二叉树的下一个节点getNextNode

# 二叉树的下一个节点

## 题目

给定一个二叉树和其中的一个结点，请找出**中序遍历**顺序的下一个结点并且返回。

**注意**，树中的结点不仅包含左右子结点，同时包含指向父结点的指针。

二叉树的数据结构：

public class TreeLinkNode {

int val;

TreeLinkNode left = null;

TreeLinkNode right = null;

TreeLinkNode next = null;//指向父节点

TreeLinkNode(int val) {

this.val = val;

}

}

## 思路分析

分成三个大类：

### 当前节点为null，则返回null；

### 当前节点有右子树，则返回右子树的最左子节点；

### 当前节点无右子树，分成两种情况：①当前节点为左子节点，则直接返回当前节点的父节点；②当前节点不是左子节点(也就是右子节点)，则不断向上寻找父节点，直到父节点为左子节点为止，返回该父节点的父节点，若寻找过程中父节点为null，则直接返回null。

## Java代码

/\*\*

\* 获取二叉树的下一个节点

\*/

public TreeLinkNode getNext(TreeLinkNode pNode){

//分成三种情况：当前节点①为null;②有右子树;③无右子树，再根据左子节点和右子节点分成两类。

**if(pNode == null) return null;**

if(pNode.right != null){//有右子树

pNode = pNode.right;

while(pNode.left != null){//寻找右子树的最左节点

pNode = pNode.left;

}

return pNode;

}else{//没有右子树

TreeLinkNode parent = pNode.next;//父节点

**if(parent == null) return null;//特殊情况:只有一个根节点**

if(pNode == parent.left){//当前节点为左子节点

return parent;

}else{//当前节点为右子节点

//中序遍历的最后一个元素，会出现parent.next 为null，需要返回null

**while(parent.next != null && parent != parent.next.left){**

**parent = parent.next;**

**}**

return parent.next;

}

}

}