广东工业大学
2005 年攻读硕士学位研究生入学考试试题

考试科目(代码)名称：数据结构
使用专业：计算机应用技术、计算机软件与理论、计算机系统结构

(考生注意：答卷封面需填写自己的准考证编号，答完后连同本试题一并交回!)

一．填空题(共 24 分)

1．数据结构是_____的数据元素的集合。

2．下列程序段的时间复杂度为_____。

for(s=O，i=n；i>0;　i--)

foz　(j=i+1；j<n;　j++)　s += j;

3．在长度为 n 的双向链表中，求前趋结点的操作是 **O**(_____)的．

4．假设以 U 和 O 分别表示进栈和退栈操作，对输入序列 a，b，c，d，e 进行的栈操作序列为 UUOUOUU000，则得到的输出序列为_____.

5．不含任何字符的串称为_____

6．设广义表 L：((((a)，b))，(((),　c)))则 L 的深度是_____,，L 的长度是_____ ，tail(head(tail{L)))的结果是_____.

7．某二叉树结点的中序序列为 A，B，c，D，E，F，G，后序序列为 B，D，c，A，F，G，E，则先序序列为_____

8．树 T 有 n 个结点且结点的度均为 k 或者 0，则树中的叶子结点总数为_____

9.在有向图的邻接矩阵中,第 i 列的非零元个数就是第 i 顶点的_____

10．如果待排序序列已接近正序或逆序，则在堆排序和快速排序之中，选用_____较为适当.

1 1．对长度为 n 的顺序表作折半查找，最多比较关键字_____次.

12.如果内存工作区的容量为 w,则在对一个长度为 n 的有序初始文件进行置换选择排序时,将产生_____　个初始归并段。

二．单项选择题(共 24 分)

13．数据逻辑结构可以分为

[A]动态结构和静态结构　　[B]线性结构和非线性结构

[C]内部结构和外部结构　　[D]紧凑结构和非紧凑结构

14．在长度为 n 的顺序表的第 i 个位置插入一个新元素的算法的平均时闻复杂度为

[A] O(1)　　[B] O(n)　　[c] O(Iog2　n)　　[D] O(i)

15．与顺序栈相比，链栈的主要优点是

[A]入栈操作更简便　[B]出栈操作更简便　[c]不出现下溢　[D]不出现上溢

16．如果链串结点中的指针占 4 个字节，每个字符占 1 个字节，则结点大小为 6 的链串的存储密度为

[A] 0.2　　[B]　0.4　　[C]　0.6　　[D]0.8

17．二维数组 A[8][9]采用列优先的存储方法，若每个元素各占 2 个存储单元，而且 A[0][0]的地址为 1000，则 A[5][7]的地址为

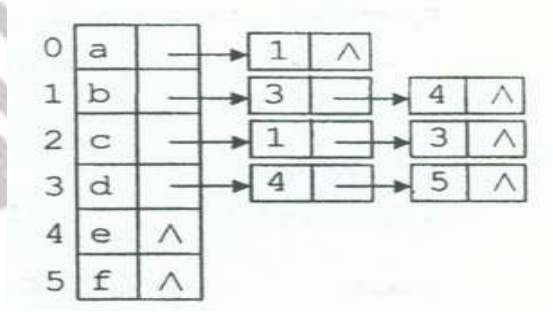
[A]　1122　　[B]　1234　　[C]　1212　　[D]　1120

18．若一棵二叉树有 11 个叶子结点，则该二叉树中度为 2 的结点个数是

[A]　10　　[B]　11　　[C]　12　　[D]　不确定的

1 9．具有 n 个顶点的有向图的弧数最多是
[A]　n　　[B]　n2　　[C]　n(n 一 1)　　[D] n(n+1)
20．对于有向图，其邻接矩阵表示比邻接表表示更易于
[A] 求一个顶点的度　　[B] 求一个顶点的邻接点
[C] 进行图的深度优先遍历　　[D] 进行图的广度优先遍历
21．要得到二叉排序树的结点的排序序列，应对该树进行
[A] 层次遍历　　[B] 先序遍历　　[C] 中序遍历　　[D]后 序遍历
22．下面关于 B 树和 B+树的叙述中，不正确的是
[A]B 树和 B+树都是平衡多叉树
[B]B 树和 B+树都能有效地支持顺序检索
[C]B 树和 B’树都能有效地支持随机检索
[D]B 树和 B+树都可用作文件的索弓 l 结构
23．在以下排序方法中，平均时间复杂度为 O(nlogn)且空间性能最佳的是
[A] 快速排序　　[B] 基数排序　　[C] 归并排序　　[D] 堆排序
24．散列文件的基本存储单位是
[A]物理记录　　[B]逻辑记录　　[C]页块　　[D] 桶

三．解答题(共 45 分)
25．(1 2 分)在对一个有向无环图进行深度优先遍历过程中，如果在 DFS 算法返回时输出当前顶点，则可得到该图的一个逆拓扑序列。给定图 G 的邻接矩阵如右图所示 。



(1)写出基于上述 DFS 算法对图 G 进行深度优先遍历得到的逆拓扑序列；
(2)画出图 G 的逆邻接表；
(3)写出各顶点的入度和出度。

26．(10 分)已知某系统在通信中只使用八种字符，其频率分别为
A(0.04)，B(0.30)，C(0.08)，D(0.09)，E(0.13)，F(0.22)，G(0.03)，H(0.11)
请为其构造哈夫曼树(要求树中所有结点的左右孩子权值必须是左大右小)，并求出各字符的哈夫曼编码．

