14. 二叉树中每个结点有两个子结点，而对一般的树，则无此限制，所 以，二叉树是树的

特殊情形。(f)

**15 二叉树是一棵结点的度最大为二的树。(f)**

16 赫夫曼树中结点个数一定是奇数。(t)

17 在二叉树的中序遍历序列中，任意一个结点均处在其左孩子结点的后面。(t)

18 假设B是一棵树，B′是对应的二叉树。则B的后根遍历相当于B′的后序遍历 。(f)

19. 通常，二叉树的第i层上有2i-1个结点。(f)

20. 中序线索二叉树的优点是便于在中序下查找直接前驱结点和直接后继结点。(t)

21 二叉树的先序遍历序列中，任意一个结点均处在其孩子结点的前面。(t)

22 由树结点的先根序列和后根序列可以唯一地确定一棵树。 (t)

二叉排序树的平均查找长度为O(logｎ)。(t)

32. 二叉排序树的最大查找长度与（LOG2N）同阶。(f)

字符串是数据对象特定的线性表。(t)

40. 空串与空格串是相同的。(f)

41. 对于一棵m阶的B-树.树中每个结点至多有m 个关键字.除根之外的所有非终端结点至

少有┌m/2┐个关键字。(f)

42. 当二叉排序树是一棵平衡二叉树时，其平均查找长度为O(log2n)。(t)

43. 广义表的表头和表尾都是广义表。(f)

44 二维数组是其数据元素为线性表的线性表。(t)

15 设有二维数组A 5 x 7 ,每一元素用相邻的4个字节存储，存储器按字节编址.

已知A的起始地址为100。则按行存储时，元素A06的第一个字节的地址是（d ）

按列存储时，元素A06的第一个字节的地址是（ a ）

a: 220 b: 200 c: 140 d: 124

16对广义表 A=（（a,(b)）,(c,()),d）执行操作gettail(gethead(gettail(A)))

的结果是：（ b ） 。

a:（） b: （（）） c: d d: (d)

17 假设用于通讯的电文仅由6个字符组成，字母在电文中出现的频率分别为7, 19, 22, 6, 32,

14。 若为这6个字母设计哈夫曼编码（设生成新的二叉树的规则是按给出的次序从左至

右的结合，新生成的二叉树总是插入在最右），则频率为7的字符编码是（ g ），频率

为32的字符编码是（ c ）。

a: 00 b: 01 c: 10 d: 11

e: 011 f: 110 g: 1110 h:1111

18 对二叉排序树（ c ）可得到有序序列。

a:按层遍历 b:前序遍历 c:中序遍历 d:后序遍历

19 设一棵二叉树BT的存储结构如下:

1 2 3 4 5 6 7 8

lchild 2 3 0 0 6 0 0 0

data A B C D E F G H

rchild 0 5 4 0 8 7 0 0

其中lchild，rchild分别为结点的左、右孩子指针域，data为结点的数据域。则

该二叉树的高度为( d )；

第3层有( a )个结点（根结点为第1层）。

A．2 B. 3 C. 4 D. 5

20 先序遍历图示二叉树可得到（ a ）的序列。

(A)

／ ＼

(B) (C)

／ ＼ ＼

(H) (D) (G)

／ ＼

(E) (F)

＼

(I)

a) A B H D E F I C G

b) H B E D F I A C G

c) H E I F D B G C A

21 在有n个结点的二叉树的二叉链表表示中，空指针数 （ b ）。

a.不定 b.n+1 c.n d.n-1

22 若某二叉树有20个叶子结点，有20个结点仅有一个孩子，则该二叉树的总结点数是

( c )。

A．40 B. 55 C. 59 D. 61

23 已知某二叉树的先序遍历次序为abcdefg中序遍历次序为badcgfe，

则该二叉树的后序遍历次序为（ c ）。层次遍历次序为（ a ）。

a: abcdefg b: cdebgfa c: bdgfeca d: edcgfba

.24 图示的三棵二叉树中( c)为最优二叉树。

A) B) C)

c a

2 7

a b c d d b

7 5 2 4 4 5

a b c d

7 5 2 4

25 已知某二叉树的后序遍历和中序遍历次序分别为DBFGECA和BDACFEG。

则其先序遍历次序为（ b ），层次遍历次序为（ a ）。

a: abcdefg b: abdcefg c: abcdfeg d: abcdegf

26 已知某树的先根遍历次序为abcdefg后根遍历次序为cdebgfa。

若将该树转换为二叉树,其后序遍历次序为（ d ）。

a: abcdefg b: cdebgfa c: cdegbfa d: edcgfba

27 设x和y是二叉树中的任意两个结点，若在先根序列中x在y之前，而在后根序列中x

在y之后，则x和y的关系是( c )。  
 A. x是y的左兄弟 B. x是y的右兄弟  
 C. x是y的祖先 D. x是y的子孙

28 用三叉链表作二叉树的存储结构，当二叉树中有n个结点时，有( d )个空指针。

A. n-1 B. n C. n+1 D. n+2

29 对一棵完全二叉树进行层序编号。则编号为n的结点若存在右孩子,其位序是( d )。

编号为n的结点若存在双亲,其位置是( a )。

a: n/2 b: 2n c:2n-1 d:2n+1 e:n f: 2(n+1)

