

安徽工业大学 2008 年招收攻读硕士学位研究生专业试卷(A)

科目名称: 数据结构 代码: 861

(注: 总分: 150 考试时间: 180 分钟)

一、单项选择题 (30×1.5=45 分)

1. 程序段 FOR(i=n-1;i>=0;i--)

FOR(j=1;j<=n;j++)

IF A[j]>A[j+1]

A[j]与A[j+1]对换;

其中 n 为正整数, 则最后一行的语句频度在最坏情况下是_____。

A. $O(n)$ B. $O(n\log_2 n)$ C. $O(n^3)$ D. $O(n^2)$

2. 用链表表示线性表的优点是_____。

A. 便于随机存取 B. 花费的存储空间较顺序存储少 C. 便于插入和删除

D. 数据元素的物理顺序与逻辑顺序相同

3. 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

4. 在循环双链表的 p 所指结点之后插入 s 所指结点的操作是_____。

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

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

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

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

5. 栈应用在_____。

A. 递归调用 B. 子程序调用 C. 表达式求值 D. A, B, C 都对

6. 设 abcdef (a 先进栈) 顺序进栈, 若在进栈操作时, 允许出栈操作, 则下面得不到的序列为_____。

A. fedcba B. bcafed C. dcefb a D. cabdef 注: 序列 xyz 表示 x 先出栈; z 最后出栈。

7. 若一个栈的输入序列为 1, 2, 3, 4, 5 则输出序列有_____种可能。

A. 14 B. 120 C. 60 D. 42

8. 循环队列存储在数组 A[0..m] 中, 则入队时队尾的操作为_____。

A. rear=rear+1 B. rear=(rear+1)%(m-1)

C. rear=(rear+1)%m D. rear=(rear+1)%(m+1)

9. 在简单模式匹配中, 当模式串位 j 与主串位 i 的比较时, 新一趟匹配开始, 主串的位移公式是_____。

A. i=i+1 B. i=j+1 C. i=i-j+1 D. i=i-j+2

10. 稀疏矩阵一般的压缩方法是_____。

A. 二维数组和三维数组 B. 三元组和散列表 C. 三元组和十字链表 D.

散列和十字链表

11. 若对 n 阶对称矩阵 A 以行序为主序方式将其下三角形的元素(包括主对角线上所有元素)依次存放于一维数组 $B[1..(n(n+1))/2]$ 中, $a[0][0]$ 存放于数组 $B[1]$ 中, 则在 B 中确定 $a_{ij}(i < j)$ 的位置 k 的关系为_____。
- A. $i*(i+1)/2+j$ B. $j*(j+1)/2+i$ C. $i*(i+1)/2+j+1$
D. $j*(j+1)/2+i+1$
12. 设广义表 $L=((a, b, c))$, 则 L 的长度和深度分别为_____。
- A. 1 和 1 B. 1 和 3 C. 1 和 2 D. 2 和 3
13. 有一个 $100*90$ 的稀疏矩阵, 非 0 元素有 10 个, 设每个整型数占 2 字节, 则用三元组表示该矩阵时, 所需的字节数是_____。
- A. 60 B. 66 C. 18000 D. 33
14. 已知广义表 $LS=((a, b, c), (d, e, f))$, 运用 $GetHead$ 和 $GetTail$ 函数取出 LS 中原子 e 的运算是_____。
- A. $GetHead (GetTail (LS))$
B. $GetHead (GetTail (GetHead (GetTail (LS))))$
C. $GetTail (GetHead (LS))$
D. $GetHead (GetTail (GetTail (GetHead (LS))))$
15. 一棵三叉树中, 已知度为 3 的结点数等于度为 2 的结点数, 且树中叶子数为 7, 则度为 2 的结点数目为_____。
- A. 4 B. 2 C. 3 D. 5
16. 下面关于二叉树的结论正确的是_____。
- A. 二叉树中, 度为 0 的结点个数等于度为 2 的结点个数加 1。
B. 二叉树中结点个数必大于 0。
C. 完全二叉树中, 任何一个结点的度, 或者为 0, 或者为 2。
D. 二叉树的度是 2
17. 设 X 是树 T 中的一个非根结点, B 是 T 所对应得二叉树, 在 B 中, X 是其双亲的右孩子, 下列结论正确的是_____。
- A. 在树 T 中, X 是其双亲的第一个孩子。 B. 在树 T 中, X 一定无右边兄弟。
C. 在树 T 中, X 一定是叶子结点。 D. 在树 T 中, X 一定有左边兄弟
18. 一棵有 n 个结点的 k 叉树, 树中所有结点的度之和为_____。
- A. $n-1$ B. kn C. n^2 D. $2n$
19. 图的广度优先搜索类似于树的_____次序遍历。
- A. 先根 B. 中根 C. 后根 D. 层次
20. 欲实现任意二叉树的后序遍历的非递归算法而不必使用栈结构, 最佳方案是二叉树采用_____存储结构。
- A. 三叉链表 B. 广义表 C. 二叉链表 D. 顺序
21. 一棵二叉树满足下列条件: 对任一结点, 若存在左、右子树, 则其值都小于它的左子树上所有结点的值, 而大于右子树上所有结点的值。现采用_____遍历方式就可以得到这棵二叉树上所有结点的递减序列。

