


# 启航考研 数据结构 习题



---

# 目录

第 1 章 绪论.....	3
第 2 章 线性表.....	3
第 3 章 栈、队列和数组.....	8
第 4 章 树与二叉树.....	10
第 5 章 图.....	12
第 6 章 查找.....	15
第 7 章 排序.....	16



启航龙图<sup>TM</sup>  
SAILING EDUCATION GROUP

## 第1章 绪论

1、设  $n$  是描述问题规模的非负整数，下面程序片段的时间复杂度是 ( )

```
x=2;
```

```
While(x<n/2)
```

```
    x=2*x;
```

A.  $O(\log_2 n)$

B.  $O(n)$

C.  $O(n \log_2 n)$

D.  $O(n^2)$

参考答案：A

2、求整数  $n(n \geq 0)$  阶乘的算法如下，其时间复杂度是 ( )

```
int fact(int n)
```

```
{
```

```
if (n<=1)
```

```
    return 1;
```

```
return n*fact(n-1);
```

```
}
```

A.  $O(\log n)$

B.  $O(n)$

C.  $(n \log n)$

D.  $O(n^2)$

参考答案：B

## 第2章 线性表

1、已知一个带有表头结点的单链表，结点结构为

data	link
------	------

假设该链表只给出了头指针 **list**。在不改变链表的前提下，请设计一个尽可能高效的算法，查找链表中倒数第  $k$  个位置上的结点 ( $k$  为正整数)。若查找成功，算法输出该结点的 **data** 值，并返回 1；否则，只返回 0。要求：

(1) 描述算法的基本设计思想；

(2) 描述算法的详细实现步骤；

(3) 根据设计思想和实现步骤，采用程序设计语言描述算法 (使用 C 或 C++ 或 JAVA 语言实现)，关键之处请给出简要注释。

参考答案：

(1) 算法基本思想如下：

从头至尾遍历单链表，并用指针 **P** 指向当前节点的前  $K$  个节点。当遍历到链表的最后一个节点时，指针 **P** 所指向的节点即为所查找的节点。

(2) 详细实现步骤：

增加两个指针变量和一个整型变量:

指针 **P1** 指向当前遍历的节点;指针 **P** 指向 **P1** 所指向节点的前 **K** 个节点,如果 **P1** 之前没有 **K** 个节点,那么 **P** 指向表头节点。整型变量 **i** 表示当前遍历了多少节点, 当  $i > k$  时, 指针 **p** 随着每次遍历, 也向前移动一个节点。当遍历完成时, **p** 或者指向表头节点, 或者指向链表中倒数第 **K** 个位置上的节点。

(3) 算法描述:

```
int LocateElement(linklist list, int k) {  
    linklist p1,p;  
    int i;  
    p1= list->link;  
    p=list;  
    i=1;  
    while(p1) {  
        p1=p1->link;  
        i++;  
        if(i>k)  
            p=p->link; //如果 i>k,则 p 也往后移  
    }  
    if(p==list)  
        return 0;//说明链表没有 k 个结点  
    else  
    {  
        printf("%d\n",p->data);  
        return 1;  
    }  
}
```

2、设将  $n(n,1)$  个整数存放于一维数组 **R** 中, 试设计一个在时间和空间两方面尽可能有效的算法, 将 **R** 中保有的序列循环左移 **P** ( $0 < P < n$ ) 个位置, 即将 **R** 中的数据由  $(X_1 X_2 \cdots X_n)$  变换为  $(X_p X_{p+1} \cdots X_n X_1 \cdots X_{p-1})$  要求:

- (1) 给出算法的基本设计思想。
- (2) 根据设计思想, 采用 **C** 或 **C++** 或 **JAVA** 语言表述算法, 关键之处给出注释。
- (3) 说明你所设计算法的时间复杂度和空间复杂度

参考答案:

(1) 基本设计思想:

