题目： 二叉排序树的判定

问题描述

给定一个二叉树，判断其是否是一个有效的二叉排序树。

假设一个二叉排序树具有如下特征：

结点的左子树只包含小于当前结点的数。

结点的右子树只包含大于当前结点的数。

所有左子树和右子树自身必须也是二叉排序树。

例如：

输入：

5

/ \

1 3

/ \

4 6

输出: false

二叉树结点定义如下，如果使用其他语言，其二叉树结点定义类似：

/\*\*

\* C++

\* struct TreeNode {

\* int val;

\* TreeNode \*left;

\* TreeNode \*right;

\* TreeNode(int x) : val(x), left(NULL), right(NULL) {}

\* };

\*/

# Python

# class TreeNode(object):

# def \_\_init\_\_(self, x):

# self.val = x

# self.left = None

# self.right = None

输入格式

第一行两个数n,root，分别表示二叉树有n个结点，第root个结点是二叉树的根。接下来共n行，第i行三个数val\_i、left\_i、right\_i，分别表示第i个结点的值val是val\_i，左儿子left是第left\_i个结点，右儿子right是第right\_i个结点。

节点0表示空。

1<=n<=100000,保证是合法的二叉树

输出格式

输出"true"如果给定二叉树是二叉排序树，否则输出"false"

样例输入

5 1

5 2 3

1 0 0

3 4 5

4 0 0

6 0 0

样例输出

false

样例说明

该样例对应的二叉树即题目描述中的二叉树。