窗体顶端

# 数据结构与算法基础

您现在的位置：[希赛网](http://www.educity.cn/)>[云阅读](http://www.educity.cn/jiaocheng/)>[软件设计师考试试题分类精解（2018版）](http://www.educity.cn/jiaocheng/zt251.html)>**试题1(2017年下半年试题57)**

第9章：数据结构与算法基础作者：[希赛软考学院](http://www.educity.cn/jiaocheng/a%cf%a3%c8%fc%c8%ed%bf%bc%d1%a7%d4%ba.html)来源：希赛软考学院2017年11月21日

# 试题1(2017年下半年试题57)

设S是一个长度为n的非空字符串，其中的字符各不相同，则其互异的非平凡子串（非空且不同于S本身）个数为（）。  
（57）A．2n-1  
B.n²  
C.n(n+1)/2  
D.(n+2)(n-1)/2

**试题分析**

比如S字串为"abcdefg",长度为7.则S中的包含的互不相同的字串有如下一些：  
1.长度为6的个数为2："abcdef"和"bcdefg"  
2.长度为5的个数为3："abcde","bcdef","cdefg"  
.  
6.长度为1的个数为7："a","b","c","d","e","f","g"  
个数总和就是2+3+4+5+6+7=(1+2+3+..+7)-1=7x(7+1)/2-1.  
其中：  
1+2+3+...+n=(1+n)+(2+(n-1))+(3+(n-2))+...（首尾两项相加的和都是n+1,共n/2个）  
=n(n+1)/2

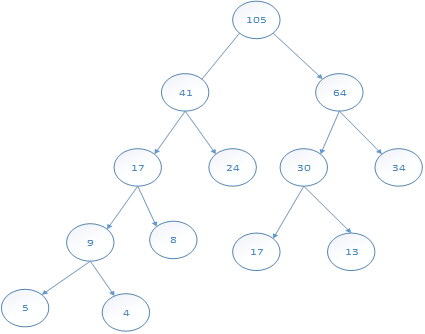
**试题答案**

（57）C

# 试题2(2017年下半年试题58)

假设某消息中只包含7个字符{a，b，c，d，e，f，g}，这7个字符在消息中出现的次数为{5，24，8，17，34，4，13}，利用哈夫曼树（最优二叉树）为该消息中的字符构造符合前缀编码要求的不等长编码。各字符的编码长度分别为（）。  
（58）A．a:4,b:2,c:3,d:3,e:2,f:4,g:3  
B.a:6,b:2,c:5,d:3,e:1,f:6,g:4  
C.a:3,b:3,c:3,d:3,e:3,f:2,g:3  
D.a:2,b:6,c:3,d:5,e:6,f:1,g:4

**试题分析**

哈夫曼树  


**试题答案**

（58）A

# 试题3(2017年下半年试题59)

设某二叉树采用二叉链表表示（即结点的两个指针分别指示左、右孩子）。当该二叉树包含k个结点时，其二叉链表结点中必有（）个空的孩子指针。  
（59）A．k-1  
B.k  
C.k+1  
D.2k

**试题分析**

二叉树的的二叉链表存储结构中每个结点有2个指针。每个结点有0个、1个或者2个空指针对应有2个、1个、0个非空指针。  
二叉树中边的个数等于非空指针的个数。  
假设二叉树中节点的总个数为N,  
假设二叉树中边的个数为M  
假设二叉树中度为0的结点的个数为n0,  
假设二叉树中度为1的结点的个数为n1，  
假设二叉树中度为2的结点的个数为n2.  
所以有n0+n1+n2=N-------------（1）  
二叉树中除了根结点之外，其他的结点都有一条便进入该结点，所以二叉树中边的总个数为M=N-1;-------（2）  
又M=n1+2\*n2;-------------------------（3）  
所以由（1）（2）（3）可得n0=n2+1;--------------------（4）  
设空节点的个数为K,则K=2\*n0+n1-------------------(5)  
结合（1）（4）（5）可以得到K=N+1.(空指针的的个数比结点总个数多1)  
由（2）可以知道边数M=N-1;(二叉树的边数为结点个数减1)  
由（4）可以知道度为0的结点的个数（叶子结点个数）=度为2的结点个数+1（n0=n2+1;）。

**试题答案**

（59）C

# 试题4(2017年下半年试题60)

以下关于无向连通图G的叙述中，不正确的是（）。  
（60）A．G中任意两个顶点之间均有边存在  
B.G中任意两个顶点之间存在路径  
C.从G中任意顶点出发可遍历图中所有顶点  
D.G的邻接矩阵是对称矩阵

**试题分析**

无向连通图不一定有边，但两个顶点之间有路径。P460。

**试题答案**

（60）A

# 试题5(2017年下半年试题61)

两个递增序列A和B的长度分别为m和n（m<n且m与n接近），将二者归并为一个长度为m+n的递增序列。当元素关系为（）时，归并过程中元素的比较次数最少。  
（61）A．a1<a2<…<am-1<am<b1<b2<…<bn-1<bn  
B.b1<b2<…<bn-1<bn<a1<a2<…<am-1<am  
C.a1<b1<a2<b2<…<am-1<bm-1<am<bm<bm+1<…<bn-1<bn  
D.b1<b2<…<bm-1<bm<a1<a2<…<am-1<am<bm+1<…<bn-1<bn

**试题分析**

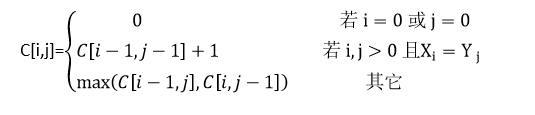
若A的最大元素小于B的最小元素，则只需要比较m次，这时归并过程中元素的比较次数最少。