将数组  $\{a_1, a_2, a_3, \dots, a_p, a_{p+1}, \dots, a_n\}$  先进行全部的逆转, 然后分别对  $\{a_p, \dots, a_{n-1}, a_n\}$  和  $\{a_1, a_2, a_3, \dots, a_p\}$  进行再次逆转。

(2) 算法描述:

```
Void sift_left (int a[],int n,int p) {  
    Reverse(a,0,n-1); //移动了  $3n/2$  次数据;  
    Reverse(a,0,n-p-1); //移动了  $3(n-p)/2$  次数据;  
    Reverse(a,n-p,n-1); //移动了  $3p/2$  次数据;  
    int Reverse(int A[],int left,int right)  
    {  
        int n=right-left+1; //设置一个辅助空间;  
        if(n<=1) return 0; //数组为空;  
        for(int i=0;i<n/2;i++){ //进行逆转;  
            int temp =A[left+i];  
            A[left+i]=A[left+n-i-1];  
            A[left+n-i-1]=temp;  
        }  
    }  
}
```

(3)

算法的时间复杂度和空间复杂度:

算法的时间复杂度为  $O(n)$ ;

算法的空间复杂度为  $O(1)$ ;

3、一个长度为  $L$  ( $L \geq 1$ ) 的升序序列  $S$ , 处在第  $\lfloor L/2 \rfloor$  个位置的数称为  $S$  的中位数。例如, 若序列  $S_1=(11,13,15,17,19)$ , 则  $S_1$  的中位数是 15, 两个序列的中位数是含它们所有元素在内的升序序列的中位数。例如, 若  $S_2=(2,4,6,8,20)$ , 则  $S_1$  和  $S_2$  的中位数是 11。现有两个等长的升序序列  $A, B$ , 试设计一个在时间和空间两方面都尽可能高效的算法, 找出两个序列  $A$  和  $B$  的中位数。要求:

(1) 给出算法的基本设计思想;

(2) 根据设计思想, 采用 C、C++ 或 JAVA 语言描述, 关键之处给出注释。

(3) 说明你所设计的算法的时间复杂度和空间复杂度。

参考答案:

(1) 算法的基本思想是:

分别求出 A, B 的中位数, 设为  $a, b$ , 求序列 A 和 B 的中位数的过程如下:

1) 若  $a=b$ , 则  $a$  或  $b$  即为所求的中位数, 将结果返回, 算法结束;

2) 若  $a < b$ , 则舍弃序列 A 中较小的一半和 B 中较大的一半, 要求舍弃的长度相等;

3) 若  $a > b$ , 则舍弃序列 A 中较大的一半和 B 中较小的一半, 要求舍弃的长度相等;

在保留的两个升序的序列中, 重复上述的 1,2,3 的过程, 直到两个序列中只含有一个元素是为止, 较小者即为所求的中位数。

(2) 算法的实现如下:

```
int midsearch(int A[],int B[],int n){
    int t1=0,w1=n-1,m1;
    int t2=0,w2=n-1,m2;
    while(t1!=w1 || t2!=w2){
        m1=(t1+w1)/2;
        m2=(t2+w2)/2;
        if(A[m1]==B[m2])
            return m1;
        if(A[m1]<B[m2])
        {
            if((t1+w1)%2==0){
                t1=m1;
                w2=m2;
            }//end if
        }
        else{
            t1=m1+1;
            w2=m2;
        }//end else
    }//end if
    else{
        if((t1+w1)%2==0){
            w1=m1;
            t2=m2;
        }//end if
        else{

```

```

        w1=m1;
        t2=m2+1;
    }//end else

    //end else
} //end while()
return A[t1]<B[t2]?A[t1]:B[t2];
} //end midsearch( )

```

(3)

