

2022 年北京工业大学计算机科学与技术专业《数据结构与算法》科目

期末试卷 A（有答案）

一、选择题

1、用有向无环图描述表达式 $(A+B)*[(A+B)/A]$ ，至少需要顶点的数目为（ ）。

A.5 B.6 C.8 D.9

2、设有一个 10 阶的对称矩阵 A，采用压缩存储方式，以行序为主存储， a_{11} 为第一元素，其存储地址为 1，每个元素占一个地址空间，则 a_{85} 的地址为（ ）。

A.13 B.33 C.18 D.40

3、连续存储设计时，存储单元的地址（ ）。

A.一定连续 B.一定不连续 C.不一定连续 D.部分连续，部分不连续

4、在下列表述中，正确的是（ ）

A.含有一个或多个空格字符的串称为空格串

B.对 n ($n>0$) 个顶点的网，求出权最小的 $n-1$ 条边便可构成其最小生成树

C.选择排序算法是不稳定的

D.平衡二叉树的左右子树的结点数之差的绝对值不超过 1

5、已知串 $S='aaab'$ ，其 next 数组值为（ ）。

A.0123 B.1123 C.1231 D.1211

6、循环队列放在一维数组 A 中，end1 指向队头元素，end2 指向队尾元素的后一个位置。假设队列两端均可进行入队和出队操作，队列中最多能容纳 $M-1$ 个元素。初始时空，下列判断队空和队满的条件中，正确的是（ ）。

A. 队空： $end1 == end2$ ；队满： $end1 == (end2 + 1) \bmod M$

B. 队空： $end1 == end2$ ；队满： $end2 == (end1 + 1) \bmod (M - 1)$

C. 队空： $end2 == (end1 + 1) \bmod M$ ；队满： $end1 == (end2 + 1) \bmod M$

D. 队空： $end1 == (end2 + 1) \bmod M$ ；队满： $end2 == (end1 + 1) \bmod (M - 1)$

7、已知字符串 S 为“abaabaabacacaabaabcc”，模式串 t 为“abaabc”，采用 KMP 算法进行匹配，第一次出现“失配” ($s \neq t$) 时， $i=j=5$ ，则下次开始匹配时， i 和 j 的值分别 ()。

A. $i=1, j=0$ B. $i=5, j=0$ C. $i=5, j=2$ D. $i=6, j=2$

8、已知一棵二叉树的前序遍历结果为 ABCDEF，中序遍历结果为 CBAEDF，则后序遍历结果为 ()。

A.CBEFDAB.FEDCBAC.CBEDFAD. 不定

9、设 X 是树 T 中的一个非根结点，B 是 T 所对应的二叉树。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 一定无右兄弟

C.在树 T 中，X 一定是叶结点

D.在树 T 中，X 一定有左兄弟

10、对关键字序列 28, 16, 32, 12, 60, 2, 5, 72 快速排序，从小到大一次划分结果为 ()。

A. (2, 5, 12, 16) 28 (60, 32, 72)

B. (5, 16, 2, 12) 28 (60, 32, 72)

C. (2, 16, 12, 5) 28 (60, 32, 72)

D. (5, 16, 2, 12) 28 (32, 60, 72)

二、填空题

11、顺序查找 n 个元素的顺序表，若查找成功，则比较关键字的次数最多为____次；当使用监视哨时，若查找失败，则比较关键字的次数为____。

12、对单链表中元素按插入方法排序的 C 语言描述算法如下，其中 L 为链表头结点指针。请填写算法中标出的空白处，完成其功能。

```

typedef struct node
{
    int data; struct node *next;
}linknode,*link;
void Insertsort(link L)
{
    link p,q,r,u;
    p=L->next;____(1)____;
    while(____(2)____)
    {
        r=L; q=L->next;
        while(____(3)____ && q->data<=p->data) {r=q; q=q->next;}
        u=p->next;____(4)____;____(5)____; p=u;
    }
}