**试题答案**

（61）A

# 试题6(2017年下半年试题62-63)

求解两个长度为n的序列X和Y的一个最长公共子序列（如序列ABCBDAB和BDCABA的一个最长公共子序列为BCBA）可以采用多种计算方法。如可以采用蛮力法，对X的每一个子序列，判断其是否也是Y的子序列，最后求出最长的即可，该方法的时间复杂度为（）。经分析发现该问题具有最优子结构，可以定义序列长度分别为i和j的两个序列X和Y的最长公共子序列的长度为C[i,j]，如下式所示。



（62）A．O(n²)  
B.O(n²lgn)  
C.O(n³)  
D.O(n2n)  
  
（63）A．O(n²)  
B.O(n²lgn)  
C.O(n³)  
D.O(n2n)

**试题分析**

1.X、Y的所有子序列都检查过后即可求出X、Y的最长公共子序列。X的一个子序列相应于下标序列1,2，...，n的一个子序列。因此，X共有2^n个子序列。当然，Y也有2^m个子序列。  
2.动态规划的一个计算最长公共子序列的方法如下，两个序列X、Y：  
设有二维数组c[i][j]表示X的i位和Y的j位之前的最长公共子序列的长度，则有题干给定的函数表现形式  
其中，c(i,j)当X的第i位与Y的第j位完全相同时为“1”，否则为“0”。  
此时，c[i][j]中最大的数便是X和Y的最长公共子序列的长度，依据该数组回溯，便可找出最长公共子序列。该算法的空间、时间复杂度均为O(n^2)。

**试题答案**

（62）D（63）A

# 试题7(2017年下半年试题64-65)

现需要对一个基本有序的数组进行排序。此时最适宜采用的算法为（）排序算法，时间复杂度为（）。  
（64）A．插入  
B.快速  
C.归并  
D.堆  
  
（65）A．O(n)  
B.O(nlgn)  
C.O(n²)  
D.O(n²lgn)

**试题分析**

1.快速排序（Quicksort）是对冒泡排序法的一种改进，它的基本思想是：通过一趟排序将要排序的数据分割成独立的两部分，其中一部分的所有数据都比另外一部分的所有数据都要小，然后再按此方法对这两部分数据分别进行快速排序，整个排序过程可以递归进行，以此达到整个数据变成有序序列；在对一个基本有序的数组进行排序时适合采用快速排序法。  
2.快速排序的平均运行时间为O(nlogn)。

**试题答案**

（64）B（65）B

# 试题8(2017年上半年试题58)

已知栈S初始为空，用I表示入栈、O表示出栈，若入栈序列为a1a2a3a4a5，则通过栈S得到出栈序列a2a4a5a3a1的合法操作序列（）。  
（58）A．IIOIIOIOOO  
B.IOIOIOIOIO  
C.IOOIIOIOIO  
D.IIOOIOIOOO

**试题分析**

IIOIIOIOOO出栈序列为：a2a4a5a3a1  
IOIOIOIOIO出栈序列为：a1a2a3a4a5  
IOOIIOIOIO无合法出栈序列，因为入栈1个元素，出栈2个元素，会产生错误。  
IIOOIOIOOO无合法出栈序列，操作序列中4次入栈6次出栈也是会产生错误的。

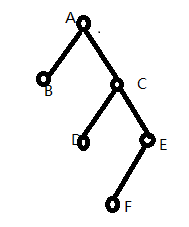
**试题答案**

（58）A

# 试题9(2017年上半年试题59)

某二叉树的先序遍历序列为ABCDEF，中序遍历序列为BADCFE，则该二叉树的高度(即层数)为（）。  
（59）A．3  
B.4  
C.5  
D.6

**试题分析**

先序遍历即先根后左子树再右子树，中序遍历为先左子树后跟再右子树。先序遍历的最开始结点A即为整棵树的根，结合中序遍历，A结点左侧B即为根节点A的左子树，右侧DCFE则为A的右子树，同理可以得出C为A的右子树的根节点,D为C的左子树，EF为C的右子树，F为E的左子树。可以得到如下图，所以该二颗树的高度为4。  


**试题答案**

（59）B

# 试题10(2017年上半年试题60)

对于n个元素的关键宇序列{k1,k2,...kn}，当且仅当满足关系ki≤k2i且ki≤k2i+1{i=1.2...[n/2]}时称其为小根堆(小顶堆)。以下序列中，（）不是小根堆。  
  
（60）A．16,25,40,55,30,50,45  
B.16,40,25,50,45,30,55  
C.16,25,39.,41,45,43,50  
D.16,40,25,53,39,55,45

**试题分析**

D答案中第二个关键字小于第五个关键字，不满足小根堆的条件。

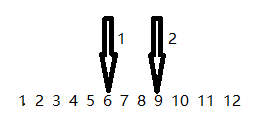
**试题答案**

（60）D

# 试题11(2017年上半年试题61)

在12个互异元素构成的有序数组a[1..12]中进行二分查找(即折半查找，向下取整)，若待查找的元素正好等于a[9]，则在此过程中，依次与数组中的（）比较后，查找成功结束。  
  
（61）A．a[6]、a[7]、a[8]、a[9]  
B.a[6]、a[9]  
C.a[6]、a[7]、a[9]  
D.a[6]、a[8]、a[9]

**试题分析**

二分查找的基本思想是将n个元素分成大致相等的两部分，取a[n/2]与x做比较，如果x=a[n/2],则找到x,算法中止；如果x<a[n/2],则只要在数组a的左半部分继续搜索x,如果x>a[n/2],则只要在数组a的右半部搜索x.故查找顺序如下图所示：  