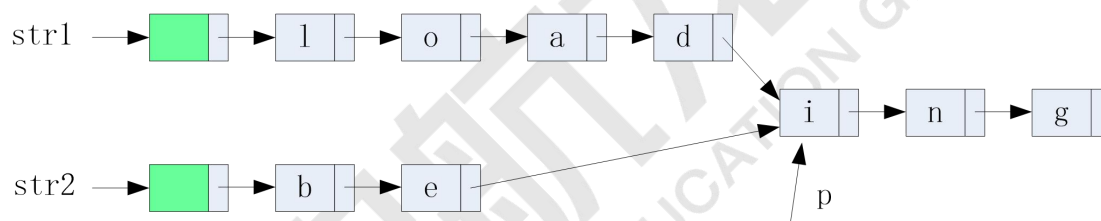
算法的时间复杂度和空间复杂度：

算法的时间复杂度为  $O(\log_2 n)$ ；

算法的空间复杂度  $O(1)$ ；

4、假定采用带头结点的单链表，如果有共同的后缀时，则可以共享相同的后缀存储空间。

例如，“loading”，“being”，如下图所示。



设 **str1** 和 **str2** 分别指向两个单词所在单链表的头结点，链表结点结构为：



设计一个时间上尽可能高效的算法，找出由 **str1** 和 **str2** 所指向两个链表共同后缀的起始位置（如图中字符 **i** 所在结点的位置 **p**）。要求：

- (1) 给出算法的基本设计思想。
- (2) 根据设计思想，采用 C 或 C++ 或 JAVA 语言描述算法，关键之处给出注释；
- (3) 说明你所设计算法的时间复杂度。

参考答案：

(1) 给出算法的基本设计思想：（4 分）

1) 分别求出 **str1** 和 **str2** 所指的两个链表长度 **m** 和 **n**；

2) 将两个链表以表尾对齐：令指针 **p, q** 分别指向 **str1** 和 **str2** 的头结点，若  $m \geq n$ ，则使 **p** 指向链表中的第  $m-n+1$  个结点；若  $m < n$ ，则使 **q** 指向同一结点。若 **p** 和 **q** 指向同一结点，则该点即为所求的共同后缀的起始位置。

(2) 算法实现：

```

Typedef struct Node
{
    Char data;
    Struct Node *next;
}SNode;
SNode *findlist(SNode *str1, SNode *str2)
{
    int m,n;
    SNode *p, *q;
    M=listlen(str1);
    N= listlen(str2);
    for(p=str1;m>n;m--)
        p=p->next;
    for(q=str2;m<n;n--)
        q=q->next;
    while(p->next!=NULL&& p->next!=q->next)
    {
        p=p->next;
        q=q->next;
    }
    return p->next;
}
int listen(SNode *head)
{
    int len=0;
    while(head->next!=NULL)
    {
        len++;
        head=head->next;
    }
    return len;}

```

### 第3章 栈、队列和数组



1、为解决计算机与打印机之间速度不匹配的问题，通常设置一个打印数据缓冲区，主机将要输出的数据依次写入该缓冲区，而打印机则依次从该缓冲区中取出数据。该缓冲区的逻辑结构应该是( )

- A.栈                      B.队列                      C.树                      D.图

参考答案：B

2、设栈 S 和队列 Q 的初始状态均为空，元素 abcdefg 依次进入栈 S。若每个元素出栈后立即进入队列 Q，且 7 个元素出队的顺序是 bdcfeag，则栈 S 的容量至少是( )

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

参考答案：C

3、若元素 a,b,c,d,e,f 依次进栈，允许进栈、退栈操作交替进行。但不允许连续三次进行退栈工作，则不可能得到的出栈序列是 ( )

- A: dcebf      B: cbdaef      C: bcaefd      D: afedcb

参考答案：D

4、某队列允许在其两端进行入队操作，但仅允许在一端进行出队操作，则不可能得到的顺序是 ( )

- A: bacde      B: dbace      C: dbcae      D: ecbad

参考答案：C

5、元素 a,b,c,d,e 依次进入初始非空的栈中。若元素进栈后可以停留，可以出栈，直到所有元素都出栈，则在所有的可能的出栈序列中，以元素 d 开头的序列的个数为 ( )

