.第五章 树

树**CCBCA CABBB CCCAB**

**一、思考题**

1、树最适合用来表示（ C ）。

(A) 有序数据元素 (B) 无序数据元素

(C) 元素之间具有分支层次关系的数据 (D) 元素之间无联系的数据

2、二叉树是非线性数据结构，所以 。

（Ａ）它不能用顺序存储结构存储; （Ｂ）它不能用链式存储结构存储;

（Ｃ）顺序和链式存储结构都能存储; （Ｄ）顺序和链式存储结构都不能使用

3、在下列情况中，可称为二叉树的是（ ）。

(A) 每个结点至多有两棵子树的树　 (B) 哈夫曼树

(C) 每个结点至多有两棵子树的有序树 (D) 每个结点只有一棵子树

4、不含任何结点的空树 。

（Ａ）是一棵树; （Ｂ）是一棵二叉树;

（Ｃ）是一棵树也是一棵二叉树; （Ｄ）既不是树也不是二叉树

5、把一棵树转换为二叉树后，这棵二叉树的形态是 。

（Ａ）唯一的 （Ｂ）有多种

（Ｃ）有多种，但根结点都没有左孩子 （Ｄ）有多种，但根结点都没有右孩子

6、二叉树的深度为k，则二叉树最多有（ ）个结点。

(A) 2k (B) 2k-1 (C) 2k-1 (D) 2k-1

7、在一棵具有5层的满二叉树中结点总数为（ ）。

(A) 31 (B) 32 (C) 33 (D) 16

8、将完全二叉树中所有结点按层逐个从左到右的顺序存放在一维数组R[1..N]中，若结点R[i]有右孩子，则其右孩子是（ ）。

(A) R[2i-1] (B) R[2i+1]

(C) R[2i] (D) R[2/i]

