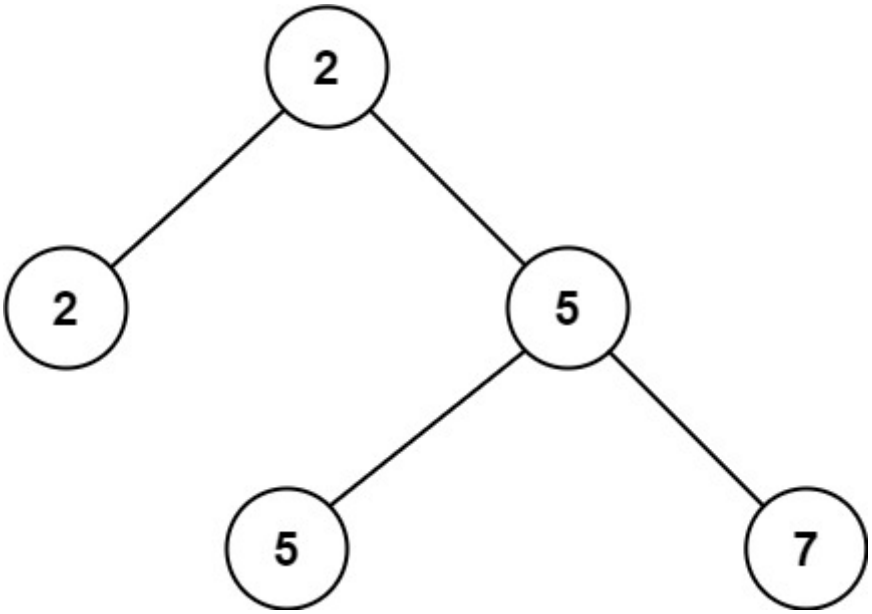
二叉树中第二小的节点

给定一个非空特殊的二叉树，每个节点都是正数，并且每个节点的子节点数量只能为 2 或 0。如果一个节点有两个子节点的话，那么该节点的值等于两个子节点中较小的一个。

更正式地说，root.val = min(root.left.val, root.right.val) 总成立。

给出这样的一个二叉树，你需要输出所有节点中的**第二小的值**。如果第二小的值不存在的话，输出 -1 。

示例 1：

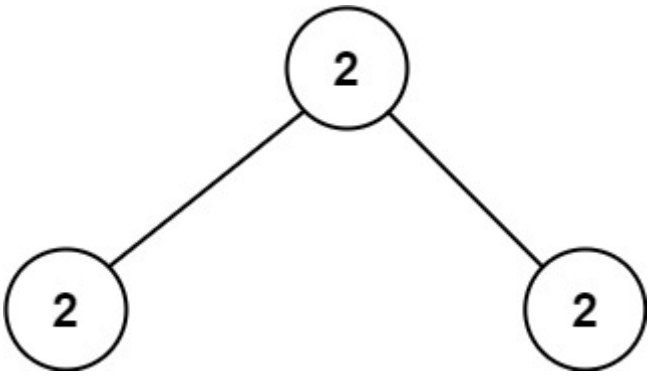


输入：root = [2,2,5,null,null,5,7]

输出：5

解释：最小的值是 2 ，第二小的值是 5 。

示例 2：



输入：root = [2,2,2]

输出：-1

解释：最小的值是 2，但是不存在第二小的值。

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     struct TreeNode *left;
 *     struct TreeNode *right;
 * };
 */
#define MAX_INPUT_NUM 1024
int gNum[MAX_INPUT_NUM] = { 0 };
int gCount = 0;

void Find(struct TreeNode* root)
{
    if (!root) {
        return;
    }
    gNum[gCount++] = root->val;
    Find(root->left);
    Find(root->right);
}
int Compare(const void *a, const void *b)
{
    return *(int *)a - *(int *)b;
}
int findSecondMinimumValue(struct TreeNode* root)
{
    gCount = 0;
    memset(gNum, 0, MAX_INPUT_NUM * sizeof(int));
    Find(root);
    if (gCount <= 1) {
        return -1;
    }
    qsort(gNum, gCount, sizeof(int), Compare);
    int i;
    for (i = 0; i < gCount - 1; i++) {
        if (gNum[i] != gNum[i + 1]) {
            return gNum[i + 1];
        }
    }
    if (i == gCount - 1) {
        return -1;
```

```
    }
    return 0;
}
```