**一、单项选择题**

1．算法必须具备的三个特性是( )。

A．可执行性、可移植性、可扩充性 B．可执行性、确定性、有穷性

C．确定性、有穷性、稳定性 D．易读性、稳定性、安全性

2．下列数据中，( )是非线性数据结构。

A．栈 B．队列

C．完全二叉树 D．顺序表

3．算法分析的两个方面是( )。

A．空间复杂度和时间复杂度 B．正确性和简明性

C．可读性和文档性 D．数据复杂性和程序复杂性

4．非空的循环单链表head的尾结点p满足( )。

A．p->next==head B．p->next==NULL

C．p==NULL D．p==head

5．在单链表指针为p的结点之后插入指针为s的结点，正确的操作是( )。

A．p->next=s;s->next=p->next; B．s->next=p->next;p->next=s;

C．p->next=s;p->next=s->next; D．p->next=s->next;p->next=s;

6．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

7．在一个有向图中，所有顶点的入度之和是所有顶点的出度之和的( )倍。

A．1/2 B．1

C．2 D．4

8．二叉排序树是( )。

A．每一分支结点的度均为2的二叉树

B．中序遍历得到一升序序列的二叉树

C．按从左到右顺序编号的二叉树

D．每一分支结点的值均小于左子树上所有结点的值，大于右子树上所有结点的值

9．若用一个大小为6的数组来实现循环队列，且当前rear和front的值分别为0和3，当从队列中删除一个元素，再加入两个元素后，rear和front的值分别是( )。

A．1和 5 B．2和4

C．4和2 D．5和1

10．下列说法中正确的是( )。

A．堆栈是在两端操作、先进后出的线性表

B．堆栈是在一端操作、先进先出的线性表

C．队列是在一端操作、先进先出的线性表

D．队列是在两端操作、先进先出的线性表

11．不带头结点的单链表head为空的判定条件是( )。

A．head==NULL B．head->next==NULL

C．head->next==head D．head!=NULL

12．设森林F中有三棵树，第一，第二，第三棵树的结点个数分别为M1，M2和M3。与森林F对应的二叉树根结点的右子树上的结点个数是( )。

A．M1 B．M1+M2

C．M3 D．M2+M3

13．具有10个叶结点的二叉树中有( )个度为2的结点。

A．8 B．9

C．10 D．ll

14．有n个叶子的哈夫曼树的结点总数为( )。

A．不确定 B．2n

C．2n+1 D．2n-1

15．利用二叉链表存储树，则根结点的右指针是( )。

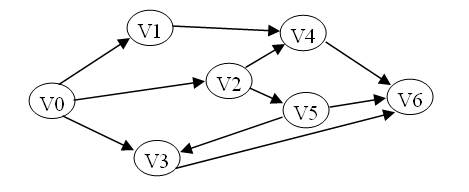
A．指向最左孩子 B．指向最右孩子

C．空 D．非空

16．对下图所示的有向图，从顶点V0出发的深度优先遍历序列是( )。

A．V0 V1 V2 V4 V6 V3 V5 B．V0 V1 V4 V6 V2 V5 V3

C．V0 V2 V5 V6 V1 V3 V4 D．V0 V1 V2 V3 V4 V5 V6



17．要连通具有n个顶点的无向图，至少需要( )条边。

A．n-l B．n

C．n+l D．2n

18．对线性表进行二分查找时，要求线性表必须( )。

A．以顺序方式存储

B．以链式方式存储

C．以顺序方式存储，且结点关键字有序排列。

D．以链式方式存储，且结点关键字有序排列。

19．静态查找表与动态查找表的根本区别在于( )。

A．它们的逻辑结构不同 B．施加在其上的操作不同

C．所包含的数据元素类型不同 D．存储实现不一样

20．一个有n个顶点的有向图最多有( )条边。

A．n B．n(n-1)

C．2n D．n(n-1)/2

**二、判断对错**

【 】1．具有n个顶点的连通图至少有n条边。

【 】2．有实现同一功能的两个算法A1和A2，其中A1的时间复杂度为T1(n)=O(2n)，A2的时间复杂度为T2(n)=O(n2)，仅就时间复杂度而言，A2算法更好。

