# 剑指offer第17题

```java
package niuke;

import java.util.ArrayList;
import java.util.Collections;


/**
* @author tfxidian E-mail: tfxidian@163.com
* @version 创建时间：2018年10月12日 下午3:56:01
* 类说明
* 给出A和B， 判断B是不是A的子结构
*/
public class SubTree {

    public static class TreeNode {
        int val = 0;
        TreeNode left = null;
        TreeNode right = null;

        public TreeNode(int val) {
            this.val = val;

        }

    }

    public static void main(String[] args) {
        TreeNode nodes[] = new TreeNode[9];
        TreeNode nodes2[] = new TreeNode[5];
        for (int i = 0; i < nodes.length; i++) {
            nodes[i] = new TreeNode(i);
        }
        for (int i = 0; i < nodes.length-1; i++) {
            nodes[i].left = nodes[i+1];
        }

        for (int i = 0; i < nodes2.length; i++) {
            nodes2[i] = new TreeNode(i+4);
        }
        for (int i = 0; i < nodes2.length-1; i++) {
            nodes2[i].right = nodes2[i+1];
        }

        TreeNode root1 = nodes[0];
        TreeNode root2 = nodes2[0];
        boolean result = HasSubtree(root1, root2);
    }

    public static boolean HasSubtree(TreeNode root1,TreeNode root2) {
        if (root1==null||root2 ==null) {
            return false;
        }
        boolean result =true;
        TreeNode node = root1;
        ArrayList<Integer> aList = new ArrayList<Integer>();
        aList = preTreee(aList, root1);
        System.out.println(aList);

        ArrayList<Integer> aList2 = new ArrayList<Integer>();
        aList2 = preTreee(aList2, root2);
        System.out.println(aList2);
        int subIndex = Collections.indexOfSubList(aList, aList2);
        System.out.println(subIndex);
        if (subIndex == -1) {
            result = false;
        }

        return result;
//        while (node!=null) {
//          System.out.println(node.val);
//          node = node.left;
//      }
//
//        node = root2;
//        while (node!=null) {
//          System.out.println(node.val);
//          node = node.right;
//      }
//        int root2Value = root2.val;
//        System.out.println("test root2value");
//        System.out.println(root1.val);
//
//        System.out.println(root2Value);
//        node = findEqualNode(root1, root2Value);


     }

    public static TreeNode findEqualNode(TreeNode root1, int root2Value){

        if (root2Value == root1.val) {
            System.out.println("-----------------------------");
            System.out.println(root1.val);
            return root1;
        }else {
            if (root1.left!=null) {
                System.out.println(root1.left.val);
                findEqualNode(root1.left, root2Value);
            }
            if (root1.right!=null) {
                System.out.println(root1.right.val);
                findEqualNode(root1.right, root2Value);
            }
        }
        return null;
    }

    public static ArrayList<Integer> preTreee(ArrayList<Integer> integers, TreeNode root) {
        integers.add(root.val);
        if (root.left!= null) {
            preTreee(integers, root.left);
        }
        if (root.right!=null) {
            preTreee(integers, root.right);
        }

        return integers;
    }


}


```