**试题答案**

（61）B

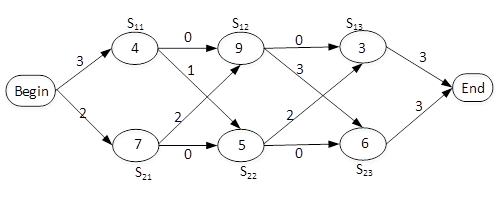
# 试题12(2017年上半年试题62-65)

某汽车加工工厂有两条装配线L1和L2，每条装配线的工位数均为n（Sij，i=1或2，j=1，2，...，n），两条装配线对应的工位完成同样的加工工作，但是所需要的时间可能不同（aij，i=1或2，j=1，2，...，n）。汽车底盘开始到进入两条装配线的时间(e1，e2)以及装配后到结束的时间(X1X2)也可能不相同。从一个工位加工后流到下一个工位需要迁移时间(tij，i=1或2，j=2，...n）。现在要以最快的时间完成一辆汽车的装配，求最优的装配路线。  
分析该问题，发现问题具有最优子结构。以L1为例，除了第一个工位之外，经过第j个工位的最短时间包含了经过L1的第j-1个工位的最短时间或者经过L2的第j-1个工位的最短时间，如式(1)。装配后到结束的最短时间包含离开L1的最短时间或者离开L2的最短时间如式（2）。

IMG_272若j=1其他（1）

IMG_273（2）

由于在求解经过L1和L2的第j个工位的最短时间均包含了经过L1的第j-1个工位的最短时间或者经过L2的第j-1个工位的最短时间，该问题具有重复子问题的性质，故采用迭代方法求解。  
该问题采用的算法设计策略是（），算法的时间复杂度为（）  
以下是一个装配调度实例，其最短的装配时间为（），装配路线为（）



（62）A．分治  
B.动态规划  
C.贪心  
D.回溯  
  
（63）A．O(lgn)  
B.O(n)  
C.O(n2)  
D.O(nlgn)  
  
（64）A．21  
B.23  
C.20  
D.26  
  
（65）A．S11→S12→S13  
B.S11→S22→S13  
C.S21→S12→S23  
D.S21→S22→S23

**试题分析**

本题考查算法基础。题目看似是非常复杂的，涉及到复杂的公式，以及算法逻辑，但如果我们先从后面两个空来分析，问题就简单得多。  
求最短装配时间与装配路线，其实是一个求最短路径的过程。此时我们可以把从起点到各个结点的最短路径逐步求出。经过分析得出最短装配路线为：S11->S22->S13，长度为21。  
解决了一个实际问题后，再来看所谓的迭代公式，其做法与之前手动求最短路径一致，算法是用一个数组将起点到各个结点的最短路径逐个求出，用已求出的最短路径来分析后面的最短路径，所以这符合动态规划法的特征，算法策略应是动态规划法。而算法的复杂度为O（n），因为用一个单重循环就可以解决这个问题。

**试题答案**

（62）B（63）B（64）A（65）B

# 试题13(2016年下半年试题22)

二维数组a[1..N，1..N]可以按行存储或按列存储。对于数组元素a[i,j]（1<=i,j<=N），当（）时，在按行和按列两种存储方式下，其偏移量相同。  
  
（22）A．i≠j  
B.i=j  
C.i>j  
D.i<j

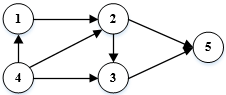
**试题分析**

i和j相等，那么这时候的行列是一样多的，则按行按列变得没有区别

**试题答案**

（22）B

# 试题14(2016年下半年试题57)

拓扑序列是有向无环图中所有顶点的一个线性序列，若有向图中存在弧<v，w>或存在从顶点v到w的路径，则在该有向图的任一拓扑序列中，v一定在w之前。下面有向图的拓扑序列是（）。  
  
（57）A．41235  
B.43125  
C.42135  
D.41325

**试题分析**

拓扑排序通俗一点来讲，其实就是依次遍历没有前驱结点的结点。而某一时刻没有前驱结点的结点有可能存在多个，所以一个图的拓扑排序可能有多个。  
4号结点没有前戏，所以拓扑排序的第一个元素是4。当4访问完了就可以访问1，1号访问完了就可以访问2，2号访问完了就可以访问3或5。所以拓扑排序结果为：412(35)。

**试题答案**

（57）A

# 试题15(2016年下半年试题58-59)

设有一个包含n个元素的有序线性表。在等概率情况下删除其中的一个元素，若采用顺序存储结构，则平均需要移动（）个元素；若采用单链表存储，则平均需要移动（）个元素。  
（58）A．1  
B.(n-1)/2  
C.logn  
D.n  
  
（59）A．0  
B.1  
C.(n-1)/2  
D.n/2

**试题分析**

若用顺序表存储，则最好情况是删除最后一个元素，此时不用移动任何元素，直接删除，最差的情况是删除第一个元素，此时需要移动n-1个元素，所以平均状态是移动(n-1)/2。

若用链表存储，直接将需要删除元素的前趋next指针指向后继元素即可，不需要移动元素，所以移动元素个数为0。

**试题答案**

（58）B（59）A

# 试题16(2016年下半年试题60)

具有3个节点的二叉树有（）种形态。  
（60）A．2  
B.3  
C.5  
D.7

**试题分析**

N个节点（N>=2）的二叉树有IMG_276

**试题答案**

（60）C

# 试题17(2016年下半年试题61)

以下关于二叉排序树（或二叉查找树、二叉搜索树）的叙述中，正确的是（）。  
  
（61）A．对二叉排序树进行先序、中序和后序遍历，都得到结点关键字的有序序列  
B.含有n个结点的二叉排序树高度为⌊log2n⌋+1  
C.从根到任意一个叶子结点的路径上，结点的关键字呈现有序排列的特点  
D.从左到右排列同层次的结点，其关键字呈现有序排列的特点

