# 习题3 树结构

3-1设二叉树T中度为1的结点11个，度为2的结点12个，则二叉树T共有(   C   )个叶子结点。

(A) 11

(B) 12

(C) 13

(D) 36

3-2 设树T的度为4，其中度为1，2，3和4的结点个数分别为4，2，1，1，则T中的叶子数为( D )。

(A) 5

(B) 6

(C) 7

(D) 8

3-3 已知一棵度为k的树中，有n1个度为1的结点，n2个度为2的结点，…，nk个度为k的结点。试计算该树的叶子结点数。

答：非叶子结点个数：n1+n2+...+nk，树的边数：1\*n1+2\*n2+...+k\*nk，树的总结点个数=边数+1= 1\*n1+2\*n2+...+k\*nk+1

因此，叶子结点个数=总结点个数-非叶子结点个数= (1\*n1+2\*n2+...+ k\*nk+1)-(n1 + n2 + ... + nk)

3-4 证明：如果二叉树T的叶子结点数为n0，度为2的结点数为n2，则n0=n2+1。

**证明：**设二叉树T共有n个结点，叶子结点数为n0 ，度=1的结点数为n1 ，度=2的结点数为n2 ，T的分支数为m。

(1)由于二叉树中所有结点的

度≤2，则有 n＝n0+n1+n2

(2)除根结点外，其余结点都有一个分支对应(唯一前驱)，则有 n-1＝m

(3)由于度=i(i=0,1,2)的结点具有i个分支，则有m＝0 + 1×n1+ 2×n2

3-5 对于任意非空二叉树，要设计出其后序遍历的非递归算法而不使用栈结构，最适合的方法是对该二叉树采用( B )存储结构。

(A) 二叉链表

(B) 三叉链表

(C) 索引

(D) 顺序

3-6 一棵二叉树的叶子结点在其先序、中序和后序序列中的相对位置( C )。

(A) 肯定发生变化

(B) 可能发生变化

(C) 不会发生变化

(D) 无法确定

3-7 设二叉树T按照二叉链表存储，则下列递归算法的主要功能是(   **B**   )。

int F(BiTree T)

{

if (!T) return 0;

x=F(T->Lchild);

y=F(T->Rchild);

if (y>x) x=y;

return x+1;

}

(A) 交换二叉树T的左右子树

(B) 计算二叉树T的高度

(C) 计算二叉树T的叶子结点数

(D) 先遍历左子树，再遍历右子树

3-8 已知二叉树T的先序序列为ABCDEF，中序序列为CBAEDF, 则T的后序序列为( **A** )。

(A) CBEFDA

(B) FEDCBA

(C) CBEDFA

(D) 不确定

3-9 简述由先序序列和中序序列构造二叉树的基本操作方法。

答：确定根节点：先序遍历的第一个元素总是二叉树的根节点。这是因为先序遍历的顺序是“根-左-右”。

划分左右子树：利用找到的根节点，在中序遍历序列中定位该根节点的位置。中序遍历的顺序是“左-根-右”，因此，根节点左边的所有元素构成左子树，右边的所有元素构成右子树。

递归：重复上述步骤直到遍历所有节点即构造出所要的二叉树。

3-10 已知二叉树的先序序列为ebadcfhgjik，中序序列为abcdefghijk，试画出该二叉树。

答：二叉树如下：

e

/ \

b f

/ \ \

a d h

/ / \

C g j

/ \

i k

3-11 已知二叉树T的中序序列和后序序列分别为

(中序) 3, 7, 11, 14, 18, 22, 27, 35

(后序) 3, 11, 7, 14, 27, 35, 22, 18

试画出二叉树T。

答：构造出的二叉树如下

18

/ \

14 22

/ \

7 35

/ \ /

3 11 27

3-12 已知二叉树T按照二叉链表存储，设计算法，计算T中叶子结点的数目。

int LeafSum(BTNode \*t)

{

int n1,n2;

if(t==NULL)

return 0;

else if(t->lchild==NULL&&t->rchild==NULL)

return 1;

else

{

n1=LeafSum(t->lchild);

n2=LeafSum(t->rchild);

return n1+n2;

