

Compile

```
shuqiny2@circinus-46 23:51:25 ~/253P/HW5/src
$ javac AVLTreeImpl.java
shuqiny2@circinus-46 23:51:32 ~/253P/HW5/src
$ java AVLTreeImpl
sample input
```

input:

```
insert 50
insert 25
insert 10
insert 5
insert 7
insert 3
insert 30
insert 20
insert 8
insert 15
find 10
find 12
delete 4
delete 20
find 22
delete 50
find 30
delete 10
find 7
```

output:

```
50 (inserted)
50 25 (inserted)
50 25 10 (inserted)
25 10 5 (inserted)
25 10 5 7 (inserted)
25 7 5 3 (inserted)
7 25 50 30 (inserted)
7 25 10 20 (inserted)
7 25 10 8 (inserted)
7 25 10 20 15 (inserted)
10 (found)
10 25 20 15 (not found!)
10 7 5 3 (not found!)
10 25 20 (deleted)
10 25 15 (not found!)
10 25 50 (deleted)
10 25 30 (found)
10 (deleted)
8 5 7 (found)
```

```
50 (inserted)
50 25 (inserted)
50 25 10 (inserted)
25 10 5 (inserted)
25 10 5 7 (inserted)
25 7 5 3 (inserted)
7 25 50 30 (inserted)
7 25 10 20 (inserted)
7 25 10 8 (inserted)
7 25 10 20 15 (inserted)
10 (found)
10 25 20 15 (not found!)
10 7 5 3 (not found!)
10 25 20 (deleted)
10 25 15 (not found!)
10 25 50 (deleted)
10 25 30 (found)
10 (deleted)
8 5 7 (found)
```

input:

```
insert 1
insert 2
insert 3
insert 4
insert 8
insert 7
insert 6
insert 5
find 6
find 9
delete 4
delete 20
find 2
delete 5
find 3
```

output:

```
1 (inserted)
1 2 (inserted)
1 2 3 (inserted)
2 3 4 (inserted)
2 3 4 8 (inserted)
2 4 8 7 (inserted)
2 4 8 7 6 (inserted)
4 7 6 5 (inserted)
3 4 7 6 (found)
3 4 7 8 (not found!)
3 4 (deleted)
2 3 7 8 (not found!)
3 2 (found)
3 6 5 (deleted)
3 (found)
```

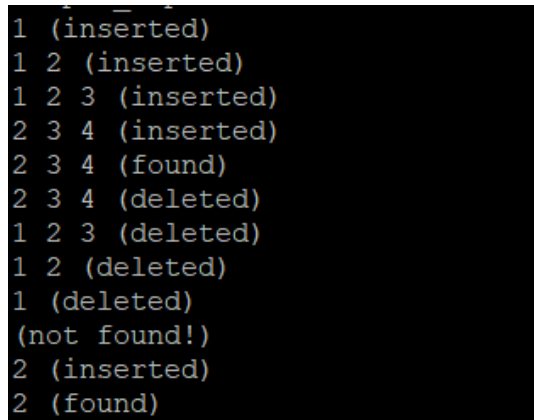
```
1 (inserted)
1 2 (inserted)
1 2 3 (inserted)
2 3 4 (inserted)
2 3 4 8 (inserted)
2 4 8 7 (inserted)
2 4 8 7 6 (inserted)
4 7 6 5 (inserted)
3 4 7 6 (found)
3 4 7 8 (not found!)
3 4 (deleted)
2 3 7 8 (not found!)
3 2 (found)
3 6 5 (deleted)
3 (found)
3 2 1 (deleted)
```

input:

```
insert 1
insert 2
insert 3
insert 4
find 4
delete 4
delete 3
delete 2
delete 1
find 2
insert 2
find 2
```

output:

```
1 (inserted)
1 2 (inserted)
1 2 3 (inserted)
2 3 4 (inserted)
2 3 4 (found)
2 3 4 (deleted)
1 2 3 (deleted)
1 2 (deleted)
1 (deleted)
(not found!)
2 (inserted)
2 (found)
```



```
1 (inserted)
1 2 (inserted)
1 2 3 (inserted)
2 3 4 (inserted)
2 3 4 (found)
2 3 4 (deleted)
1 2 3 (deleted)
1 2 (deleted)
1 (deleted)
(not found!)
2 (inserted)
2 (found)
```