9、设a,b为一棵二叉树上的两个结点，在中序遍历时，a在b前面的条件是（ ）。

(A) a在b的右方 (B) a在b的左方

(C) a是b的祖先 (D) a是b的子孙

10、由二叉树的前序和后序遍历序列（ ）惟一确定这棵二叉树。

(A) 能 (B) 不能

11、某二叉树的中序序列为ABCDEFG，后序序列为BDCAFGE，则其左子树中结点数目为（ ）。

(A) 3　 (B) 2 (C) 4 (D) 5

12、若以{4,5,6,7,8}作为权值构造哈夫曼树，则该树的带权路径长度为（ C ）。

(A) 67　 (B) 68 (C) 69 (D) 70

13、按照二叉树的定义，具有3个结点的二叉树有（ ）种。

(A) 3　 (B) 4 (C) 5 (D) 6

14、将一棵有100个结点的完全二叉树从根这一层开始，每一层上从左到右依次对结点进行编号，根结点的编号为1，则编号为49的结点的左孩子编号为（ ）。

(A) 98　 (B) 99 (C) 50 (D) 48

15、对某二叉树进行先序遍历的结果为ABDEFC，中序遍历的结果为DBFEAC，则后序遍历的结果是（ ）。

(A) DBFEAC (B) DFEBCA (C) BDFECA (D) BDEFAC

**二、应用题**

1. 一个人深度为L的满k叉树有如下性质：第L层上的结点均为叶子结点，其余各层上每个结点都有K棵非空子树。如果按层次顺序从1开始对全部结点编号，问：

（1）各层的结点数是多少？

（2）编号为n结点的双亲结点（若存在）的编号是多少？

（3）编号为n的结点的第i个孩子结点（若存在）的编号是多少？

（4）编号为n的结点有右兄弟的条件是什么？其右兄弟的编号是多少？

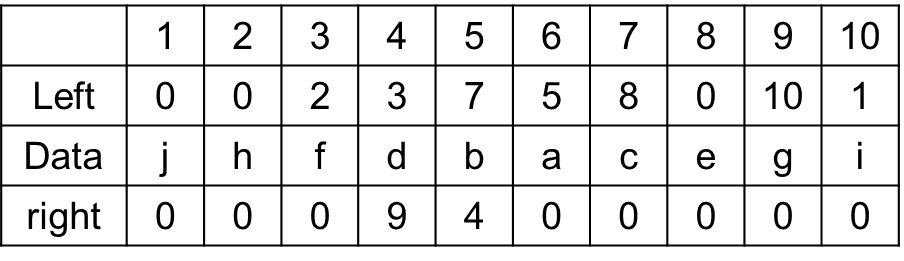
解答·：（1）各层结点数目为Ki-1；

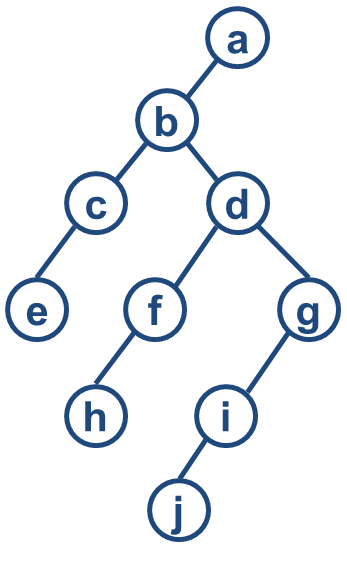
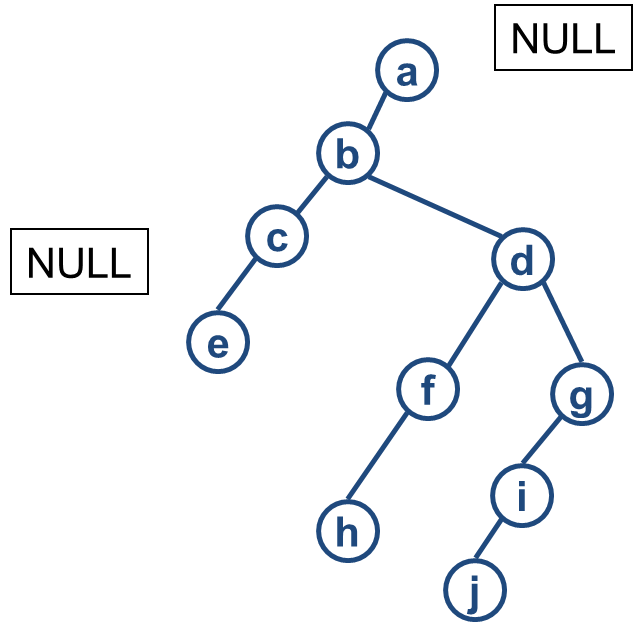
（2）编号为n结点的双亲结点（若存在）的编号是[[n+(k-2)]/n]（n≧1）,n=1时无双亲;

（3）编号为n的结点的第i个孩子结点（若存在）的编号(n-1)×k+i+1；

(4) 编号为n的结点有右兄弟的条件是(n-1)%k≠0 其右兄弟的编号是n+1。

6、已知二叉树存储结构，画出二叉树的逻辑结构及后序线索树



 后序序列：

**echfjigdba**

**线索**

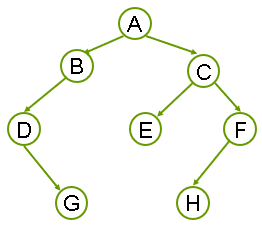
**左前趋(无左子树时)**

**右后继(无右子树时)**

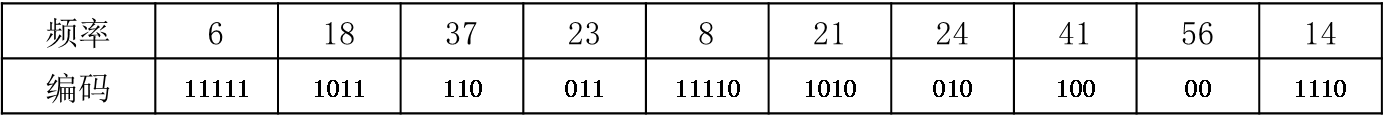
7.已知一棵树的中序序列和后序序列分别为DGBAECHF和GDBEHFCA，画出这棵二叉树。

