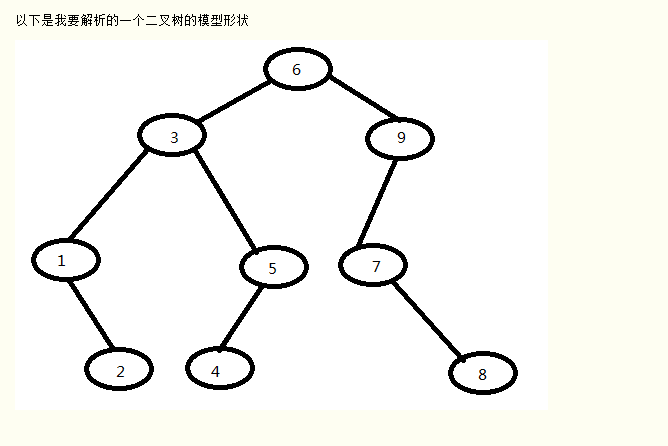
分四步走   
（1）确定树的根节点。树根是当前树中所有元素在先序遍历中最先出现的元素，即先序遍历的第一个结点就是二叉树的跟   
（2）求解树的子树。找到根在中序遍历的位置，位置左边就是二叉树的左孩子，位置右边是二叉树的右孩子，如果根结点左边或右边为空，那么该方向子树为空；如果根节点左边和右边都为空，那么根节点已经为叶子节点。   
（3）对二叉树的左、右孩子分别进行步骤（1）（2），直到求出二叉树的结构为止。

（4）后序遍历二叉树。



**package** facehandjava.tree;  
  
**import** java.util.Stack;  
  
**public class** RebuildTree {  
 **public static void** main(String[] args) {  
 **int**[] before = {6, 3, 1, 2, 5, 4, 9, 7, 8};  
 **int**[] middle = {1, 2, 3, 4, 5, 6, 7, 8, 9};  
 Node rebuildTree = *RebuildTree*(before, middle);  
 *afterTree*(rebuildTree);  
 }  
  
 **public static** Node RebuildTree(**int**[] before, **int**[] middle) {  
 **int** l = before.**length**;  
 **if** (l != middle.**length** || l == 0) {  
 **return null**;  
 }  
 **int** i = 0;  
 **while** (before[0] != middle[i]) {  
 i++;  
 }  
 Node node = **new** Node(before[0], **null**, **null**);  
 **int**[] b\_left = **new int**[i];  
 **int**[] m\_left = **new int**[i];  
 **int**[] b\_right = **new int**[l - i - 1];  
 **int**[] m\_right = **new int**[l - i - 1];  
 **for** (**int** j = 0; j < l; j++) {  
 **if** (j < i) {  
 b\_left[j] = before[j + 1];  
 m\_left[j] = middle[j];  
 } **else if** (j > i) {  
 b\_right[j - i - 1] = before[j];  
 m\_right[j - i - 1] = middle[j];  
 }  
 }  
 node.setLeftNode(*RebuildTree*(b\_left, m\_left));  
 node.setRightNode(*RebuildTree*(b\_right, m\_right));  
 **return** node;  
 }  
  
*// public static void afterTree(Node node) {  
// if (node == null) {  
// return;  
// }  
// afterTree(node.getLeftNode());  
// afterTree(node.getRightNode());  
// System.out.print(node.getData());  
// }* **public static void** afterTree(Node node) {  
 Stack<Node> stack = **new** Stack<>();  
 Stack<Node> out = **new** Stack<>();  
 **while** (node != **null** || !stack.isEmpty()) {  
 **if** (node != **null**) {  
 stack.push(node);  
 out.push(node);  
 node = node.getRightNode();  
 } **else** {  
 node = stack.pop();  
 node = node.getLeftNode();  
 }  
 }  
 **while** (out.size() > 0) {  
 System.***out***.print(out.pop().getData());  
 }  
 }  
  
}