```
1
    /*Author: Bochen (mddboc@foxmail.com)
2
    Last Modified: Tue Apr 10 22:28:44 CST 2018*/
3
4
    /*Given a binary tree, determine if it is height-balanced.
5
6
    For this problem, a height-balanced binary tree is defined as:
7
8
     seeds a binary tree in which the depth of the two subtrees of every node never
            differ by more than 1.
9
10
     Example 1:
11
12
     Given the following tree [3,9,20,null,null,15,7]:
13
14
           . . / . \
15
           9 1 20
16
17
           . . / . . . .
18
         15 - 7
     Return true.
19
20
21
    Example 2:
22
23
     Given the following tree [1,2,2,3,3,null,null,4,4]:
24
25
     . . . . . . . . 1
26
     . . . . . . . . 2 . . . 2
27
     . . . . . . . . / . \
28
     . . . . . . . . . 3 . .
29
30
     31
     . . . . . . . . 4 . . . 4
32
     Return false.*/
33
34
35
    import java.util.*;
36
37
38
   class TreeNode {
39
    · · · int val;
40
        TreeNode left;
41
     TreeNode right;
42
    TreeNode(int x) {
43
44
           val = x;
45
       • • }
46
    }
47
48
    public class Test {
49
     public static void main(String[] args) {
50
     TreeNode root = new TreeNode(3);
51
52
     root.left = new TreeNode(9);
53
     root.right = new TreeNode(20);
     root.right.left = new TreeNode(15);
54
55
     root.right.right = new TreeNode(7);
56
57
    new Solution().isBalanced(root);
58
    · · · · · }
59
    }
60
61
62
    class Solution {
63
64
    public boolean isBalanced(TreeNode root) {
65
66
     if (root == null) {
67
               return true;
68
    69
70
     return isBalancedHelper(root);
71
     . . . . . }
```

```
73
     private boolean isBalancedHelper(TreeNode root) {
74
75
     if (root.left == null && root.right == null) {
76
                 return true;
77
     } else if (root.left == null && root.right != null) {
78
                 return treeHeight(root.right) == 1;
79
      } else if (root.left !=null && root.right == null) {
80
                 return treeHeight(root.left) == 1;
81
      ----} else {
        int leftHeight = treeHeight(root.left);
82
83
                int rightHeight = treeHeight(root.right);
84
85
      if (Math.abs(leftHeight - rightHeight) <= 1) {</pre>
86
87
                     return isBalancedHelper(root.left) && isBalancedHelper(root.right);
88
89
                 } else {
90
                     return false;
      · · · · · · · · · }
91
                 }
92
     . . . . }
93
94
95
     private int treeHeight(TreeNode root) {
96
97
     if (root == null) {
98
                 return 0;
99
100
101
     return 1 + Math.max(treeHeight(root.left), treeHeight(root.right));
102
     . . . . . }
103
     }
```