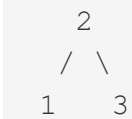
给定一个二叉树，在树的最后一行找到最左边的值。

示例 1:

输入：

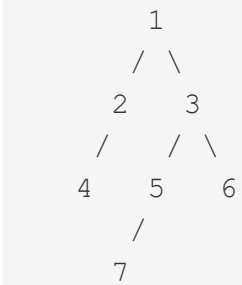


输出：

```
1
```

示例 2:

输入：



输出：

```
7
```

思路，深度遍历，先遍历左子树，再遍历又子树，记录最大的树的度，若是当前的度大于最大度，则左边的值更新，若是不大于，则继续遍历。

代码

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    void dfs(TreeNode* root,int &left,int dc,int &sd){
        if(dc>sd){
            left=root->val;
```

```
            sd=dc;
        }
        if(root->left!=NULL){
            dfs(root->left,left,dc+1,sd);
        }
        if(root->right!=NULL){
            dfs(root->right,left,dc+1,sd);
        }
    }
    int findBottomLeftValue(TreeNode* root) {
        int sd=0;
        int left;
        dfs(root,left,1,sd);
        return left;
    }
};
```