30 设森林F中有三棵树，第一、第二和第三棵树的结点个数分别为m1、m2和m3，则与

森林F对应的二叉树根结点的右子树上的结点个数是( d )。（左边是m1-1）

A. m1 B. m1+m2 C. m3 D. m2+m3

31 下列二叉树中，( a )可用于实现符号不等长高效编码。

a:最优二叉树 b:次优查找树 c:二叉平衡树 d:二叉排序树

32 邻接表存储结构下图的深度优先遍历算法类似于二叉树的(　a )遍历。

A. 先根　　　 B. 中根　　 　C. 后根　　 D. 层次

33 设无向图G = (V,E)和G’= (V’,E’)，若G’是G的生成树，则下面不正确的说法是( b )。

A. G’是G的子图         B. G’是G的连通分量

C. G’是G的无环子图      D. G’是G的极小连通子图且V’= V

34 任何一个连通图的最小生成树( b )。

A．只有一棵 B. 有一棵或多棵 C. 一定有多棵 D. 可能不存在

e f e f

35 深度优先遍历图使用了数据结构（b ），而广度优先遍历图使用了数据结构（ c ）。

A）数组 B）栈 C）队列 D）线性表

36 已知某有向图的邻接表存储结构如图所示。

0 E 2 1 ∧

1 D 0 3 4 ∧

2 C  4

3 B 1 2 0 ∧

4 A 2 ∧

根据存储结构依教材中的算法其深度优先遍历次序为（ d ）。

广度优先遍历此序为（ c ）。各强连通分量的顶点集为（ h ）。

a: abcde. b: edcba. c: ecdab. d: ecadb.

e: abc及ed f: bc及aed g: ab及ced h: ac及bed

37 下列查找方法中（ a ）适用于查找单链表。

A）顺序查找 B）折半查找 C）分块查找 D）hash查找

38 下列算法中（c ）适用于求图的最小代价生成树。（ b ）能对图作广度优先遍历。

A）DFS算法 B）BFS算法 C）Prim算法 D）Dijkstra算法

44 在一个含有n个元素的有序表上进行折半查找，找到一个元素最多要进行( b )次元素

比较。

A．⎣log2(n)⎦ B. ⎣log2(n)⎦+1 C. ⎣log2(n+1)⎦ D. ⎣log2(n+1)⎦+1

45 设输入序列为20, 45, 30, 89, 70, 38, 62，19依次插入到一棵2-3树中(初始状态为空)，

该B-树为（ b ）。再删除38，该B-树为（ f ）。

（ 30 62 ） （ 45 ）

（19，20）（ 38 45 ） （ 70，89 ） （ 30 ） （ 70 ）

（19 20） （38 ）（ 62 ） （ 89 ）

a: b:

（ 45 70 ） （ 45 ）

（20） （ 62 ） （ 89 ） （ 20 ） （ 70 ）

（19）（ 30 ） （ 19 ） ( 30,38 ）（ 62 ） （ 89 ）

c: d:

（ 30 70 ） （ 45 ）

（19，20） （ 45 62） （ 89 ） （ 20 ） （ 70 ）

（19 ） （30 ）（ 62 ） （ 89 ）

e: f:

46根据插入次序（80，90，100，110，85，70，75，60，72）建立二叉排序树。

图（ a ）是最终变化的结果。若仍以该插入次序建立平衡二叉树。图（ c ）是最

终变化的结果。

80 80

70 90 75 90

60 75 85 100 60 70 85 100

72 110 72 110

a: b:

90 90

75 100 80 100

70 80 110 75 70 85 110

60 72 85 60 72

c: d:

47 若有序表中关键字序列为：14，20，25，32，34，45，57，69，77，83，92。对其进行

折半查找，则在等概率情况下，查找成功时的平均查找长度是( c )。查找32时需进

行( c )次比较。

A. 1 B. 2 C. 3 D. 4

48 已知哈希表地址空间为A[9]，哈希函数为H(k)=k mod 7，采用线性探测再散列处理冲突。

若依次将数据序列：76,45,88,21,94,77,17存入该散列表中，则元素17存储的下标为( h )；

在等概率情况下查找成功的平均查找长度为( c )。

A. 0 B. 1 C. 2 D. 3

E. 4 F. 5 G. 6 H. 7

49 若从二叉树的根结点到其它任一结点的路径上所经过的结点序列按其关键字递增有序，

则该二叉树是( c )。

A. 二叉排序树 B. 赫夫曼树 C. 堆 D. 平衡二叉树

50 当待排序序列的关键字次序为倒序时，若需为之进行正序排序，下列方案中( d )为佳。

A. 起泡排序 B. 快速排序

C. 直接插入排序 D. 简单选择排序

25 (算法填空)

Status Preordertraverse(Bitree T,Status(\*Visit)(Telemtype e)) {

//先序非递归遍历二叉树。

Initstack ( S ); Push ( S,T );