Leetcode 814

LeetCode

Explore

Problems

Mock

Contest

Discuss

Store

Premium

+

🔔

👤

Description

Solution

Discuss (706)

Submissions

Success

Details >

Runtime: 0 ms, faster than 100.00% of Java online submissions for Binary Tree Pruning.




Memory Usage: 36.6 MB, less than 8.59% of Java online submissions for Binary Tree Pruning.

Next challenges:

Binary Tree Longest Consecutive Sequence

Univalued Binary Tree

Pseudo-Palindromic Paths in a Binary Tree

Show off your acceptance:   

Time Submitted	Status	Runtime	Memory	Language
11/11/2020 22:53	Accepted	0 ms	36.6 MB	java
11/11/2020 22:26	Wrong Answer	N/A	N/A	java

Problems

Pick One

< Prev

814/1649

Next >

Java

Autocomplete

```
11 * this.left = left;
12 * this.right = right;
13 * }
14 * }
15 */
16 class Solution {
17 public TreeNode pruneTree(TreeNode root) {
18     if (root == null)
19         return root;
20
21     root.left = pruneTree(root.left);
22     root.right = pruneTree(root.right);
23     if (root.val == 1)
24         return root;
25     if (root.left == null && root.right == null) // leaf node 0
26         return null;
27
28     return root;
29 }
30 }
```

Testcase

Run Code Result

Debugger

Accepted

Runtime: 0 ms

Your input

[1,null,0,0,1]

[1,0,1,0,0,1]

...

Output

[1,null,0,null,1]

[1,null,1,null,1]

...

Diff

Expected

[1,null,0,null,1]

[1,null,1,null,1]

...

Console

How to create a testcase

Run Code

Submit

Leetcode 310

LeetCode

Explore

Problems

Mock

Contest

Discuss

Store

Premium

LeetCode

Explore

Problems

Mock

Contest

Discuss

Store

Success

Details

Runtime: 10 ms, faster than 86.09% of Java online submissions for Minimum Height Trees.

Memory Usage: 43 MB, less than 6.51% of Java online submissions for Minimum Height Trees.

Next challenges:

Course ScheduleCourse Schedule II

Show off your acceptance:

Time Submitted	Status	Runtime	Memory	Language
11/11/2020 21:58	Accepted	10 ms	43 MB	java
11/11/2020 21:57	Wrong Answer	N/A	N/A	java

Problems

Pick One

< Prev

310/1649

Next >

Java

Autocomplete

```
11 List<List<Integer>> graph = new ArrayList<>();
12 int[] degree = new int[n];
13 for (int i = 0; i < n; i++)
14     graph.add(new ArrayList<>());
15 for (int i = 0; i < edge.length; i++) {
16     graph.get(edge[i][0]).add(edge[i][1]);
17     graph.get(edge[i][1]).add(edge[i][0]);
18     degree[edge[i][0]]++;
19     degree[edge[i][1]]++;
20 }
21
22
23 for (int i = 0; i < n; i++)
24     if(degree[i] == 1) // leaf node
25         leaf.add(i);
26
27 // work like topo
28 // remove leaf nodes and the leftover is/are root
29 while(n > 2) {
30     n -= leaf.size();
31     List<Integer> tmp = new ArrayList<>();
32     for (int u: leaf){
33         for (int v: graph.get(u)) {
34             degree[v]--;
35             if(degree[v] == 1)
36                 tmp.add(v);
37         }
38     }
39     leaf = List.copyOf(tmp);
40 }
41
42 return leaf;
43
44
45 }
```

Your previous code was restored from your local storage. [Reset to default](#)

Console

Contribute

Run Code

Submit