**试题分析**

二叉排序树或者是一棵空树，或者是具有下列性质的二叉树：  
（1）若左子树不空，则左子树上所有结点的值均小于或等于它的根结点的值；  
（2）若右子树不空，则右子树上所有结点的值均大于或等于它的根结点的值；  
（3）左、右子树也分别为二叉排序树  
那么同层次的节点，右子树大于根节点，根节点大于左子树，则右子树大于左子树，则同层次有序排列。

**试题答案**

（61）D

# 试题18(2016年下半年试题62-63)

下表为某文件中字符的出现频率，采用霍夫曼编码对下列字符编码，则字符序列“bee”的编码为（）；编码“110001001101”的对应的字符序列为（）。  
IMG_277（62）A．10111011101  
B.10111001100  
C.001100100  
D.110011011  
  
（63）A．bad  
B.bee  
C.face  
D.bace

**试题分析**

110001001101中：f(1100)a(0)c(100)e(1101)。

**试题答案**

（62）A（63）C

# 试题19(2016年下半年试题64-65)

两个矩阵Am\*n和Bn\*p相乘，用基本的方法进行，则需要的乘法次数为m\*n\*p。多个矩阵相乘满足结合律，不同的乘法顺序所需要的乘法次数不同。考虑采用动态规划方法确定Mi，M(i+1)，…，Mj多个矩阵连乘的最优顺序，即所需要的乘法次数最少。最少乘法次数用m[i,j]表示，其递归式定义为：  
IMG_278  
其中i、j和k为矩阵下标，矩阵序列中Mi的维度为（pi-1）\*pi采用自底向上的方法实现该算法来确定n个矩阵相乘的顺序，其时间复杂度为（）。若四个矩阵M1、M2、M3、M4相乘的维度序列为2、6、3、10、3，采用上述算法求解，则乘法次数为（）。  
（64）A．O（n2）  
B.O（n2lgn）  
C.O（n3）  
D.O（n3lgn）  
  
（65）A．156  
B.144  
C.180  
D.360

**试题分析**

四个矩阵分别为：  
2\*66\*33\*1010\*3  
先计算：M1\*M2及M3\*M4，计算次数分别为：  
2\*6\*3=36，3\*10\*3=90。  
然后结果相乘，计算次数为：  
2\*3\*3=18。  
36+90+18=144。

**试题答案**

（64）C（65）B

# 试题20(2016年上半年试题57)

若元素以a，b，c，d，e的顺序进入一个初始为空的栈中，每个元素进栈、出栈各1次，要求出栈的第一个元素为d，则合法的出栈序列共有（）种。  
（57）A．4  
B.5  
C.6  
D.24

**试题分析**

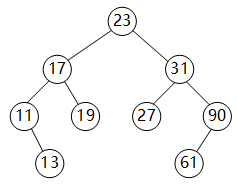
一共5个元素a,b,c,d,e，而d被要求作为第一个元素出栈。当d出栈后的情况应为：  
有一个元素e还未入栈，而栈中已有a，b，c。栈中的a，b，c出栈顺序是已无可变性，必须是：c，b，a，此时，只是分析e在什么位置出栈即可。  
c，b，a，三个元素，有四个空位，所以可以产生的序列可能为：  
（1）d，e，c，b，a  
（2）d，c，e，b，a  
（3）d，c，b，e，a  
（4）d，c，b，a，e

**试题答案**

（57）A

# 试题21(2016年上半年试题58)

设有二叉排序树（或二叉查找树）如下图所示，建立该二叉树的关键码序列不可能是（）。



（58）A．233117191127139061  
B.231719312790611113  
C.231727193113119061  
D.233190612717191113

**试题分析**

注意27在31的后面可以得到答案C

**试题答案**

（58）C

# 试题22(2016年上半年试题59)

若一棵二叉树的高度（即层数）为h，则该二叉树（）。  
（59）A．有2h个结点  
B.有2h-1个结点  
C.最少有2h-1个结点  
D.最多有2h-1个结点

**试题分析**

一颗高度为h的二叉树，结点数最多时，即为满二叉树。  
而高度为h的满二叉树有2h-1个结点，所以一棵二叉树的高度（即层数）为h，则它最多有2h-1个结点。

**试题答案**

（59）D

# 试题23(2016年上半年试题60)

在13个元素构成的有序表A[1..13]中进行折半查找（或称为二分查找，向下取整）。那么以下叙述中，错误的是（）。  
（60）A．无论要查找哪个元素，都是先与A[7]进行比较  
B.若要查找的元素等于A[9]，则分别需与A[7]、A[11]、A[9]进行比较  
C.无论要查找的元素是否在A[]中，最多与表中的4个元素比较即可  
D.若待查找的元素不在A[]中，最少需要与表中的3个元素进行比较

**试题分析**

B选项错误之处在于，要查找a[9]元素，第一次比较的是A[7]（下标计算方法为：[1+13]/2=7），第2次比较的是A[10]（下标计算方法为：[8+13]/2=10）。  
注意：题目要求计算下标时，向下取整。

**试题答案**

（60）B

# 试题24(2016年上半年试题61)

以下关于图的遍历的叙述中，正确的是（）。  
（61）A．图的遍历是从给定的源点出发对每一个顶点仅访问一次的过程  
B.图的深度优先遍历方法不适用于无向图  
C.使用队列对图进行广度优先遍历  
D.图中有回路时则无法进行遍历

