CSGE602040 - Data Structures and Algorithms
Even Semester - 2020/2021
Lab 3
Deadline: Friday, 30-4-2021, 23.55 WIB

Mr. Ustadz Shares Takjil

Descriptions

In this month of Ramadan 1442 H, Mr. Ustadz has a new hobby of making takjil for breaking the fast. Mr. Ustadz is pursuing this hobby in his own kitchen. Mr. Ustadz can make all kinds of takjil just by watching cooking tutorials on YouTube. Before Mr. Ustadz made his takjil, he wanted to collect information on the distance between his neighbours who will be receiving his takjil. Then Mr. Ustadz wants to see the information in Binary Search Tree (BST).

After that, Mr. Ustadz divided the takjil delivery groups to **N** BSTs. Then, Mr. Ustadz will enter the distance of the first recipient's house as the root tree, which is the relative distance between the first recipient and Mr. Ustadz's house. The next recipient will be calculated the relative distance to the previous recipient. After Mr. Ustadz enters the distance of their house, Mr. Ustadz will call his neighbors to confirm whether they are at home or not. If they are not at home, the distance between their house and Mr. Ustadz's house will be deleted from the takjil delivery group data. Assume that the activity of erasing the distance from the takjil delivery group data will be carried out after the BST has been successfully created.

Queries that need to be done is :

- **REMOVE a_i**: Removing the relative distance of the neighbor's house from the previous neighbor's house from the BST. In this query, ai is guaranteed to be valid or in the tree.

Mr. Ustadz asked for help from you, who is a reliable programmer, to provide the best route for takjil delivery so that Mr. Ustadz can make takjil solemnly.

Input

The first line contains the integer **N** which represents the number of Binary Search Trees (BST) that will be executed.

For each BST, there are **2 + M** lines containing:

- The integer **P** which states the number of relative distance nodes in the tree and is followed by **P** of the integers $a_1, a_2, \ldots, a_p$ which is the distance that has been recorded by Mr. Ustadz in the form of the **PostOrder Traversal** BST sequence. This PostOrder is guaranteed to form BST.
- Integer **M**, which represents the number of **REMOVE a_i** queries.

Output

The output consists of **N** integers which represent the height of the tree after execution. Issue **-1** if the tree is empty (has 0 nodes).

Limitations

$1 \le N \le 100000$

$1 \le P \le 100$

$0 \le M \le P$

$1 \le a_i \le 10^9$

Input Example 1

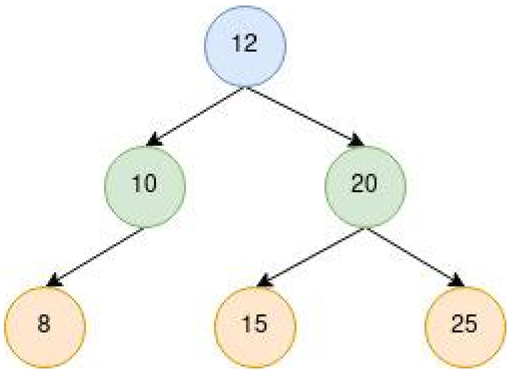
```
1
6 8 10 15 25 20 12
2
REMOVE 10
REMOVE 8
```

Output Example 1

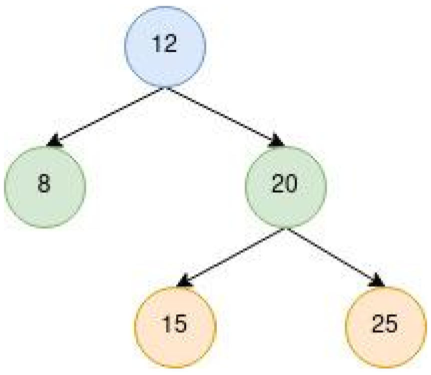
```
2
```

Explanation 1

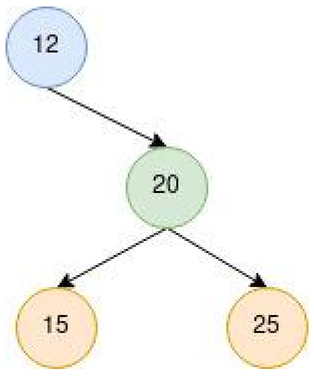
The tree that has the PostOrder Traversal is as follows:



After that, if we run the **REMOVE 10** query, the tree will change to something like this:



After that, if we run the **REMOVE 8** query, the tree will change to something like this:



The image above is the final shape of the tree. From there, we can see that the tree's height is **2**.

Input Example 2

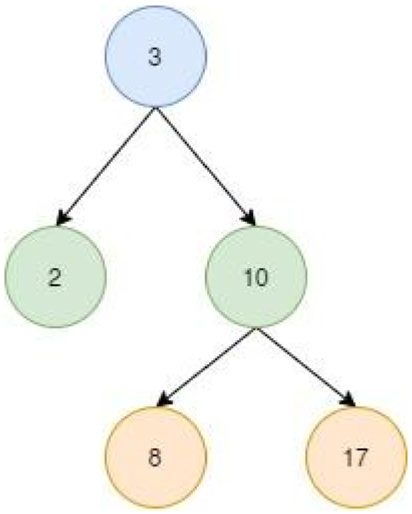
```
2
5 2 8 17 10 3
1
REMOVE 10
3 20 50 30
3
REMOVE 30
REMOVE 20
REMOVE 50
```

Output Example 2

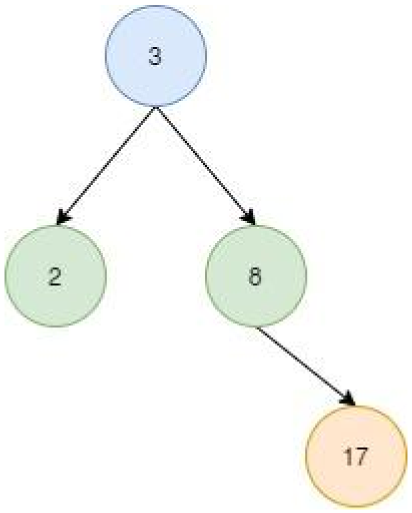
```
2
-1
```

Explanation 2

Tree 1
The first tree that has a PostOrder Traversal is as follows:

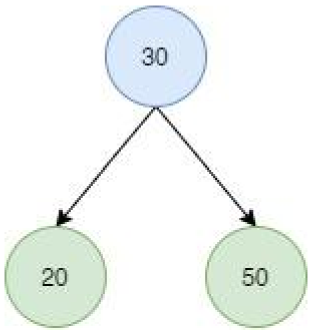


After that, if we run the **REMOVE 10** query, the tree will change to something like this:

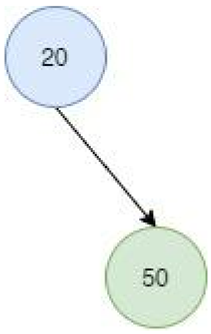


From the final shape of the first tree, we can see that the tree's height is **2**.

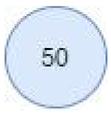
Tree 2
The second tree that has PostOrder Traversal is as follows:



After that, if we run the **REMOVE 30** query, the tree will change to something like this:



After that, if we run the **REMOVE 20** query, the tree will change to something like this:



After we run the **REMOVE 50** query, the tree will be empty. Since the final contents of the second tree are empty, we know that the height of the tree is **-1**.