}

}

3-13 已知二叉树T按照二叉链表存储，设计算法，交换T的左子树和右子树。

void ExchangeTree(BiTree \*T)

{

BitNode \*temp;

if (\*T)

{

temp = (\*T)->lchild;

(\*T)->lchild = (\*T)->rchild;

(\*T)->rchild = temp;

ExchangeTree(&((\*T)->lchild));

ExchangeTree(&((\*T)->rchild));

}

}

3-14 先序后继线索化算法是根据二叉链表建立先序后继线索二叉链表，其基本原则是在前驱空指针域中写入后继线索，即将右子树的( B )指针写入左子树的最后一个叶子结点右指针域。

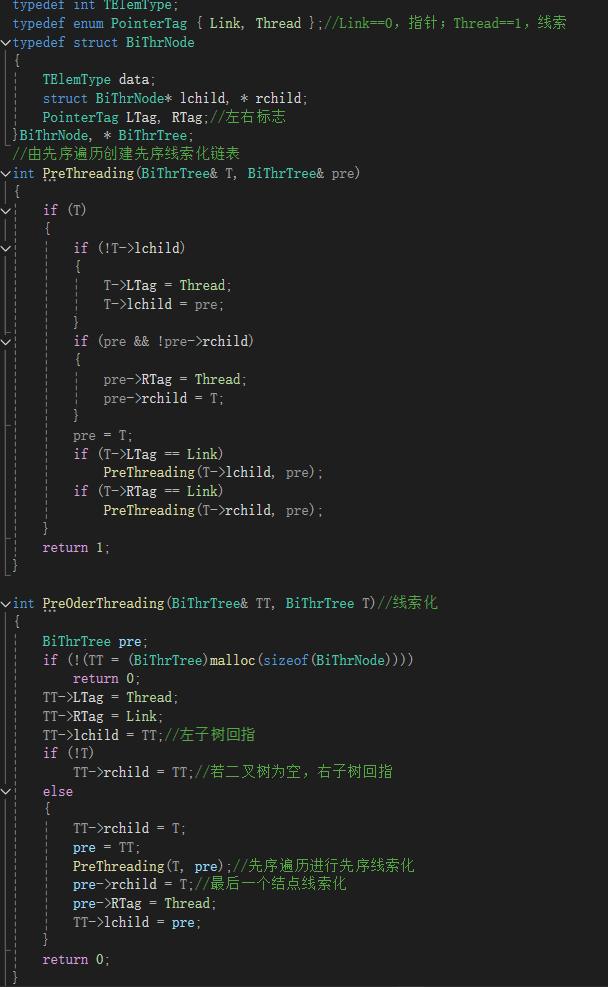
(A) 线索

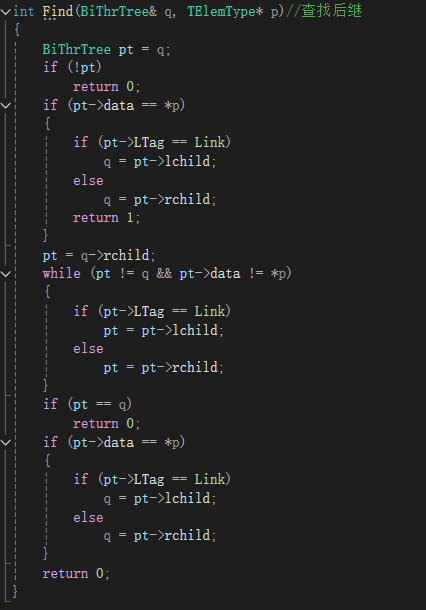
(B) 根结点

(C) 前驱结点

(D) 后继结点

3-15 设计算法，在先序线索二叉树中，查找给定结点p在先序序列中的后继。





3-16对n (n≥2)个权值均不相同的字符构造哈夫曼树T，不正确的叙述( **A** )。

(A) T一定是一棵完全二叉树

(B) T中一定没有度为1的结点

(C) T中两个权值最小的结点一定是兄弟结点

(D) T中任一分支结点的权值一定不小于下一层任一结点的权值

3-17设计一个求结点x在二叉树中的双亲结点算法。