```

- 13、设有两个算法在同一机器上运行，其执行时间分别为 $100n^2$ 和 $2n$ ，要使前者快于后者， n 至少为_____。
- 14、外排序的基本操作过程是_____和_____。
- 15、检索是为了在文件中寻找满足一定条件的记录而设置的操作。检索可以按_____检索。也可以按_____检索；按_____检索又可以有_____检索和_____检索。
- 16、设广义表 $L = ((), ())$ ，则 $\text{head}(L)$ 是_____； $\text{tail}(L)$ 是_____； L 的长度是_____；深度是_____。
- 17、设数组 $a[1..50, 1..80]$ 的基地址为 2000，每个元素占 2 个存储单元，若以行序为主序顺序存储，则元素 $a[45, 68]$ 的存储地址为_____；若以列序为主序顺序存储，则元素 $a[45, 68]$ 的存储地址为_____。
- 18、设有一个空栈，栈顶指针为 1000H（十六进制），现有输入序列为 1, 2, 3, 4, 5，经过 PUSH, PUSH, POP, PUSH, POP, PUSH, PUSH 之后，输出序列是_____，而栈顶指针值是_____。设栈为顺序栈，每个元素占 4 个字节。

三、判断题

- 19、对处理大量数据的外存介质而言，索引顺序存取方法是一种方便的文件组织方法。
()
- 20、直接访问文件也能顺序访问，只是一般效率不高。()
- 21、循环队列也存在空间溢出问题。()

- 22、设模式串的长度为 m ，目标串的长度为 n ，当 $n \approx m$ 且处理只匹配一次的模式时，朴素的匹配（即子串定位函数）算法所花的时间代价可能会更为节省。（ ）
- 23、用一维数组存储二叉树时，总是以前序遍历顺序存储结点。（ ）
- 24、深度为 k 的二叉树中结点总数小于等于 $2^k - 1$ 。（ ）
- 25、为提高排序速度，进行外排序时，必须选用最快的内排序算法。（ ）
- 26、在外部排序过程中，对长度为 n 的初始序列进行“置换-选择”排序时，可以得到的最大初始有序段的长度不超过 $n/2$ 。（ ）
- 27、无环有向图才能进行拓扑排序。（ ）
- 28、B-树中所有结点的平衡因子都为零。（ ）

四、简答题

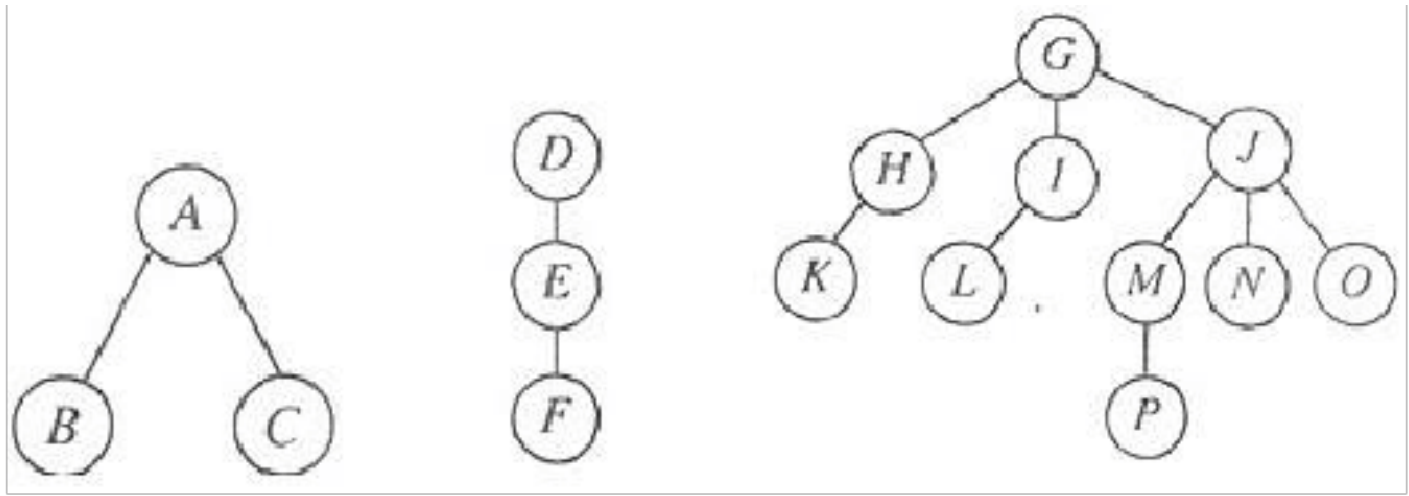
29、调用下列 C 函数 $f(n)$ ，回答下列问题：

- (1) 试指出 $f(n)$ 值的大小，并写出， $f(n)$ 值的推导过程。
- (2) 假定 $n=5$ ，试指出， $f(5)$ 值的大小和执行， $f(5)$ 时的输出结果。