- A. 先序 B. 中序 C. 后序 D. 层次
22. 设给定权值的叶子总数有 n 个, 其哈夫曼树的结点总数为_____。
- A. 不确定 B. $2n$ C. $2n+1$ D. $2n-1$
23. 某二叉树的先序遍历序列和后序遍历序列正好相反, 则此二叉树一定是_____。
- A. 空或只有一个结点 B. 完全二叉树 C. 单支树 D. 高度等于结点数
24. 对_____进行相应的遍历仍需要栈的支持。
- A. 先序线索树 B. 中序线索树 C. 后序线索树 D. A 与 B
25. 具有 7 个顶点的有向图至少应有_____条边才能确保一个强连通图。
- A. 6 B. 7 C. 8 D. 9 B
26. 采用邻接表存储图的深度优先遍历算法类似于树的_____。
- A. 中根遍历 B. 先根遍历 C. 后根遍历 D. 层次遍历
27. 判断有向图是否存在回路, 除了可以利用拓扑排序方法外, 还可以利用_____。
- A. 求关键路径的方法 B. 求最短路径的 Dijkstra 算法
- C. 深度优先遍历算法 D. 广度优先遍历算法
28. 二叉排序树的查找效率与二叉排序树的_____有关, 当 _____时, 查找效率最低, 其查找长度为 n 。
- A. 高度; 结点太多 B. 结点的个数; 完全二叉树
- C. 形状; 呈单叉树 D. 结点的位置; 结点的结构太复杂
29. 假定有 k 个关键字互为同义词, 若用线性探测法把这 k 个关键字存入哈希表中, 至少要进行多少次探测_____。
- A. $k-1$ 次 B. k 次 C. $k+1$ 次 D. $k(k+1)/2$ 次
30. 4. 若需在 $O(n\log_2 n)$ 的时间内完成对数组的排序, 且要求排序是稳定的, 则可选的排序方法是_____。
- A. 快速排序 B. 堆排序 C. 直接插入排序 D. 归并排序

二、填空 (10×2=20 分)

1. 数据结构的存储结构基本上有顺序、_____、索引和散列等四种。
2. 在链表中进行_____操作的效率比在顺序存储结构中进行相同操作的效率高。
3. 广义表的深度定义为广义表中括号被嵌套的_____。
4. 设一棵完全二叉树叶子结点数为 k , 最后一层结点数 > 2 , 则该二叉树的高度为_____。
5. 一棵树按照左孩子一右兄弟表示法转换成对应的二叉树, 则该二叉树中树根结点肯定没有_____。
6. 设图 $G=(V, E)$, $V=\{1, 2, 3, 4\}$, $E=\{\langle 1, 2 \rangle, \langle 1, 3 \rangle, \langle 2, 4 \rangle, \langle 3, 4 \rangle\}$, 从顶点 1 出发, 对图 G 进行广度优先搜索的序列有_____种。

7. 在哈希查找中, 装填因子为 α , 若用 m 表示哈希表的长度, n 表示哈希存储的元素个数, 则 α 等于_____。
8. 在顺序表(8, 11, 15, 19, 25, 26, 30, 33, 42, 48, 50)中, 用折半法查找关键字值 20, 需做的关键字比较次数为_____。
9. 快速排序的递归算法在平均情况下的空间复杂度为_____。
10. 设图的顶点数为 n , 则求解最短路径的 Dijkstra 算法的时间复杂度为_____。

三、判断 (10×1=10 分)