- A.3                      B.4                      C.5                      D.6

参考答案：B

6、已知循环队列存储在一维数组 A[0.....n-1]中，且队列非空时 front 和 rear 分别指向对头和队尾。若初始时队列为空，且要求第一个进入队列的元素存储在 A[0]处，则初始时 front 和 rear 的值分别为 ( )

- A.0,0                      B.0, n-1                      C.n-1, 0                      D.n-1,n-1

参考答案：B

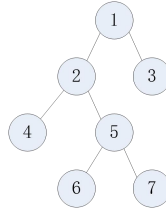
7、已知操作符包括 '+', '-', '\*', '/', '(' 和 ')', 将中缀表达式  $a+b-a*((c+d)/e-f)+g$  转化为等价的后缀表达式  $ab+acd+e/f-* -g+$  时，用栈来存放暂时还不能确定的运算次序的操作符，若栈初始时空，则转换过程中同时保存在栈中的操作符的最大个数是

- A. 5                      B. 7                      C. 8                      D. 11

参考答案：A

## 第4章 树与二叉树

1、给定二叉树图所示。设  $N$  代表二叉树的根， $L$  代表根结点的左子树， $R$  代表根结点的右子树。若遍历后的结点序列为 3, 1, 7, 5, 6, 2, 4, 则其遍历方式是 ( )



A. LRN

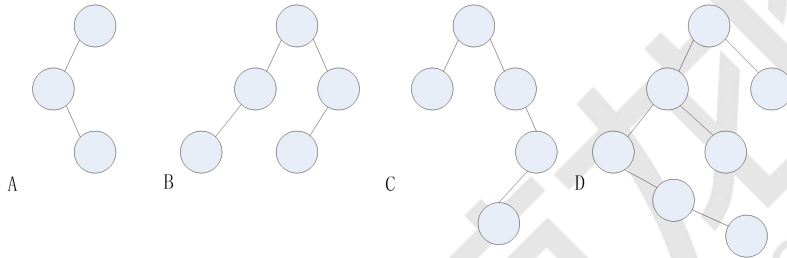
B.NRL

C.RLN

D.RNL

参考答案: D

2、下列二叉排序树中，满足平衡二叉树定义的是( )



参考答案: B

3、已知一棵完全二叉树的第 6 层（设根为第 1 层）有 8 个叶结点，则完全二叉树的结点个数最多是 ( )

A.39

B.52

C.111

D.119

参考答案: C

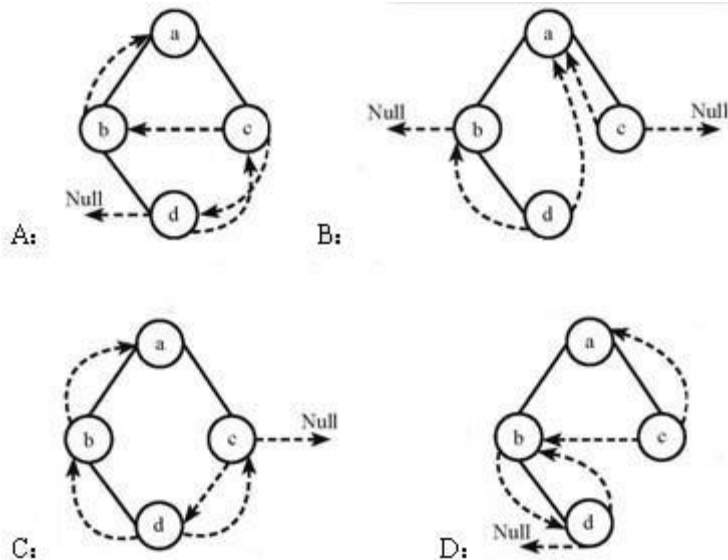
4、将森林转换为对应的二叉树，若在二叉树中，结点  $u$  是结点  $v$  的父结点的父结点，则在原来的森林中， $u$  和  $v$  可能具有的关系是