C 函数：

```
int f(int n)
{
    int i,j,k,sum=0;
    for(i=1;i<n+1;i++)
    {for(j=n;j>i-1;j--)
        for(k=1;k<j+1;k++)
            sum++;
        printf("sum=%d\n",sum);
    }
    return(sum);
}
```

30、将下列由三棵树组成的森林转换为二叉树（只要求给出转换结果）。



31、二叉树的带权路径长度（WPL）是二叉树中所有叶结点的带权路径长度之和，给定一棵二叉树 T，采用二叉链表存储，节点结构为：

left	weight	right
------	--------	-------

其中叶节点的 weight 域保存该结点的非负权值。设 root 为指向 T 的根节点的指针，设计求 T 的 WPL 的算法。要求：

- (1) 给出算法的基本设计思想。
- (2) 使用 C 或 C++ 语言，给出二叉树结点的数据类型定义。
- (3) 根据设计思想，采用 C 或 C++ 语言描述算法，关键之处给出注释。

五、算法设计题

32、已知二叉树的结点形式为 (llink, data, count, rlink)，在树中查找值为 X 的结点，若找到，则记数 (count) 加 1；否则，作为一个新结点插入树中，插入后仍为二叉排序树，写出其非递归算法。

33、假定用两个一维数组 L[N]和 R[N]作为有 N 个结点 1, 2, ..., N 的二叉树的存储结构。L[i]和 R[i]分别指示结点 i 的左儿子和右儿子，L[i]=0 (R[i]=0) 表示 i 的左 (右) 儿

子为空。试写一个算法，由 L 和 R 建立一个一维数组 T[n]，使 T[i]存放结点 i 的父亲；然后再写一个判别结点 u 是否为结点 V 的后代的算法。

34、用邻接多重表存储结构，编写 FIRST-ADJ (G, V)函数，函数返回值为第一个邻接点，若 V 没有邻接点，返回零。

35、设有两个栈 S_1 ， S_2 都采用顺序栈方式，并且共享一个存储区 [0..maxsize-1]，为了尽量利用空间，减少溢出的可能，可采用栈顶相向，迎面增长的存储方式。试设计 S_1 ， S_2 有关入栈和出栈的操作算法。

参考答案

一、选择题

1、【答案】A

2、【答案】B

3、【答案】A

4、【答案】C

5、【答案】A

6、【答案】A

7、【答案】C

8、【答案】A

9、【答案】D

10、【答案】B

二、填空题

11、【答案】 n ; $n+1$

12、【答案】

(1) $L \rightarrow next = NULL$ //置空链表，然后将原链表结点逐个插入到有序表中

(1) $p \neq NULL$ //当链表尚未到尾， p 为工作指针

(2) $q \neq NULL$ //查 P 结点在链表中的插入位置，这时 q 是工作指针

(4) $p \rightarrow next = r \rightarrow next$ //将 P 结点链入链表中

(5) $r \rightarrow next = p$ // r 是 q 的前驱， u 是下个待插入结点的指针

13、【答案】15

14、【答案】生成有序归并段（顺串）；归并

15、【答案】关键字；记录号；记录号；顺序；直接

16、【答案】（ ）；（（ ））；2；2

- 17、【答案】 9174 ; 8788
- 18、【答案】 23 ; 100CH

三、判断题

- 19、【答案】 ×
- 20、【答案】 ×
- 21、【答案】 √
- 22、【答案】 √
- 23、【答案】 ×
- 24、【答案】 √
- 25、【答案】 ×
- 26、【答案】 ×
- 27、【答案】 √
- 28、【答案】 √

四、简答题

29、答：（1）第一层 for 循环判断 n+1 次，往下执行 n 次，第二层 for 执行次数为（n+（n-1）+（n-2）+...+1），第三层循环体受第一层循环和第二层循环的控制，其执行次数如表 1-1 所示。

i=	1	2	3	...	n
j=n	n	n	n	...	n
j=n-1	n-1	n-1	n-1	...	
...	...	...			
j=3	3	3	3		
j=2	2	2			
j=1	1				

