**Sorting:**

**1.Sort the Half Sorted**

Given an integer array of which both first half and second half are sorted. The task is to merge two sorted halves of the array into a single sorted array.  
**Note**: The two halves can be of arbitrary sizes.

**Example 1:**

**Input:**

N = 6

arr[] = {2 3 8 -1 7 10}

**Output:** -1 2 3 7 8 10

**Explanation:** {2 3 8} and {-1 7 10} are sorted

in the original array. The overall sorted

version is {-1 2 3 7 8 10}

**Example 2:**

**Input:**

N = 5

arr[] = {-4 6 9 -1 3}

**Output:** -4 -1 3 6 9

**Explanation:** {-4 -1} and {3 6 9} are sorted

in the original array. The overall sorted

version is {-4 -1 3 6 9}

**2.Sorting Employees**

You have records of employee name as string (**Ename**) and salary as positive integer (**S**). You have to sort the records on the basis of employee salary, if salary is same then use employee name for comparison.

**Example 1:**

**Input:** N = 2

arr[] = {{xbnnskd, 100}, {geek, 50}}

**Output:** {geek 50}, {xbnnskd 100}

**Explanation**: geek has lowest salary

as 50 and xbnnskd has more salary.

**Example 2:**

**Input:** N = 2

arr[] = {{shyam, 50}, {ram, 50}}

**Output:** ram 50 shyam 50

**Binary Search Tree:**

1. **Maximum sum leaf to root path**

Given a Binary Tree, find the maximum sum path from a leaf to root.

**Example 1:**

**Input:**

1

/ \

2 3

**Output:**

4

**Explanation:**

Following the path 3 -> 1, results in a

sum of 4, which is the maximum path sum

from leaf to root for the given tree.

**Example 2:**

**Input:**

10

/ \

-2 7

/ \

8 -4

**Output:**

17

**Explanation :**

Following the path 7 -> 10, results in a

sum of 17, which is the maximum path sum

from leaf to root for the given tree.

**2. Delete nodes greater than k**

Given a **BST** and a **value k**, the task is to delete the nodes having values **greater than or equal to k**.

**Example 1:**

**Input:**

4

/ \

1 9

k = 2

**Output:**

1