1. 数据结构的抽象操作的定义与具体实现有关。()
2. 在链式存储表中存取表中的数据元素时, 不一定要按顺序访问。()
3. 链表中的头结点仅起到标识的作用。()
4. 消除递归不一定需要使用栈。()
5. 循环队列只能通过链式存储结构实现。()
6. 一个广义表的表尾总是一个表。()
7. 若一个叶子结点是某二叉树中序遍历序列的最后一个结点, 那么它也是该二叉树的先序遍历序列的最后一个结点。()
8. 对一个无向连通图进行一次深度优先搜索可以访问图中的所有顶点。()
9. 邻接矩阵适用于稀疏图(边数远小于顶点数的平方), 邻接表适用于稠密图(边数接近于顶点数的平方)。()
10. 对二叉排序树进行先序遍历得到的结点的值的序列是一个有序序列。()

四、应用题 (7×5=35 分)

1. 已知一棵二叉树的中序序列和后序序列分别为 BDCEAFHG 和 DECBHGFA, 画出这棵二叉树。并写出其先序遍历序列。(5 分)

2. 已知广义表:

$B = (b, c)$

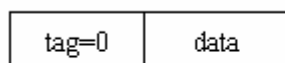
$C = (a, (d, e, f))$

$D = (B, C)$

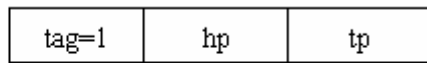
$E = (a, E)$

设广义表原子节点和表节点的存储映像如下图所示 (其中 hp 表示表头指针, tp 表示表尾

指针), 请画出广义表 B、C、D、E 的存储映像图。(5 分)



原子结点

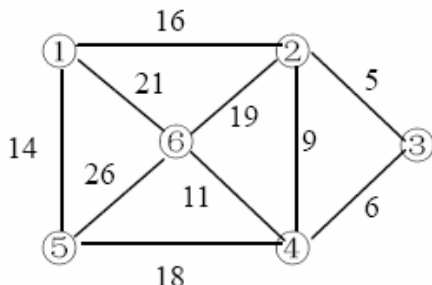


表结点

3. 已知一个文件中有 5 个字符 a、b、c、d、e, 各个字符的出现的次数

依次分别是 3、4、8、10、16，试为这 5 个字符编码，以节省存储空间。
(5 分)

4. 对于下图所示的无向连通图，请用 Prim 算法构造其最小生成树，设算法从图中顶点 1 开始处理。(5 分。注：要求写出求解过程)



5. 给出待排序序列的关键字序列为 {87, 52, 61, 27, 37, 45 }, 请写出对该序列进行堆排序的过程 (注：升序排序，写出每趟排序的过程)。(5 分)

6. 请将下列计算二叉数深度的算法补充完整，每个空格一分。(5 分)

```
int BtreeDepth (BTreeNode *BT)
{
    (1) _____
    if (BT==NULL)
        return (1);
    else
    {
        (3) _____
        int dep1,dep2;
        dep1= (2) _____;
        dep2= (3) _____;
        if (dep1>dep2) return (4) _____;
        return (5) _____;
    }
}
```

7. 已知完全二叉树的第 8 层有 7 片叶子，请指出所有可能的情况下的叶子数目，不需要画出图形，文字说明即可。(5 分)

五、算法设计题(40 分)

1. 已知带头节点的单链表 L，写一算法，删除其中的重复结点(15 分)。
设单链表节点存储结构定义如下：

```
Typedef struct node{
    DataType data; /*每个元素数据信息*/
    struct node *next; /*存放后继元素的地址*/
} Lnode,*LinkList;
```

2. 奇偶交换排序算法如下所述：第一趟对所有下标 i 为奇数的元素，将 $a[i]$ 与 $a[i+1]$ 比较，第二趟对所有下标 i 为偶数的元素，将 $a[i]$ 与 $a[i+1]$ 比较，如 $a[i] > a[i+1]$ 则将二者交换；第三趟对奇数 i ，第四趟对偶数 i ， \dots ，重复上述处理过程直到整个文件有序（10 分）。

(1) 问此种排序算法的结束条件是什么？（3 分）

(2) 用 C 语言实现此算法。（7 分）

设待排序文件采用如下存储结构：

```
#define MAXSIZE 100
typedef struct {
    DataType r[MAXSIZE + 1];
    int length; /*表长度*/
} SqList, *PsqlList;
```

3. 写出对中序线索二叉树进行中序遍历的算法，不允许使用栈。（15 分）

设中序线索二叉树的存储结构定义如下：

```
typedef char DataType;          /*不妨设数据类型为字符型 */
typedef struct Threadnode {
    int ltag, rtag;
    DataType data;
    struct Threadnode *lchild, *rchild;
} Threadnode;
```