While ( !stackempty( S ) )

{ While ( gettop( S, p )&& \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ )

{ if (!Visit (p->data ) ) return ERROR;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ;

}

Pop ( S , p );

if ( \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ )

{ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_; push( S, p->rchild ); }

}

return ok;

}

26 (算法填空)

下列算法试图完成在数组A中搜索有无关键字key，若有，返回数组下标,若无，返回-1。在“ ”处填上合适的内容,完成该算法。

int BinarySearch (keytype A [], int low,int high, keytype key )

{

while ( )

middle = (low+high) /2;

if ( )

return middle;

if (key < A[middle])

;

else

;

}

return -1;

} //end of BinarySearch

27 (算法填空)

下列函数为堆排序中的堆调整过程（调整H.r[s]的关键字，使H.r[s..m]成为一小顶堆）。请在“ ”处填上合适的内容,完成该算法。

Void heapadjust( heaptype @ H , int s , int m ) {

rc=H.r[s];

for (j=2\*s;j<=m;j\*=2) {

if (j<m && ) ++j;

if ( ) break;

H.r[s]=H.r[j]; s=j;

}

;

}//heapadjust

1 已知在电文中只出现频率为 ( 5,26,7,23,20,19 )的６个字符，

画出你建的哈夫曼树，并给出其哈夫曼编码。

2.已知某二叉树的后序遍历和中序遍历次序分别为DBFGECA和BDACFEG

请画出该二叉树，并为之建立先序线索。

3 已知某二叉树的先序遍历次序为：a,b,c,d,e,f,g.中序遍历次序为：b,a,d,f,e,g,c

画出该二叉树，并在该二叉树上建立中序线索。

4 某二叉树的中序遍历次序为BEGFDAC， 先序遍历次序为ABDEFGC。

试画出该二叉树，并为之建立中序线索（图示之）。

5 已知某二叉树的后序遍历和中序遍历次序分别为FBEDGCA和FBADECG，

请构造并画出该二叉树。

6 设某一电文只出现a,b,c,d,e,f,g 7个字母；出现频率分别为30%,10%,05%,04%,13%,18%

及20%，请给出各字母的哈夫曼编码。

7 将图示森林转换为二叉树，并对该二叉树先序全序线索化。

8 将图示森林转换为二叉树，并对该二叉树中序全序线索化。

9 某二叉树的结点数据采用顺序存储表示如下：

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C |  | D |  | E |  |  | F |  |  |  | G | H |  |  |  |  | I |

（1）试画出此二叉树的图形表示。

（2）将此二叉树看作森林的二叉树表示，试将它还原为森林。

10 已知某有向图如图所示：

1）给出其十字链表存储结构

　 2）给出其深度优先遍历次序。

3）给出其广度优先遍历次序。

4）给出各强连通分量。

11 设输入序列为20,45,30,89,70,38,62,19，依次插入到一棵2-3树中(初始状态为空)，请画出该B-树。

1. 右图为一棵3阶B－树。 （20，25）

1)画出在该树上插入元素15 ／ │ ＼

后的B－树。 (10，14)（21）（35）

2)接着，再删除元素35，画出删除后的B－树。

13 已知Hash函数为 H（K）=K mod 13 ，散列地址为0 --14，用线性探测再散列处理

冲突，给出关键字（56，34，68，23，16，70，48，35，83，12，14，57）

在散列地址的分布。并指出平均成功的查找长度是多少?

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

14 根据插入次序（20，30，70，60，10，100，110，90，80。）建立平衡的二叉排序树。

15 设哈希表长为16，哈希函数为H(key)=key mod 13，用开放定址法的二次探测再散列

处理冲突（di=12，-12，22，-22，32，-32……）。依次存入12个元素：56，82，17,24,

36,21,83,96,13,34,57,50。请画出它们在表中的分布情形。

11二叉树用二叉链表存储表示。

typedef struct BiTNode {

TelemType data;

Struct BiTNode \*lchild, \*rchild;

} BiTNode, \*BiTree;

试编写销毁二叉树T的算法DestroyBiTree ( BiTree &T)。

12二叉树用二叉链表存储表示。

typedef struct BiTNode {

TelemType data;

Struct BiTNode \*lchild, \*rchild;

} BiTNode, \*BiTree;

试编写算法，求元素值为x的结点的左孩子（返回x的左孩子的指针）。

13 设计一算法，计算给定二叉树T中度为2的结点个数。

14编一算法：按层序遍历二叉树T。

15试编写先序遍历二叉树T的递归算法PreorderBiTree ( BiTree &T)。

16 写出一个将树中每个结点的左右孩子对换的算法

SWAPTREE(T) 即如:

原二叉树 转换后

T SWAPTREE(T)

↓ ↓

(A) (A)

/ \ / \

(B) (C) (C) (B)

/ \ / \ / \

(D) (E) (F) (F) (E) (D)

\ \ / /

(G) (H) (H) (G)

17 二叉树用二叉链表存储表示。

试编写后序遍历二叉树T的递归算法PostorderBiTree ( BiTree T)。

18 写一个计算二叉树中叶子结点个数的递归算法。