执行次数为 $f(n) = (1+2+...+n) + (2+3+...+n) + ... + n = n * n (n+1) / 2 - n (n-1) / 6$ 。

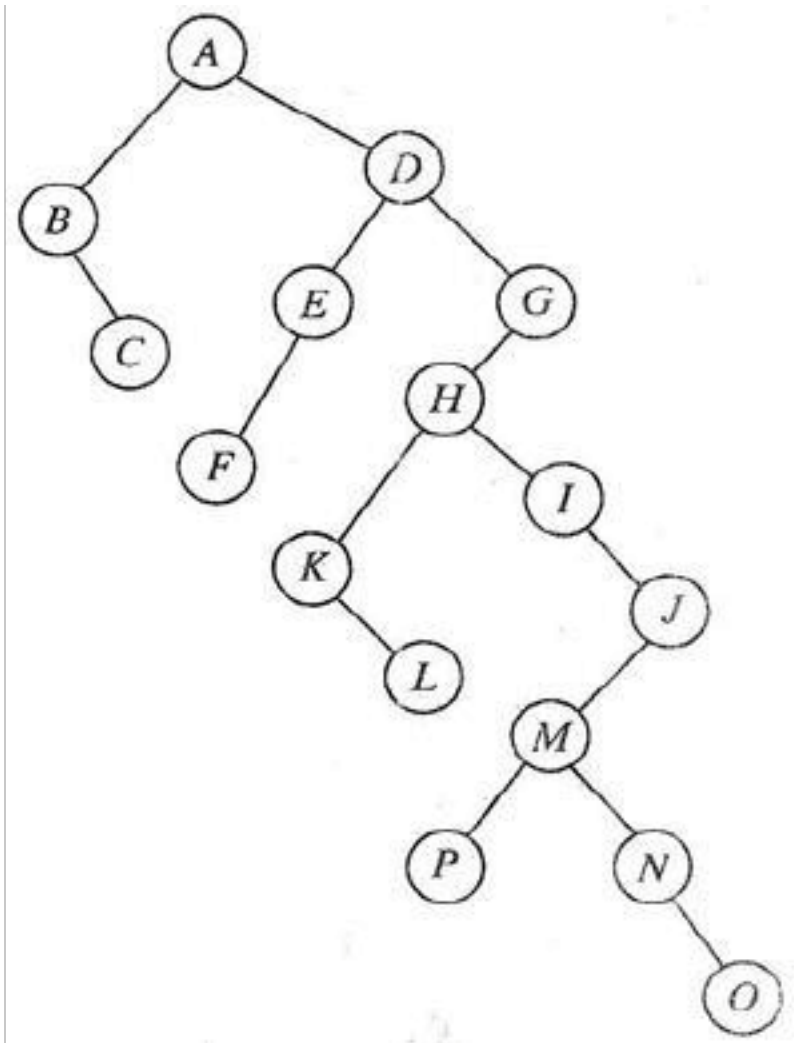
（2）在 n=5 对，f（5）=55，执行过程中，输出结果为：

sum=15
sum=29
sum=41
sum=50
sum=55

30、答：森林转换为二叉树分以下三步：

- (1) 连线（将兄弟结点相连，各树的根看作兄弟）。
- (2) 切线（保留最左边子女为独生子女，将其他子女分支切掉）。
- (3) 旋转（以最左边树的根为轴，顺时针向下旋转 45 度）。

所以由上面三棵树转换得到的二叉树如图所示：



31、答：（1）算法的基本思路是利用递归的思想来求解二叉树的带权路径长度，如果当前节点不是叶子节点，那么当前节点为根的树的带权路径长度便等于它的子树的带权路径长度之和，对于此函数要传入一个当前节点的树高的形参，那么递归调用孩子节点时只需要将这个形参加一即可。

(2)

```
typedef struct BiNode
{
    int weight; //权值
    struct BiNode *left; //左孩子指针
    struct BiNode *right; //右孩子指针
} BiNode, *BiTree;
```

