```java
/*Author: Bochen (mddboc@foxmail.com)
Last Modified: Tue Apr 10 22:28:44 CST 2018*/

/*Given a binary tree and a sum, determine if the tree has a root-to-leaf path such that adding up all the values along the path equals the given sum.

        For example:
        Given the below binary tree and sum = 22,

        5
        / \
        4   8
        /   / \
        11  13  4
        /  \      \
        7    2      1
        return true, as there exist a root-to-leaf path 5->4->11->2 which sum is 22.*/


import java.util.*;


class TreeNode {
    int val;
    TreeNode left;
    TreeNode right;

    TreeNode(int x) {
        val = x;
    }
}

public class Test {
    public static void main(String[] args) {

        TreeNode root = new TreeNode(3);
        root.left = new TreeNode(9);
        root.right = new TreeNode(20);
        root.right.left = new TreeNode(15);
        root.right.right = new TreeNode(7);

        new Solution().isBalanced(root);
    }
}


class Solution {
    public boolean hasPathSum(TreeNode root, int sum) {

        if (root == null) {
            return false;
        }

        return hasPathSumHelper(root, sum);
    }

    private boolean hasPathSumHelper(TreeNode root, int sum) {

        if (root.left == null && root.right == null) {
            return root.val == sum;
        } else if (root.left == null && root.right != null) {
            return hasPathSumHelper(root.right, sum - root.val);
        } else if (root.left != null && root.right == null) {
            return hasPathSumHelper(root.left, sum - root.val);
        } else {
            return hasPathSumHelper(root.left, sum - root.val)
                    || hasPathSumHelper(root.right, sum - root.val);
        }
    }
}
```