题目：给定一棵二叉树和其中的一个节点，如何找出中序遍历序列的下一个节点？树中的节点除了有两个分别指向左、右子节点的指针，还有一个指向父节点的指针。

解法一 有指向父节点的指针

如果该节点有右子树，则后继结点是右子树上最左的节点

如果该节点没有右子树，则向上找，直到某一节点是其父节点的左孩子，后继结点是当前的父节点

```
TreeNode* getNext(TreeNode* node)
{
    if(!node)
        return nullptr;
    TreeNode* res;
    if(node->right)
    {
        TreeNode* tmp=node->right;
        while(tmp->left)
        {
            tmp=tmp->left;
        }
        res=tmp;
    }
    else
    {
        TreeNode* parent=node->next;
        while(parent && node!=parent->left)
        {
            node=node->next;
            parent=parent->next;
        }
        res=parent;
    }
    return res;
}
```

解法二 题目没有指向父节点的指针

利用栈完成中序遍历，保存上一次出栈节点和当前出栈节点，如果上一次出栈节点是题目给定节点，则当前出栈节点是它的后继结点

```
TreeNode* getNext(TreeNode* root,TreeNode* node)
{
    if(!root || !node)
        return nullptr;
    stack<TreeNode*> s;
    TreeNode*pre,*cur=nullptr;
    while(!s.empty() || root)
    {
        if(root)
        {
            s.push(root);
            root=root->left;
        }
        else
        {
            pre=cur;
            cur=s.top();
            s.pop();
            if(pre==node)
            {
                break;
            }
            root=root->right;
        }
    }
    return pre==node? cur:nullptr;
}
```