

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)
```

```
shuqiny2@circinus-4 10:34:40 ~/253P/HW5/src
$ java AVLTreeImpl
sample_input
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
delete 1
```

output:

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

```
shuqiny2@circinus-4 10:34:53 ~/253P/HW5/src
$ java AVLTreeImpl
sample_input2
1 (inserted)
1 2 (inserted)
1 2 3 (inserted)
2 3 4 (inserted)
2 3 4 8 (inserted)
2 4 8 7 (inserted)
4 8 7 6 (inserted)
4 7 6 5 (inserted)
4 7 6 (found)
4 7 8 (not found!)
4 (deleted)
3 7 8 (not found!)
3 2 (found)
3 7 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)
2 3 (deleted)
2 (deleted)
1 (deleted)
(not found!)
2 (inserted)
2 (found)
```

```
shuqiny2@cirrus-4 10:35:05 ~/253P/HW5/src
$ java AVLTreeImpl
sample_input3
1 (inserted)
1 2 (inserted)
1 2 3 (inserted)
2 3 4 (inserted)
2 3 4 (found)
2 3 4 (deleted)
2 3 (deleted)
2 (deleted)
1 (deleted)
(not found!)
2 (inserted)
2 (found)
```

## Leetcode 814

LeetCode

ExploreProblemsMockContestDiscussStore

☆ Premium

+

🔔

👤

DescriptionSolutionDiscuss (706)Submissions

SuccessDetails >




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 SequenceUnivalued 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

ProblemsPick One< Prev814/1649Next >

JavaAutocomplete

```
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 }
```

TestcaseRun Code ResultDebugger

AcceptedRuntime: 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]

...

ConsoleHow to create a testcaseRun CodeSubmit

## Leetcode 310

LeetCode

Explore

Problems

Mock

Contest

Discuss

Store

Premium

🔍

🔔

👤

Description

Solution

Discuss (435)

Submissions

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

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

```
List<List<Integer>> graph = new ArrayList<>();
int[] degree = new int[n];
for (int i = 0; i < n; i++)
    graph.add(new ArrayList<>());
for (int i = 0; i < edge.length; i++) {
    graph.get(edge[i][0]).add(edge[i][1]);
    graph.get(edge[i][1]).add(edge[i][0]);
    degree[edge[i][0]]++;
    degree[edge[i][1]]++;
}

for (int i = 0; i < n; i++)
    if(degree[i] == 1) // leaf node
        leaf.add(i);

// work like topo
// remove leaf nodes and the leftover is/are root
while(n > 2) {
    n -= leaf.size();
    List<Integer> tmp = new ArrayList<>();
    for (int u: leaf){
        for (int v: graph.get(u)) {
            degree[v]--;
            if(degree[v] == 1)
                tmp.add(v);
        }
    }
    leaf = List.copyOf(tmp);
}

return leaf;
}
```

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

Console

Contribute i

Run Code

Submit