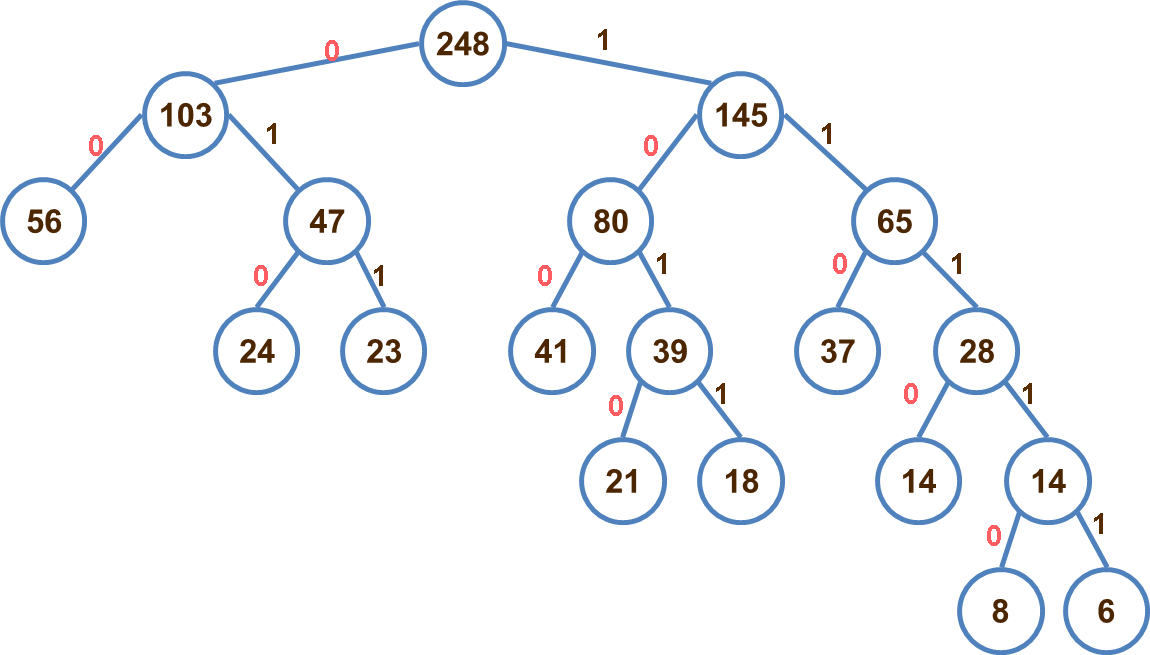
后序：GDBEHFCA

中序： DGB A ECHF



8.假设用于通信的电文由10种不同的符号来 组成，这些符号在电文中出现的频率为8, 21, 37, 24, 6, 18, 23, 41, 56, 14，试为这10个符号设计相应的哈夫曼编码，并求哈夫曼树的WPL。





编码方案：左分支设0，右分支设1

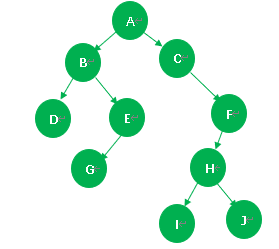
or： 左分支设1，右分支设0

编码：由根到叶子给出0、1序列

WPL=56\*2+24\*3+23\*3+41\*3+21\*4+18\*4+37\*3+14\*4+8\*5+6\*5

= 662

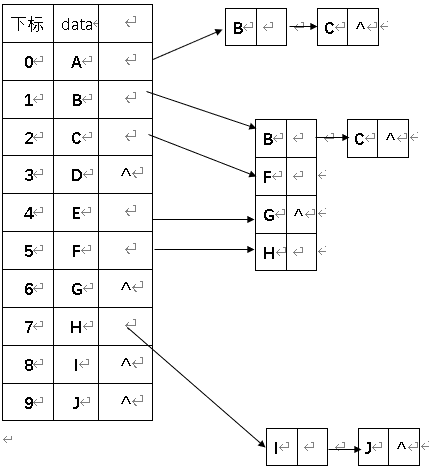
10. 分别画出图5.131中二叉树的顺序存储结构和二叉链表存储结构的图示。并写出该二叉树的前序、中序和后序遍历序列。



顺序存储结构：

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| 数据 | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** |
| LChild | **1** | **3** | **-1** | **-1** | **6** | **7** | **-1** | **8** | **-1** | **-1** |
| RChild | **2** | **4** | **5** | **-1** | **-1** | **-1** | **-1** | **9** | **-1** | **-1** |