**试题分析**

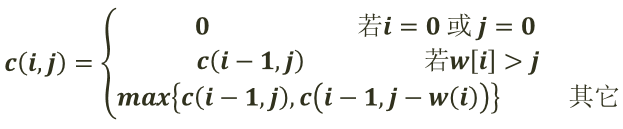
队列的特点是先进先出，广度优先刚好合适

**试题答案**

（61）C

# 试题25(2016年上半年试题62-65)

考虑一个背包问题，共有n=5个物品，背包容量为W=10，物品的重量和价值分别为：w={2，2，6，5，4}，v={6,3，5，4，6}，求背包问题的最大装包价值。若此为0-1背包问题，分析该问题具有最优子结构，定义递归式为



其中c（i，j）表示i个物品、容量为j的0-1背包问题的最大装包价值，最终要求解c（n,W）。

采用自底向上的动态规划方法求解，得到最大装包价值为（），算法的时间复杂度为（）。

若此为部分背包问题，首先采用归并排序算法，根据物品的单位重量价值从大到小排序，然后依次将物品放入背包直至所有物品放入背包中或者背包再无容量，则得到的最大装包价值为（），算法的时间复杂度为（）。

（62）A．11  
B.14  
C.15  
D.16.67  
  
（63）A．Θ(nW)  
B.Θ(nlgn)  
C.Θ(n2)  
D.Θ(nlgnW)  
  
（64）A．11  
B.14  
C.15  
D.16.67  
  
（65）A．Θ(nW)  
B.Θ(nlgn)  
C.Θ(n2)  
D.Θ(nlgnW)

**试题分析**

这是典型的01背包问题，动态规划算法中，自底向上（递推）：从小范围递推计算到大范围，可以看到装第一个和第五个物品价值是最高的，这时候V=12了，然后占了6的重量了，只能装物品2了，价值15，第二个问题是部分背包，部分背包的时候计算每个物品单位重量价值多少，单位重量v={31.55/60.81.5},可以看到125的单位价值最高，选择125后背包重量还只有8，还有2个重量可以选择3得等5/3的价值，就是1.67，所以第三问为16.67  
再来看复杂度，都没有进行指数级别的运算，问题1只需要找n个物品与价值W相乘，问题3计算单位物品价值然后考虑背包大小就可以了

**试题答案**

（62）C（63）A（64）D（65）B

# 试题26(2015年下半年试题57)

对于一个长度为n(n>1)且元素互异的序列，每其所有元素依次通过一个初始为空的栈后，再通过一个初始为空的队列。假设队列和栈的容量都足够大，且只要栈非空就可以进行出栈操作，只要队列非空就可以进行出队操作，那么以下叙述中，正确的是（）。

（57）A．出队序列和出栈序一定互为逆序  
B.出队序列和出栈序列一定相同  
C.入栈序列与入队序列一定相同  
D.入栈序列与入队序列一定互为逆序

**试题分析**

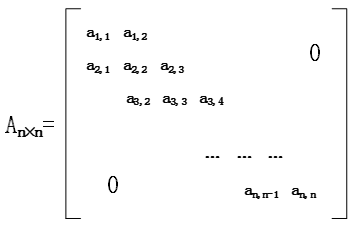
从题目的描述来看，出栈之后，直接入队，然后出队。所以：入队序列=出栈序列，又因为出队序列=入队序列。所以出队序列和出栈序列一定相同。

**试题答案**

（57）B

# 试题27(2015年下半年试题58)

设某n阶三对角矩阵Anxn的示意图如下图所示。若将该三对角矩阵的非零元素按行存储在一维数组B[k]（1≤k≤3\*n-2）中，则k与i、j的对应关系是（）。



（58）A．k=2i+j-2  
B.k=2i-j+2  
C.k=3i+j-1  
D.K=3i-j+2

**试题分析**

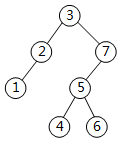
该题最简单的解题思路是代入法。当i=1,j=1时，k=1。  
选项A：k=2i+j-2=2+1-2=1；  
选项B：k=2i-j+2=2-1+2=3；  
选项C：k=3i+j-1=3+1-1=3；  
选项D：k=3i-j+2=3+1+2=4。  
此时可以除排B，C，D，直接选A。若用一个例子，不能排除所有错误选项，则而举一个例子来进行代入，排除更多错误选项。

**试题答案**

（58）A

# 试题28(2015年下半年试题59)

对于非空的二叉树，设D代表根结点，L代表根结点的左子树R代表根结点的右子树。若对下图所示的二叉树进行遍历后的结点序列为7654321，则遍历方式是（）。



（59）A．LRD  
B.DRL  
C.RLD  
D.RDL

**试题分析**

该题突破了常规的遍历树的方式，采用了新的遍历方式。但是做题进行判断时还是比较容易的，因为先根（包括根左右与根右左）的遍历，则根结点3会是第1个访问的结点；后根（左右根与根右左）的遍历，则根结点3会是最后1个访问的结点。给出的序列中3既不在第1个位置，也不在最后1个位置，所以先根后根都可除排，而A、B、C三个选项中，A与C是后根，B选项是先根，都可排除，只能选D。D是右根左的访问方式，与结点序列完全吻合。

**试题答案**

（59）D

# 试题29(2015年下半年试题60)

在55个互异元素构成的有序表A[1..55]中进行折半查找（或二分查找，向下取整）。若需要找的元素等于A[19]，则在查找过程中参与比较的元素依次为（）、A[19]。  
（60）A．A[28]、A[30]、A[15]、A[20]  
B.A[28]、A[14]、A[21]、A[17]  
C.A[28]、A[15]、A[22]、A[18]  
D.A[28]、A[18]、A[22]、A[20]