I. 父子关系    II. 兄弟关系    III.  $u$  的父结点与  $v$  的父结点是兄弟关系

A. 只有 II    B. I 和 II    C. I 和 III    D. I、II 和 III

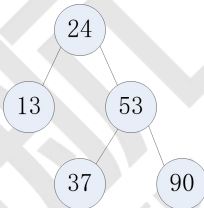
参考答案: B

5、下列线索二叉树中（用虚线表示线索），符合后序线索树定义的是 ( )



参考答案: D

6、在下列所示的平衡二叉树中插入关键字 48 后得到一棵新平衡二叉树，在新平衡二叉树中，关键字 37 所在结点的左、右子结点中保存的关键字分别是 ( )



A: 13, 48      B: 24, 48      C: 24, 53      D: 24, 90

参考答案:C

7、在一棵度为 4 的树 T 中，若有 20 个度为 4 的结点，10 个度为 3 的结点，1 个度为 2 的结点，10 个度为 1 的结点，则树 T 的叶节点个数是 ( )

A: 41      B: 82      C: 113      D: 122

参考答案: B

8、对  $n$  ( $n$  大于等于 2) 个权值均不相同的字符构成哈夫曼树，关于该树的叙述中，错误的是 ( )

- A: 该树一定是一棵完全二叉树
- B: 树中一定没有度为 1 的结点
- C: 树中两个权值最小的结点一定是兄弟结点
- D: 树中任一非叶结点的权值一定不小于下一任一结点的权值

参考答案: A

9、若一颗完全二叉树有 768 各节点，则该二叉树叶节点的个数为( )

A.257

B.258

C.384

D.385

参考答案: C

10、若一颗二叉树的前序遍历和后序遍历分别为 1,2,3,4 和 4,3,2,1, 则该二叉树的中序遍历不会是 ( )

A.1,2,3,4

B.2,3,4,1

C.3,2,4,1

D.4,3,2,1

参考答案: C

11、已知一颗有 2011 个结点的树, 其叶子结点的个数为 116, 该树对应的二叉树中无右孩子的结点个数最多是 ( )

A.115

B.116

C.1895

D.1896

参考答案: D

12、对于下列关键序列, 不能构成某二叉树排序中的一条查找路径的序列是 ( )

A.95, 22, 91, 24, 94, 71

B.92, 20, 91, 34, 88, 35

C.21, 89, 77, 29, 36, 38

D.12, 25, 71, 68, 33, 34

参考答案: A

13、若一棵二叉树的先序遍历序列是 a,e,b,d,c, 后序遍历序列为 b,c,d,e,a, 则根结点的孩子结点 ( )。

A. 只有 e

B. 有 e、b

C. 有 e、c

D. 无法确定

参考答案: A

14、若平衡二叉树的高度为 6, 且所有非叶结点的平衡因子均为 1, 则该平衡二叉树的结点总数为( )

A. 10

B. 20

C. 32

D. 33

参考答案: B

## 第5章 图

1、下列关于无向连通图特性的叙述中, 正确的是

I. 所有顶点的度之和为偶数

II. 边数大于顶点个数减 1

III. 至少有一个顶点的度为 1

A. 只有 I

B. 只有 II

C. I 和 II

D. I 和 III

参考答案: A

2、若无向图  $G=(V,E)$  中含 7 个顶点, 则保证图 G 在任何情况下都是连通的, 则需要的边数最少是 ( )