链表存储结构：



前序遍历：ABDEGCFHIJ

中序遍历：DBGEACIHJF

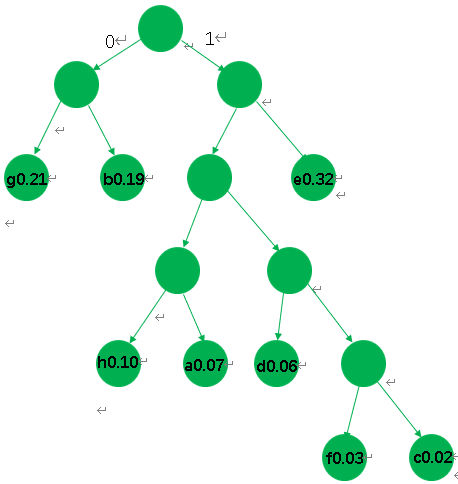
后序遍历：DGEBIJHFCA

12、假设用于通信的电文由字符集{a, b, c, d, e, f, g, h}中的字母构成，这8个字母在电文中出现的概率分别为{ 0.07, 0.19, 0.02, 0.06, 0.32, 0.03, 0.21, 0.10 }。

（1）为这8个字母设计哈夫曼编码。

（2）求出哈夫曼树的带权路径长度WPL。

（3）若用三位二进制数对这8个字母进行等长编码，则哈夫曼编码的平均码长是等长编码的百分之几？



1. 哈夫曼编码 根据上图可得编码表：

a:1001 b:01 c:10111 d:1010 e:11 f:10110 g:00 h:1000

(2) 带权路径长度WPL： WPL=4\*0.07+2\*0.19+5\*0.02+4\*0.06+2\*0.32+5\*0.03+2\*0.21+4\*0.10=2.61

(3)用三位二进行数进行的等长编码平均长度为3，而根据哈夫曼树编码的平均码长为：

2.61/3=0.87=87%

其平均码长是等长码的87%。 所以平均压缩率为13%。

**三、算法设计**

已知二叉树采用二叉树链表存储结构，指向根节点存储地址的指针为t。试编写一算法，判断该二叉树是否为完全二叉树。

算法设计：

先以先序遍历输入一个二叉树，然后构建一个队列，先让根节点入队，队首出队并访问队首的左孩子、右孩子（没有则用’#'代替）。当出队访问的p不存在时，若p后全为NULL，则二叉树为完全二叉树。

|  |  |  |
| --- | --- | --- |
| 功能描述 | 输入 | 输出 |
| 判断该二叉树是  否为完全二叉树 | 二叉链表根结点地址 | int |
| 函数名 | 形参 | 函数类型 |

数据结构设计：

typedef struct BiTNode

{

char data;

struct BiTNode \*lchild,\*rchild;

}BiTNode,\*BiTree;

算法实现：

#include<stdio.h>

#include<stdlib.h>

#include<malloc.h>

#define OK 1

#define FALSE 0

#define TRUE 1

#define ERROR 0

#define OVERFLOW -2

#define MAXQSIZE 100

typedef struct BiTNode

{

char data;

struct BiTNode \*lchild,\*rchild;

}BiTNode,\*BiTree;

BiTree T;

int CreateBiTree(BiTree &T)

{

char ch;

scanf("%c",&ch);

if(ch=='#') T=NULL;

else

{

T = (BiTNode \*)malloc(sizeof(BiTNode));

T->data = ch;

CreateBiTree(T->lchild);

CreateBiTree(T->rchild);

}

return OK;

}

void Judge(BiTree T)

{ int f=0,r=0,s=0;

BiTree p,q[MAXQSIZE];

q[r++]=T;

while(f<r)

{

p=q[f++];

if(p)

{

q[r++]=p->lchild;

q[r++]=p->rchild;

}

else

{

while(f<r)

{

p=q[f++];

if(p)s++;

}

}

}

if(s!=0)printf("二叉树不是完全二叉树\n");

if(s==0)printf("二叉树是完全二叉树\n");

}

int main()

{

printf("请依次输入二叉树的结点：\n");

CreateBiTree(T);

Judge(T);

system("pause");

}