(3) 具体算法实现如下：

```
int CalcWPL(BiTree BT, int height)
{
    if(BT == NULL) //如果当前节点为空节点
        return 0;
    //如果当前节点的左右孩子节点都为空，即当前节点为叶子节点，直接返回当前节点的 WPL
    if(!BT->left && !BT->right)
        return BT->weight * height;
    //如果当前节点不是叶子节点，则对当前节点的左右子树进行递归，返回左右子树 WPL 之和
    else
        return CalcWPL(BT->left, height+1) + CalcWPL(BT->right, height+1);
}
```

五、算法设计题

32、答：算法如下：

```

typedef struct node
{ElemType data;int count;struct node *llink,*rlink;
}BiTNode,*BSTree;
void Search_InsertX(BSTree t,ElemType X)
//在二叉排序树t中查找值为X的结点,若查到,则其结点的count域值增1,否则,将其
//插入到二叉排序树中
{p=t;
while(p!=null && p->data!=X) //查找值为X的结点,f指向当前结点的双亲
{f=p;
if(p->data<X) p=p->rlink; else p=p->llink;
}
if(!p) // 无值为x的结点,插入之
{p=(BiTNode *)malloc(sizeof (BiTNode));
p->data=X;p->llink=null;p->rlink=null;
if(f->data>X)f->llink=p;else f->rlink=p;
}
else p->count++;// 查询成功,值域为X的结点的count增1
} // Search_InsertX

```

33、答：算法如下：

```

int Generation (int U,V,N,L[],R[],T[])
//L[]和R[]是含有N个元素且指示二叉树结点i左儿子和右儿子的一维数组
//本算法据此建立结点i的双亲数组T,并判断结点U是否是结点V的后代
{for(i=1;i<=N;i++) T[i]=0; //T数组初始化
for(i=1;i<=N;i++) if(L[i]!=0) T[L[i]]=i; //若结点i的左子女是L,则结点L的双亲是结点i
for(i=1;i<=N;i++) if(R[i]!=0) T[R[i]]=i; //若结点i的右子女是R,则R的双亲是i
int parent=U; //判断U是否是V的后代
while(parent!=V && parent!=0) parent=T[parent];
if(parent==V) {printf("结点U是结点V的后代"); return(1);}
else{ printf("结点U不是结点V 的后代");return(0);}
}结束Generation

```

34、答：算法如下：

```

int FirstAdj(AdjMuList g , vertype v)
//在邻接多重表g中,求v的第一邻接点,若存在,返回第一邻接点,否则返回0
{i=GraphLocateVertex(g,v); //确定顶点v在邻接多重表向量中的下标,不考虑不存在v的情况
p=g[i].firstedge;           //取第一个边结点
if(p==null ) return (0);
    else {ifp->(ivex==i) return (p->jvex); else return (p->ivex);}
        //返回第一邻接点,ivex和jvex中必有一个等于i
} // FirstAdj

```

35、答：栈的定义：

```

#define maxsize    两栈共享顺序存储空间所能达到的最多元素数
#define ElemType int    //假设元素类型为整型
typedef struct
{ElemType stack[maxsize]; //栈空间
  int top[2];              //top为两个栈顶指针
}stk;
stk s;                    //s是如上定义的结构类型变量,为全局变量

```

(1)入栈操作：

```

int push(int i,int x)
//入栈操作。i为栈号,i=0表示左栈s1,i=1表示右栈s2,x是入栈元素。入栈成功返回1,否则返回0
{if(i<0||i>1){printf("栈号输入不对\n");exit(0);}
  if(s.top[1]-s.top[0]==1) {printf("栈已满\n");return(0);}
switch(i)
  {case 0: s.stack[++s.top[0]]=x; return(1); break;
   case 1: s.stack[--s.top[1]]=x; return(1);
   }
} //push

```

(2)出栈操作

```
ElemType pop(int i)
//出栈算法。i代表栈号,i=0时为s1栈,i=1时为s2栈。出栈成功返回出栈元素,否则返回-1
{if(i<0 || i>1){printf("栈号输入错误\n");exit(0);}
  switch(i)
  {case 0: if(s.top[0]==-1) {printf("栈空\n");return(-1);}
    else return(s.stack[s.top[0]--]);
  case 1: if(s.top[1]==maxsize {printf("栈空\n"); return(-1);}
    else return(s.stack[s.top[1]++]);
  }
} //switch      } //算法结束
```