**试题分析**

折半查找时，下标计算过程为（注：key的值与A[19]相同）：  
1、mid=[(1+55)/2]=28，把A[28]与key的值比较后，缩小查找范围为：A[1]至A[27]；  
2、mid=[(1+27)/2]=14，把A[14]与key的值比较后，缩小查找范围为：A[15]至A[27]；  
3、mid=[(15+27)/2]=21，把A[21]与key的值比较后，缩小查找范围为：A[15]至A[20]；  
4、mid=[(15+20)/2]=17，把A[17]与key的值比较后，缩小查找范围为：A[18]至A[20]；  
5、mid=[(18+20)/2]=19，把A[19]与key的值比较后，发现值相等，找到目标。

**试题答案**

（60）B

# 试题30(2015年下半年试题61)

设一个包含n个顶点、e条弧的简单有向图采用邻接矩阵存储结构（即矩阵元素A[i][j]团等于1或0，分别表示顶点i与顶点j之间有弧或无弧）,则该矩阵购非零元素数目为（）。

（61）A．e  
B.2e  
C.n-e  
D.n+e

**试题分析**

用邻接矩阵存储有向图，图中每一条弧对应矩阵一个非零元素，题目中提到一共有e条弧，所以一共e个非零元素。

**试题答案**

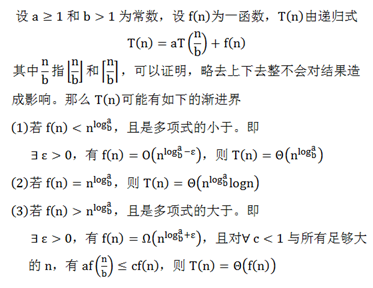
（61）A

# 试题31(2015年下半年试题62-63)

已知算法A的运行时间函数为T(n)=8T(n/2)+n2，其中n表示问题的规模，则该算法的时间复杂度为（）。另已知算法B的运行时间函数为T(n)=XT(n/4)+n2，其中n表示问题的规模。对充分大的n，若要算法B比算法A快，则X的最大值为（）。

（62）A．θ(n)  
B.θ(nlgn)  
C.θ(n2)  
D.θ(n3)  
  
（63）A．15  
B.17  
C.63  
D.65

**试题分析**

本题需要用到特定形式的递归式分析法：  
  
在本题中，a=8,b=2，故符合（1）的情况。  
时间复杂度为：O（n3）。  
a=16,b=4

**试题答案**

（62）D（63）C

# 试题32(2015年下半年试题64-65)

在某应用中，需要先排序一组大规模的记录，其关键字为整数。若这组记录的关键字基本上有序，则适宜采用（）排序算法。若这组记录的关键字的取值均在0到9之间（含），则适宜采用（）排序算法。

（64）A．插入  
B.归并  
C.快速  
D.计数  
  
（65）A．插入  
B.归并  
C.快速  
D.计数

**试题分析**

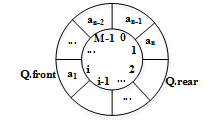
插入排序中的希尔排序的基本思想是：先将整个待排序的记录序列分割成为若干子序列分别进行直接插入排序，待整个序列中的记录“基本有序”时，再对全体记录进行依次直接插入排序。所以当数列基本有序时，采用插入排序算法是比较合适的。  
计数排序是一个非基于比较的排序算法，该算法于1954年由HaroldH.Seward提出。它的优势在于在对一定范围内的整数排序时，它的复杂度为Ο(n+k)（其中k是整数的范围），快于任何比较排序算法。

**试题答案**

（64）A（65）D

# 试题33(2015年上半年试题57)

设某循环队列Q的定义中有front和rear两个域变量，其中，front指示队头元素的位置，rear指示队尾元素之后的位置，如下图所示。若该队列的容量为M，则其长度为（）。



（57）A．(Q.rear-Q.front+1)  
B.(Q.rcar-Q.front+M)  
C.(Q.rear-Q.front+1)%M  
D.(Q.rear-Q.front+M)%M

**试题分析**

本题考查数据结构基础知识。  
根据图示，可以计算得知（Q.rear-Q.front+M）%M为队列中的元素个数（即队列长度）。

**试题答案**

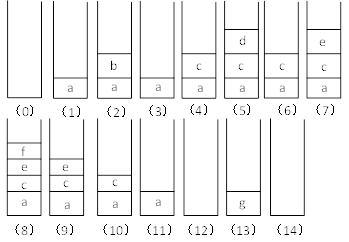
（57）D

# 试题34(2015年上半年试题58)

设栈S和队列Q的初始状态为空，元素abcdef依次进入栈S。要求每个元素出栈后立即进入队列Q，若7个元素出队列的顺序为bdfecag，则栈S的容量最小应该是（）。  
（58）A．5  
B.4  
C.3  
D.2

**试题分析**

本题考查数据结构基础知识。  
根据队列的特点，元素出队的顺序与入队的顺序相同，因此，可知这7个元素的出栈顺序为bdfecag。对于入栈序列abcdefg，得出出栈序列bdfecag的操作过程为：push（a入）、push（b入）、pop（b出）、push（c入）、push（d入）、pop（d出）、push（e入）、push（f入）、pop（f出）、pop（e出）、pop（c出）、pop（a出）、push（g入）、pop（g出），如下图所示，从中可知栈S中元素最多时为4.因此，S的容量最小为4.