A : 6

B: 15

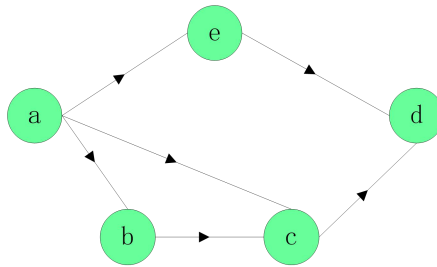
C: 16

D: 20

参考答案: C

3、对下图进行拓补排序，可以得到不同的拓补序列的个数是（ ）

- A: 4      B: 3      C: 2      D: 1



参考答案: B

4、下列关于图的叙述中正确的是（ ）

I 回路是简单路径

II 存储稀疏图，用邻接矩阵比邻接表更省空间

III 若有向图中存在拓扑序列，则该图不存在回路

- 仅 I      仅 I, II      C. 仅 III      D. 仅 I, III

参考答案: C

5、对有  $n$  个顶点、 $e$  条边且使用邻接表存储的有向图进行广度优先遍历，其算法时间复杂度是（ ）。

- A.  $O(n)$       B.  $O(e)$       C.  $O(n+e)$       D.  $O(n*e)$

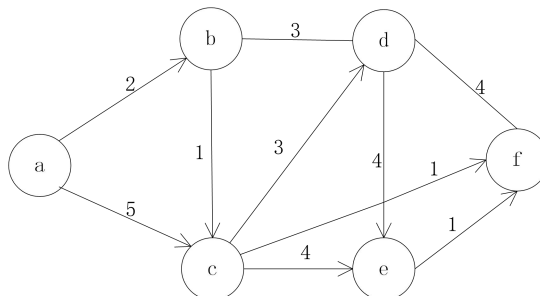
参考答案: C

6、若用邻接矩阵存储有向图，矩阵中主对角线以下的元素均为零，则关于该图的拓扑排序的结论是（ ）。

- A. 存在，且唯一      B. 存在，且不唯一  
C. 存在，可能不唯一      D. 无法确定是否存在

参考答案: C

7、对如下有向带权图，若采用地杰斯特拉算法求从源点  $a$  到其他各顶点的最短路径，则得到的第一条最短路径的目标顶点是  $b$ ，第二条最短路径的目标顶点是  $c$ ，后续得到的其余各最短路径的目标顶点依次是（ ）。



A. d,e,f

B.e,d,f

C.f,d,e

D.f,e,d

参考答案: C

8、下列关于最小生成树的叙述中, 正确的是 ( )。

(1)、最小生成树的代价唯一

(2)、所有权值最小的边一定会出现在所有的最小生成树中

(3)、使用普里姆算法从不同顶点开始得到的最小生成树一定相同

(4)、使用普里姆算法和克鲁斯卡尔算法得到的最小生成树总不相同

A. 仅 1

B.仅 2

C.仅 1、3

D.仅 2、4

参考答案 A.

9、带权图(权值非负, 表示边连接的两顶点间的距离)的最短路径问题是找出从初始顶点到目标顶点之间的一条最短路径。假定从初始顶点到目标顶点之间存在路径, 现有一种解决该问题的方法:

①设最短路径初始时仅包含初始顶点, 令当前顶点  $u$  为初始顶点;

②选择离  $u$  最近且尚未在最短路径中的一个顶点  $v$ , 加入到最短路径中, 修改当前顶点  $u=v$ ;

③重复步骤②, 直到  $u$  是目标顶点时为止。

请问上述方法能否求得最短路径? 若该方法可行, 请证明之; 否则, 请举例说明。

参考答案: 该方法不一定能求得最短路径。例如, 图 a 中, 设初始顶点为 1, 目标顶点为 4, 欲求从顶点 1 到顶点 4 之间的最短路径。显然, 这两点之间的最短路径长度为 2。但利用给定方法求得的路径长度为 3, 因此这条路径并不是这两点之间的最短路径。图 b 中, 设初始顶点为 1, 目标顶点为 3, 欲求从顶点 1 到顶点 3 之间的最短路径。利用给定的方法, 无法求出顶点 1 到顶点 3 的路径。

