AVL Logger

by: Arya Sudewa

Time Limit: 1 s
Memory Limit: 256 MB

Deskripsi

Buatlah sebuah AVL tree yang memiliki sistem logging, di mana setiap operasi akan menghasilkan output sesuai dengan format yang telah ditentukan.

Diberikan **N** perintah. Setiap perintah dapat berupa salah satu dari tiga operasi berikut:

1. **INSERT X**: Masukkan bilangan **X** ke dalam tree jika belum ada.
2. **DELETE X**: Hapus bilangan **X** dari tree jika ada.
3. **FIND X**: Cari apakah bilangan **X** ada di tree. Jika ada, tampilkan jalur traversal dari root hingga node **X**.

Batasan

- $1 \leq \mathbf{N} \leq 10^5$
- $-10^9 \leq \mathbf{X} \leq 10^9$

Format Masukan

Baris pertama berisi sebuah bilangan bulat **N**, menyatakan jumlah perintah.

N baris berikutnya masing-masing berisi sebuah perintah dalam format: **INSERT X**, **DELETE X**, atau **FIND X**.

Format Keluaran

1. Untuk **INSERT X**:
   - Jika **X** belum ada di tree, output [**X inserted**].
   - Jika **X** sudah ada di tree, output [**X is already in the tree**].
2. Untuk **DELETE X**:
   - Jika **X** ada di AVL, output [**X deleted**].
   - Jika **X** tidak ada di tree, output [**X is not found in the tree**].
3. Untuk **FIND X**:
   - Jika **X** ada di tree, output [**X found with path:**] dan output jalur yang dilalui dari root hingga node dengan bilangan **X**.
   - Jika **X** tidak ada di tree, output [**X not found**].

Contoh Masukan

```
7
INSERT 6
INSERT 3
INSERT 9
INSERT 4
FIND 4
DELETE 6
FIND 6
```

Contoh Keluaran

```
6 inserted
3 inserted
9 inserted
4 inserted
4 found with path: 6 3 4
6 deleted
6 not found
```

Penjelasan Contoh

Perintah 1-4:

INSERT 6
INSERT 3
INSERT 9
INSERT 4

Struktur tree saat ini:

Perintah 5:

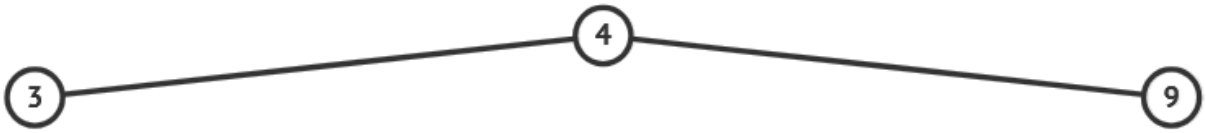
FIND 4

Dapat dilihat dari gambar tree sebelumnya, node 4 dapat ditemukan dengan melakukan traversal dari root **6**, ke kiri menuju **3**, dan ke kanan menuju **4**.

Perintah 6:

DELETE 6

Struktur AVL saat ini:



Perintah 7:

FIND 6

Dapat dilihat dari gambar tree sebelumnya, node 6 tidak ditemukan karena sudah dihapus di perintah sebelumnya.

AVL Logger

by: Arya Sudewa

Time Limit: 1 s
Memory Limit: 256 MB

Description

Create an AVL Tree with a logging system, where each operation generates output according to the specified format.

You are given **N** commands. Each command is one of the following three operations:

1. **INSERT X**: Insert the integer **X** into the tree if it does not already exist.
2. **DELETE X**: Remove the integer **X** from the tree if it exists.
3. **FIND X**: Search for the integer **X** in the tree. If found, display the traversal path from the root to the node **X**.

Constraints

- $1 \leq \mathbf{N} \leq 10^5$
- $-10^9 \leq \mathbf{X} \leq 10^9$

Input Format

The first line contains a single integer **N**, the number of commands.

The next **N** lines each contain a command in one of the following formats: **INSERT X**, **DELETE X**, or **FIND X**.

Output Format

1. For **INSERT X**:
   - If **X** is not in the tree, output [**X inserted**].
   - If **X** is already in the tree, output [**X is already in the tree**].
2. For **DELETE X**:
   - If **X** exists in the tree, output [**X deleted**].
   - If **X** is not in the tree, output [**X is not found in the tree**].
3. For **FIND X**:
   - If **X** is found, output [**X found with path:**] followed by the traversal path from the root to the node **X**.
   - If **X** is not found, output [**X not found**].

Sample Input

```
7
INSERT 6
INSERT 3
INSERT 9
INSERT 4
FIND 4
DELETE 6
FIND 6
```

Sample Output

```
6 inserted
3 inserted
9 inserted
4 inserted
4 found with path: 6 3 4
6 deleted
6 not found
```

Explanation

Commands 1-4:

INSERT 6
INSERT 3
INSERT 9
INSERT 4

Current tree structure:

Command 5:

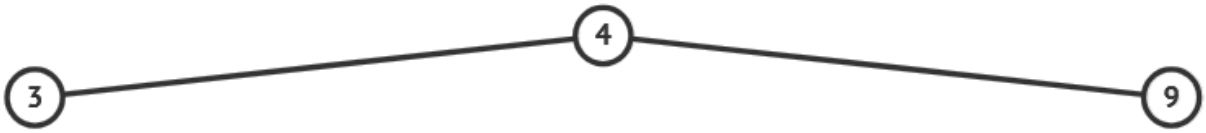
FIND 4

As seen in the tree structure above, node 4 can be found by traversing from the root **6**, going left to **3**, and then right to **4**.

Command 6:

DELETE 6

Current tree structure:



Command 7:

FIND 6

As seen in the tree structure above, node 6 cannot be found because it was deleted in the previous command.