**试题答案**

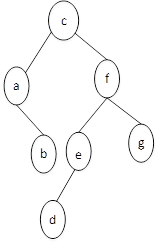
（58）B

# 试题35(2015年上半年试题59)

某二叉树的先序遍历序列为cabfedg,中序遍历序列为abcdefg,则该二叉树是（）。  
（59）A．完全二叉树  
B.最优二叉树  
C.平衡二叉树  
D.满二叉树

**试题分析**

本题考查数据结构基础知识。  
根据题中所给的遍历序列，可知其对应的二叉树如下图所示。



**试题答案**

（59）C

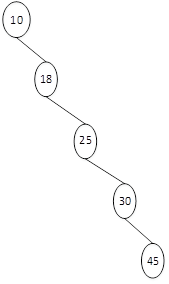
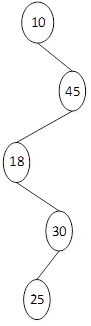
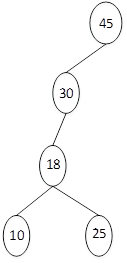
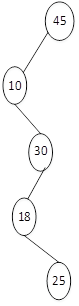
# 试题36(2015年上半年试题60)

对某有序顺序表进行折半查找时，（）不可能构成查找过程中关键字的比较序列。  
（60）A．45,10,30,18,25  
B.45,30,18,25,10  
C.10,45,18,30,25  
D.10,18,25,30,45

**试题分析**

本题考查数据结构基础知识。

进行折半查找时，首先与表中间位置上的元素进行比较，若待查找的元素大于中间元素，则接下来在后半区(是比中间元素更大者组成的杳序子表)进行折半查找，否则在前半区(是比中间元素更小者组成的有序子表)进行折半查找。二分查找过程可用二分查找判定树来描述，即大于中间元素时走右分支，小于中间元素时走左分支，等于时查找成功结束。

四个选项的比较序列在二分查找判定树上的查找路径表示为：  


选项A选项B选项C选项D

显然，选项B是不可能的查找路径。

**试题答案**

（60）B

# 试题37(2015年上半年试题61)

用某排序方法对一元素序列进行非递减排序时，若该方法可保证在排序前后排序码相同者的相对位置不变，则称该排序方法是稳定的。简单选择排序法排序方法是不稳定的，（）可以说明这个性质。  
（61）A．214821\*6317  
B.172121\*4863  
C.63214821\*17  
D.21\*17486321

**试题分析**

本题考查数据结构基础知识。  
对选项A进行简单选择排序时，第一趟需交换17和21，导致21与21\*的相对位置发生变化，最后的非递减序列为1721\*214863，说明简单选择排序是不稳定的排序方法。

**试题答案**

（61）A

# 试题38(2015年上半年试题62-63)

优先队列通常采用（）数据结构实现，向优先队列中插入—个元素的时间复杂度为（）。  
（62）A．堆  
B.栈  
C.队列  
D.线性表  
  
（63）A．Θ（n）  
B.Θ(1)  
C.Θ(lgn)  
D.Θ(n2)

**试题分析**

普通的队列是一种先进先出的数据结构，元素在队列尾追加，而从队列头删除。在优先队列中，元素被赋予优先级。当访问元素时，具有最高优先级的元素最先删除。优先队列具有最高级先出（largest-in，first-out）的行为特征。优先队列一般采用二叉堆数据结构实现，由于是二叉堆，所以插入和删除一个元素的时间复杂度均为O(lgn)。

**试题答案**

（62）A（63）C

# 试题39(2015年上半年试题64-65)

在n个数的数组中确定其第i(1≤i≤n)小的数时，可以采用快速排序算法中的划分思想,对n个元素划分，先确定第k小的数，根据i和k的大小关系,进一步处理，最终得到第i小的数。划分过程中，最佳的基准元素选择的方法是选择待划分数组的（）元素。此时，算法在最坏情况下的时间复杂度为（不考虑所有元素均相等的情况）（）。  
（64）A．第一个  
B.最后一个  
C.中位数  
D.随机一个  
  
（65）A．Θ（n）  
B.Θ（lgn）  
C.Θ（nlgn）  
D.Θ（n2）

**试题分析**

本题考查算法设计与分析的相关知识。

中位数的含义：将一组数据按照由小到大(或由大到小)的顺序排列，如果数据的个数是奇数，则处于中间位置的数就是这组数据的中位数；如果数据的个数是偶数，则中间两个数据的平均数就是这组数据的中位数。根据题干的描述，选择的基准元素将数组分得越均匀越好，因此中位数是最佳选择。

对于该问题，若每次都是选择中位数作为基准元素，则时间复杂度的递归式为：

T(n)=T(n/2)+cn

求解该递归式，得到T(n)=Θ(n)。

**试题答案**

（64）C（65）D

# 试题40(2014年下半年试题57)

对于线性表，相对于顺序存储，采用链表存储的缺点是（）。  
（57）A．数据元素之间的关系需要占用存储空间，导致存储密度不高  
B.表中结点必须占用地址连续的存储单元，存储密度不高  
C.插入新元素时需要遍历整个链表，运算的时间效率不高  
D.删除元素时需要遍历整个链表，运算的时间效率不高

**试题分析**

链表最大的优点是没有大小限制也就是说它是动态的。。你可以任意添加大小通过结构体你可以将很多相关的数据放到一起。。但是因为链表在内存里存放是不连续的。所以你不能快速的查找和修改。链表存储的缺点为数据元素之间的关系需要占用存储空间，导致存储密度不高。

**试题答案**

（57）A

# 试题41(2014年下半年试题58)

若一个栈初始为空，其输入序列是1，2，3，…，n-1，n，其输出序列的第一个元素为k(1≤k≤「n/2」)，则输出序列的最后一个元素是（）。  
（58）A．值为n的元素  
B.值为1的元素  
C.值为n-k的元素  
D.不确定的

**试题分析**

本题考查数据结构基础知识。  
以n等于4举例说明。输入序列为1234，输出序列的第一个元素可以为1或2。若为1，则输出序列可能为1234、1243、1342、1324、1432；若为2，则输出序列为2134、2143、2314、2341、2431。