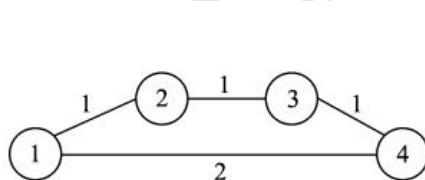


图 a

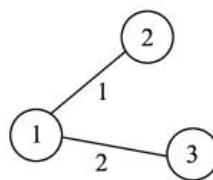


图 b

10、已知有 6 个顶点(顶点编号为 0-5)的有向带权图  $G$ , 其邻接矩阵  $A$  为上三角阵, 按行为主序(行优先)保存在下面的一维数组中:

4	6	$\infty$	$\infty$	$\infty$	1	$\infty$	$\infty$	$\infty$	4	3	$\infty$	$\infty$	3	3
---	---	----------	----------	----------	---	----------	----------	----------	---	---	----------	----------	---	---

要求:

(1) 写出图  $G$  的邻接矩阵  $A$ ;

(2) 画出有向带权图;

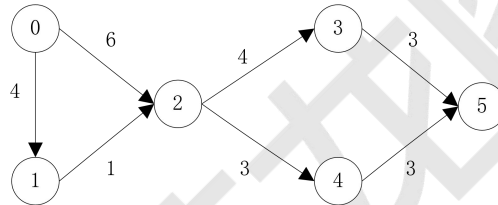
(3) 求图 G 的关键路径, 并计算该关键路径的长度;

参考答案: 本题是一道综合题目, 但是不难, 中间有一个小的陷阱, 就是对角线上的元素都是 0, 所以没有存对角线上的元素。

邻接矩阵如下:

0	4	6	$\infty$	$\infty$	$\infty$
$\infty$	0	1	$\infty$	$\infty$	$\infty$
$\infty$	$\infty$	0	4	3	$\infty$
$\infty$	$\infty$	$\infty$	0	$\infty$	3
$\infty$	$\infty$	$\infty$	$\infty$	0	3
$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	0

有向图为:



关键路径为:

0~2~3~5。该关键路径长度为 13。

## 第6章 查找

1、下列叙述中, 不符合 m 阶 B 树定义要求的是 ( )

A. 根节点最多有 m 棵子树      B. 所有叶结点都在同一层上

C. 各结点内关键字均升序或降序排列      D. 叶结点之间通过指针链接

参考答案: D

2、已知一个长度为 16 的顺序表 L, 其元素按关键字有序排列, 若采用折半查找法查找一个不存在的元素, 则比较次数最多是 ( )

A: 4      B: 5      C: 6      D: 7

参考答案: B

3、为提高散列表的查找效率, 可以采取的正确措施 ( )

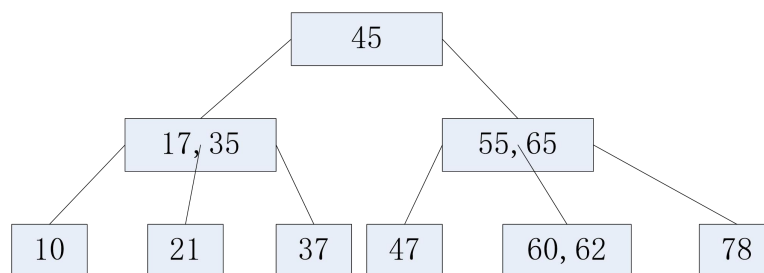
I 增大装填因子      II 设计冲突少的散列函数

III 处理冲突时, 避免产生聚集现象

A 仅 I      B. 仅 II      C. 仅 I, II      D. 仅 II, III

参考答案: D

4、已知一颗 B 树, 如下图所示。删除关键字 78 得到一颗新 B 树, 其最右叶节点中的关键字 ( )



- A. 60                      B. 60,62                      C. 62,65                      D. 65

参考答案：D

5、将关键字序列（7、8、30、11、18、9、14、）散列存储到散列列表中，散列表的存储空间是一个下标从0开始的一个一维数组散列函数维： $H(\text{key}) = (\text{key} \times 3) \text{MOD } 7$ ，处理冲突采用线性探测再散列法，要求装填（载）因子为0.7。