【 】3．链表的单个结点内部的存储空间可以是不连续的。

【 】4．在完全二叉树中，若一个结点没有左孩子，则它必是叶子结点。

【 】5．假定有4个元素按照A，B，C，D的顺序入栈，入栈过程中允许出栈,则CBDA是可能的出栈序列。

【 】6．顺序存储方式结构只能用于线性结构，不能用于非线性结构。

【 】7．二叉树中不存在度大于2的结点。

【 】8．高度为k的满二叉树是具有2k+1−1个结点的二叉树。

【 】9．对于一棵含有n个结点的树，将其结点按从上到下且从左至右按1至n进行编号，则对其任意一个编号为i的结点，如果它有左孩子，则其左孩子结点的编号为2i。

【 】10．在二叉排序树中，最大值结点和最小值结点一定是叶子结点。

**三、 填空题**

1．给定序列{100，86，48，73，35，39，42，57，66，21}，按堆结构的定义，则它一定是 【1】 堆。

2．从循环队列中删除数据元素时，需要判满队列是否已经为空，判断条件是：

【2】 。

3．在一个带头结点的单链表中，p所指结点既不是首元结点，也不是尾元结点，删除p结点的直接后继结点的语句序列是 【3】 、 【4】 。

4．对关键字序列（52，80，63，44，48，91）进行一趟快速排序之后得到的结果为

【5】 。

5．将含102个结点的完全二叉树从根这一层开始，每层上从左到右依次对结点编号，根结点的编号为1。编号为32的结点X的双亲编号为 【6】 、左孩子节点编号为 【7】 、右孩子节点编号为 【8】 。

6．如果一个图的邻接矩阵是非对称矩阵，则该图必然是 【9】 。

7．如以{ 2,3,6,7,9,12 }作为叶子结点的权值构造哈夫曼树，则其最短带权路径长度为 【10】 。

8．遍历图有 【11】 、 【12】 方法。

9．在单链表中，要删除某一指定的结点，必须找到该结点的 【13】 结点。

10．对于一个具有n个顶点和e条边的无向图，若采用邻接表表示，则表头向量的大小为 【14】 ，邻接表中所有结点的总数是 【15】 。

**四、按照指定功能，完成下列算法**

1．销毁顺序表L

Status DetroyList\_Sq ( SqList &L )

{ 【1】 ;

【2】 ;

【3】 ;

【4】 ;

return OK ;

}

2、在二叉排序树中查找关键字等于key 的元素

Status SearchBST ( BiTree T, KeyType key, BiTree f, BiTree &p)

//在根指针T所指二叉排序树中递归地查找其关键字等于key的数据元素，若查找

//成功， 则指针p指向该数据元素结点，并返回TRUE，否则指针指向查找路径上访

//问的最后一个结点并返回FALSE，指针f指向T的双亲，其初始调用值为NULL

{ if ( !T ) { 【5】 }

else if ( EQ( key, T->data.key ) ) { 【6】 }

else if ( LT( key, T->data.key ) ) 【7】

else 【8】

}

3、出栈操作算法

Status Pop ( SqStack &S, ElemType &e )

{ 【9】 ；

【10】 ；

return OK ;

}

**五、简答题**

1．有一份电文中共使用8个字符：a、b、c、d、e、f、g、h，它们出现的频度分别为：4、7、5、2、9、8、6、3，试设计一个编码使该段电文的总长度为最短。要求：

（1）构造出哈夫曼树（要求所有左子树根结点的权值小于等于右子树根结点的权值）；

（2）求出每个字符的哈夫曼编码；

（3）计算经哈夫曼编码后上述报文的最终长度。

2．对于如下所示的加权无向图，画出用克鲁斯卡尔（Kruskal）算法构造最小生成树的过程中，当生成树有2条、4条、6条边时的生成树。

**9**

**7**

**8**

**2**

**12**

**6**

**4**

**4**

**3**

**1**

**5**

**11**

**10**

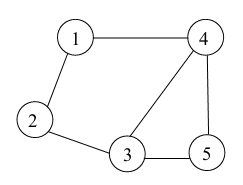
第2题图

3．已知一个有向图如下图所示，请给出该图的：

（1）邻接矩阵；

（2）邻接表；

（3）自顶点1出发进行深度优先遍历和广度优先遍历各一个遍历顶点序列。



第3题图

4．设结点关键码的输入序列为{ 53，68，55，17，82，10，45 }，试利用二叉排序树的插入算法，分别画出插入过程中二叉排序树有3个、5个、7个节点时的二叉排序树。

5．若一棵二叉树先序遍历和中序遍历序列分别为：先序序列为 ABDEHCFGI，中序序列为DBEHAFCIG，试画出这棵二叉树，并给出该二叉树的后续遍历序列。

**六、给出下列程序结果**

#include < stdio.h >

int c = 0 ;

void move( char x, int n, char z )

{ printf( "第%i步: 将%i号盘从%c移到%c\n", ++c, n, x, z ) ;

}

void hanoi( int n, char x, char y, char z )

{ if( n==1 )

move( x, 1, z ) ;

else

{ hanoi( n-1, x, z, y ) ;

move( x, n, z ) ;

hanoi( n-1, y, x, z ) ;

}

}

void main( )

{ int n = 3 ;

hanoi( n, 'a', 'b', 'c' ) ;

}

**七、算法设计**

假设以顺序存储结构实现一个双向栈，即在一维数组的存储空间中存在两个栈，它们的栈底分别设在数组的两个端点。试编写实现这个双向栈tws的三个操作：初始化inistack(tws)、入栈 push(tws,i,x) 和出栈 pop(tws,i) 的算法，其中i取值为0或1，分别指示设在数组两端的两个栈，即通过i的取值区分是对两个栈中的哪一个栈进行入栈和出栈操作。