```c
#include <stdio.h>
#include <stdlib.h>

#define MAXN 1024

char PreOrder[MAXN];//"ABDECGHI"    示例1的前序："ABCDEFSGHIJKLMNO"
char inOrder[MAXN];//"DBEACHGI"     示例1的中序："BCDASFHIJGEMNOLK"
char postOrder[MAXN];//"DEBHIGCA"

typedef struct binaryTreeNode//二叉树结构
{
    char data;//节点值
    struct binaryTreeNode *lchild;//左子节点
    struct binaryTreeNode *rchild;//右子节点
    //根据需要加入其它字段
    //int level;//层次

}BiNODE; //树的结构


typedef struct LRtagBiTree//左右标志表示树节点
{
    int ltag;
    char data;
    int rtag;
}LRBTree;

/*
13
```

```
0 A 0
0 B 0
1 C 1
0 D 0
1 E 1
1 F 0
1 G 1
0 H 1
0 I 0
0 J 0
1 K 1
1 L 1
1 M 1
*/

BiNODE * initial()//树节点初始化
{
    BiNODE* node;
    node=malloc(sizeof(BiNODE));
    node->lchild=NULL;
    node->rchild=NULL;
    //node->level=0;
    //node->parent='\0';
    //node->data='\0';
    return node;
}

BiNODE* BiTreeFromInPre(char* inorder, char* preorder, int
length)//根据前序中序生成子树、
```

```
{

}

BiNODE* BiTreeFromInPost(char* inorder, char* postorder, int
length)//根据前序中序生成子树
{

}

BiNODE* BinaryTreeFromLRtag()
{
}

//前序，中序，后序 的递归和非递归遍历
// 用队列实现层次


//1.    （选做）使用
上机练习补充（一）中的方法建立若干棵m次树。每棵树根结点的指针构成链表，用这个链表
表示森林。将森林转换为二叉树输出
//2.    将
一棵二叉树转换为森林，用补充练习一中的前序遍历方法分别输出森林中的若干棵树。（可以
用示例1 的树做为输入）

//3.    （选做）穿线排序  输入为一个待排序的整数序列，输出为穿线树的中序序列

//4.    求给定的二叉树的结点的个数
//5.    求给定的二叉树的高度
```

//6.　　判断给定的一棵二叉树是否是满树
//7.　　判断给定的一个二叉树是否是完全二叉树

//8.
如果二叉树的结点的值不允许重复，且是可比较大小的。判断这棵二叉树的每一个子树是否满足：左子树上的结点的值都小于根，右子树上的结点的值都大于根。