问题：

- （1）请画出所构造的散列表；
- （2）分别计算等概率情况下，查找成功和查找不成功的平均查找长度。

参考答案：（1）由装载因子0.7，数据总数7个，所以存储空间长度为10。所以，构造的散列表的哈希地址为：

$$H(7) = (7 \times 3) \text{MOD } 7 = 0$$

$$H(8) = (8 \times 3) \text{MOD } 7 = 3$$

$$H(11) = (11 \times 3) \text{MOD } 7 = 5$$

$$H(18) = (18 \times 3) \text{MOD } 7 = 5$$

$$H(9) = (9 \times 3) \text{MOD } 7 = 6$$

$$H(14) = (14 \times 3) \text{MOD } 7 = 0$$

$$H(30) = (30 \times 3) \text{MOD } 7 = 6$$

根据哈希地址可以得出如下图所示的哈希表。

下标	0	1	2	3	4	5	6	7	8	9
关键字	7	14		8		11	30	18	9	

②、平均查找长度

查找成功的平均查找长度:ASL 成功  $= (1+1+1+1+2+3+3)/7 = 12/7$ 。

查找不成功的平均查找长度:ASL 不成功  $= (3+2+1+2+1+5+4)/7 = 18/7$ 。

## 第7章 排序

1、已知关键序列 5，8，12，19，28，20，15，22 是小根堆（最小堆），插入关键字 3，调



整后得到的小根堆是 ( )

- A.3, 5, 12, 8, 28, 20, 15, 22, 19
- B.3, 5, 12, 19, 20, 15, 22, 8, 28
- C.3, 8, 12, 5, 20, 15, 22, 28, 19
- D.3, 12, 5, 8, 28, 20, 15, 22, 19

参考答案: A

2、若数据元素序列 11, 12, 13, 7, 8, 9, 23, 4, 5 是采用下列排序方法之一得到的第二趟排序后的结果, 则该排序算法只能是

- A.起泡排序
- B.插入排序
- C.选择排序
- D.二路归并排序

参考答案: B

3、采用递归方式对顺序表进行快速排序, 下列关于递归次数的叙述中, 正确的是 ( )

- A: 递归次数与初始数据的排列次序无关
- B: 每次划分后, 先处理较长的分区可以减少递归次数
- C: 每次划分后, 先处理较短的分区可以减少递归次数
- D: 递归次数与每次划分后得到的分区处理顺序无关

参考答案: D

4、对一组数据 (2, 12, 16, 88, 5, 10) 进行排序, 若前三趟排序结果如下

第一趟: 2, 12, 16, 5, 10, 88

第二趟: 2, 12, 5, 10, 16, 88

第三趟: 2, 5, 10, 12, 16, 88

则采用的排序方法可能是 ( )

- A.起泡排序
- B.希尔排序
- C.归并排序
- D.基数排序

参考答案:A

5、为实现快速排序算法, 待排序的序列宜采用的存储方式是 ( )

- A.顺序存储
- B.散列存储
- C.链式存储
- D.索引存储

参考答案: A

6、已知序列 25,13,10,12,9 是大根堆, 在序列尾部插入新元素 18, 将其再调整为大根堆, 调整过程中元素之间的比较次数是 ( )

- A.1
- B.2
- C.4
- D.5

参考答案：B

7、在内部排序过程中，对尚未确定最终位置的所有元素进行一遍处理称为一趟排序。下列排序方法中，每一趟排序结束都至少能够确定一个元素最终位置的方法（ ）

- I.简单选择排序      II.希尔排序      III.快速排序      IV 堆 排 序  
V.二路归并排序

A. 仅 I，III，IV    B. 仅 I，III，V    C. 仅 II，III，IV    D. 仅 III，IV，V

参考答案：A

8、对同一待排序序列分别进行折半插入排序和直接插入排序，两者之间的不同之处是（ ）

- A. 排序的总趟数      B. 元素的移动次数  
C. 使用辅助空间的数量      D. 元素之间的比较次数

参考答案：D



20计算机考研群：738222741