27．(7 分)设散列函数为 H(k)=k*(k-1)%11，用二次探测法处理冲突。画出依次插入元素 7，4，5，3，6，2，8，1 1 后该哈希表的状态．

28．(8 分)给定关键字序列(20，11，12，9，28，42，7，23，26)，从空树开始依次插入，构造一棵平衡二叉排序树。分别画出构造过程中插入 12、42、7 和 26 后树的平衡状态．

29．(8 分)对序列(37，11，63，22，50，96，55，31)执行升序的堆排序算法，以二叉树的形式画出初始堆和排序过程中各趟的结果．

四．算法题(共 57 分)

30．(7 分)阅读下列排序算法，并回答问题

(1)设队列 P=(2，4，5，7，8)。请写出执行算法 f30(P)后的队列 P；

(2)简述算法 f30 对任意队列 Q 的操作意义．

```
void  f30(Queue  &Q){
Stack  S;  InitStack(S);
while(!QueueEmpty(Q))
Push(S，DeQueue(Q))；
while(!StackEmpty(S))
EnQueue(Q，Pop(S))；
}
```

31. (7 分) 已知有向图的邻接矩阵表和算法 f31 描述如下：

```
typedef  struct  {
char   vexs [MaxNum] ;                    // 字符类型的顶点表
int     arcs [MaxNum] [MaxNum];      // 0 表示无边，1 表示有弧
int      n,   e;                                 // 图中当前的顶点数和弧数
}  MGraph;            // 图的邻接矩阵表示结构类型
void f31(MGraph &G,    int  i)    {
d=0;
for    (j=0;    j<G.n;    j++)    {
if (G.arcs[i][j])       {   d++; G.arcs[i][j] = 0;    }
if (G.arcs[j][i])       {   d++; G.arcs[j][i] = 0;    }
}
G.e -= d;
}
```

阅读算法 f31，并回答问题

(1) 有向图 G 的邻接矩阵 G.arcs 如右图所示，请写出执行算法 f31（G，3）后图 G 的邻接矩阵；

$$G.arcs = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(2) 简述该算法 f31 的功能。

32. (7 分) 阅读排序算法 f32，并回答问题

(1) 算法 f32 属哪一种排序算法?

(2) 简述变量 lc 在算法 f32 中的作用。

```
void f32(SqList &L) {
lc=L.length;
for (i=L. length; i>l && lc!=0;  i —) {
change=0;
for (j=l; j<lc; j++)
if (L.r[j].key > L.r[j+1].key) {
x=L.r[j]; L.r[j]=L.r[j+1]; L.r[j+l]=x; change=j; }
lc=change;
}
}
```

33. (7 分)二叉树的存储结构的类型定义如下:

```
typedef struct BiTNode {
char      data;
BiTNode *lchild, *rchild;
} BiTNode, *BiTree;
```

算法 f33 对二叉树 T 进行非递归先序遍历，请在空缺处填入合适内容，使其成为完整的算法。

```
void f33(BiTree T, void (*visit) (TElemType)) {
Stack S;
InitStack(S);
Push(S,NULL);
__________①__________ ;
while (p!=NULL) {
_____②_____ ;
if (p->rchild!=NULL) _____③_____ ;
if (p->lchild!=NULL) _____④_____ ;
else Pop(S,p);
}
}
```

34. (7 分) 算法 f34 对带头结点单链表 L 进行简单选择排序。请在空缺处填入合适内容，

使其成为完整的算法.

```
void f34(LinkList L) {
_____①_____;
while (p!=NULL) {
q=p->next; r=p;
while (q!=NULL) {
if ( _____②_____ ) r=q;
q=q->next;
}
if ( _____③_____ ) {
x=p->data;
p->data=r->data;
r->data=x;
}
_____④_____ ;
}
}
```

35. (7 分) 已知图的邻接表表示的类型定义如下:

```
typedef struct EdgeNode {
int              adjvex;     // 邻接点域
struct EdgeNode *nextEdge;  //边表链指针域
} EdgeNode;                  // 边结点类型
typedef struct {
char     vertex;            //顶点域
EdgeNode *firstEdge;      // 边表头指针
} VertexNode;                          // 顶点表结点类型
typedef struct {
VertexNode   vertices [MaxNum];   //顶点表
int  n, e;                    // 图中当前的顶点数
```

```
} ALGraph;                                    // 邻接表类型
Status visited[MaxNum];
```

算法 f35 输出图 G 的深度优先生成树（或森林）的边。请在下列算法的空缺处填入合适内容，使其成为完整算法。

```
void DFSTree(ALGraph G, int i) {
visited[i] = TRUE;
p = G.adjlist[i].firstEdge;
while ( ________①________ ) {
if (!visited[p->adjvex]) {
printf("<%c, %c>\n", G.vertices[i], G.vertices[j]);
____________②_____________
}
____________③_____________
}
}
void f35(ALGraph G) {
for (i=0; i<G.n; i++)
visited[i] = FALSE;
for (i=0; i<G.n; i++)
if ( ________④________ ) DFSTree (G, i) ;
}
```

(15 分) 顺序表类型 SqList 定义如下:

```
typedef int KeyType;
typedef struct {
KeyType key;
InfoType otherlnfo;
} RedType;
typedef struct {
RedType r [MAXSIZE+1]; // r[0]保留未用，元素从 r [1] 开始存放 int
length;
} SqList;
```

编写堆判定算法,如果顺序表 L 不是堆,则返回 0;如果是堆,则返回 1.

在算法中应有必要的注释,并在算法首部以注释的形式简述算法的基本思想、参数的意思和算法时间复杂度