;

}

}

}

// } Driver Code Ends

class Solution

{

void merge(int arr[], int low, int mid, int high)

{

ArrayList<Integer> temp = new ArrayList<>();

int left=low;

int right=mid+1;

while(left<=mid && right<=high){

if(arr[left]<=arr[right]){

temp.add(arr[left]);

left++;

}

else{

temp.add(arr[right]);

right++;

}

}

//if elements are still left in the arr

while(left<=mid){

temp.add(arr[left]);

left++;

}

while(right<=high){

temp.add(arr[right]);

right++;

}

//putting temp array into original

for (int i = low; i <= high; i++) {

arr[i] = temp.get(i - low);

}

}

void mergeSort(int arr[], int low, int high)

{

//code here

if(low==high)return;

int mid= (low+high)/2;

mergeSort(arr, low , mid);

mergeSort(arr, mid+1 ,high);

merge(arr,low,mid,high);

}

}