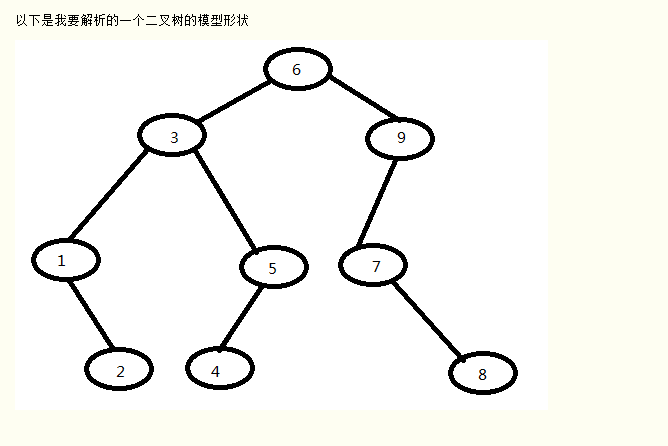
**题目描述**

操作给定的二叉树，将其变换为源二叉树的镜像。



**package** facehandjava.tree;  
  
**public class** MirrorTree {  
 **public static** Node init() {*//注意必须逆序建立，先建立子节点，再逆序往上建立，因为非叶子结点会使用到下面的节点，而初始化是按顺序初始化的，不逆序建立会报错* Node J = **new** Node(8, **null**, **null**);  
 Node H = **new** Node(4, **null**, **null**);  
 Node G = **new** Node(2, **null**, **null**);  
 Node F = **new** Node(7, **null**, J);  
 Node E = **new** Node(5, H, **null**);  
 Node D = **new** Node(1, **null**, G);  
 Node C = **new** Node(9, F, **null**);  
 Node B = **new** Node(3, D, E);  
 Node A = **new** Node(6, B, C);  
 **return** A; *//返回根节点* }  
  
 **public static void** main(String[] args) {  
*// L2RRecursiveBinaryTree tree = new L2RRecursiveBinaryTree();* Node root = MirrorTree.*init*();  
 System.***out***.println(**"原型树从左到右打印"**);  
 System.***out***.println(**"639157248"**);  
 System.***out***.println(**"镜像树从左到右打印"**);  
 *MirrorTree*(root);  
 L2RRecursiveBinaryTree.*L2RRecursiveBinaryTree*(root);  
 }  
  
 **public static void** MirrorTree(Node node) {  
 **if** (node == **null**) {  
 **return**;  
 }  
 **if** (node != **null**) {  
 Node temp = node.getRightNode();  
 node.setRightNode(node.getLeftNode());  
 node.setLeftNode(temp);  
 }  
 *MirrorTree*(node.getLeftNode());  
 *MirrorTree*(node.getRightNode());  
 }  
  
}