**试题答案**

（58）D

# 试题42(2014年下半年试题59)

某个二叉查找树（即二叉排序树）中进行查找时，效率最差的情形是该二叉查找树是（）。  
（59）A．完全二叉树  
B.平衡二叉树  
C.单枝树  
D.满二叉树

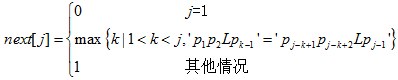
**试题分析**

单枝树时该二叉查找树效率最低。

**试题答案**

（59）C

# 试题43(2014年下半年试题60)

在字符串的KMP模式匹配算法中，需先求解模式串的next函数值，其定义如下式所示，j表示模式串中字符的序号（从1开始）。若模式串p为“abaac”，则其next函数值为（）。  


（60）A．01234  
B.01122  
C.01211  
D.01111

**试题分析**

本题考查字符串的模式匹配运算知识。  
KMP是进行字符串模式匹配运算效率较高的算法。根据对next函数的定义，模式串前两个字符的next值为0、1。对于第3个字符"a"，其在模式串中的前缀为"ab"，从该子串找不出前缀和后缀相同的部分，因此，根据定义，该位置字符的next值为1。  
对于第4个字符"a"，其在模式串中的前缀为"aba"，该子串只有长度为1的前缀"a"和后缀"a"相同，根据定义，该位置字符的next值为2。  
对于第5个字符"a"，其在模式串中的前缀为"abaa0"，该子串只有长度为1的前缀"a"和后缀"a"相同，根据定义，该位置字符的next值为2。  
综上可得，模式串"abaac"的next函数值为01122。

**试题答案**

（60）B

# 试题44(2014年下半年试题61-62)

快速排序算法在排序过程中，在待排序数组中确定一个元素为基准元素，根据基准元素把待排序数组划分成两个部分，前面一部分元素值小于等于基准元素，而后面一部分元素值大于基准元素。然后再分别对前后两个部分进一步进行划分。根据上述描述，快速排序算法采用了（）算法设计策略。日知确定基准元素操作的时间复杂度为Θ(n)，则快速排序算法的最好和最坏情况下的时间复杂度为（）。  
（61）A．分治  
B.动态规划  
C.贪心  
D.回溯  
  
（62）A．Θ(n)和Θ(nlgn)  
B.Θ(n)和Θ(n2)  
C.Θ(nlgn)和Θ(nlgn)  
D.Θ(nlgn)和Θ(n2)

**试题分析**



**试题答案**

（61）A（62）D

# 试题45(2014年下半年试题63)

对一待排序序列分别进行直接插入排序和简单选择排序，若待排序序列中有两个元素的值相同，则（）保证这两个元素在排序前后的相对位置不变。  
（63）A．直接插入排序和简单选择排序都可以  
B.直接插入排序和简单选择排序都不能  
C.只有直接插入排序可以  
D.只有简单选择排序可以

**试题分析**

直接插入排序才是稳定的排序算法。

**试题答案**

（63）C

# 试题46(2014年下半年试题64-65)

已知一个文件中出现的各字符及其对应的频率如下表所示。若采用定长编码，则该文件中字符的码长应为（）。若采用Huffman编码，则字符序列“face”的编码应为（）。  
IMG_293

（64）A．2  
B.3  
C.4  
D.5  
  
（65）A．110001001101  
B.001110110011  
C.101000010100  
D.010111101011

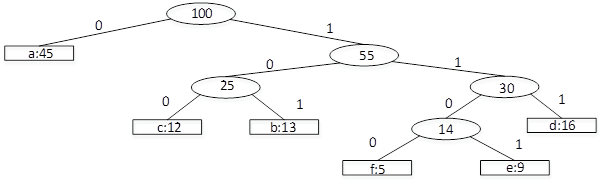
**试题分析**

本题考查Huffman编码的相关知识。

字符在计算机中是用二进制表示的，每个字符用不同的二进制编码来表示。码的长度影响存储空间和传输效率。若是定长编码方法，用2位码长，只能表示4个字符，即00、01、10和11；若用3位码长，则可以表示8个字符，即000、001、010、011、100、101、110、111。对于题中给出的例子，一共有6个字符，因此采用3位码长的编码可以表示这些字符。

Huffman编码是一种最优的不定长编码方法，可以有效的压缩数据。要使用Huffman编码，除了知道文件中出现的字符之外，还需要知道每个字符出现的频率。下图（a）是题干中给出对应的编码树，可以看到，每个字符及其对应编码为图（b），因此字符序列"face"的编码应为110001001101，即65选择A。

（a）



（b）

|  |  |
| --- | --- |
| 字符 | 编码 |
| a | 0 |
| b | 101 |
| c | 100 |
| d | 111 |
| e | 1101 |
| f | 1100 |

**试题答案**

（64）B（65）A

# 试题47(2014年上半年试题57)

若对线性表的最常用操作是访问任意指定序号的元素，并在表尾加入和删除元素，则适宜采用（）存储。

（57）A．顺序表  
B.单链表  
C.双向链表  
D.哈希表

**试题分析**

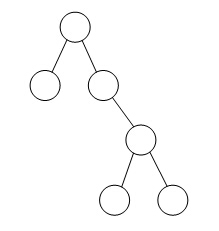
考查线性表的特性  
题意：对线性表的最常用操作是访问任意指定序号的元素，并在表尾加入和删除元素。  
要访问任意指定序号的元素，最快速的访问方式自然是采用数组存储（顺序表），但采用数组存储时，在数组中间位置或者头部插入、删除元素效率太低，需要移动大量元素，而题意中在表尾加入和删除元素，则正好消除了这种缺陷。

**试题答案**

（57）A

# 试题48(2014年上半年试题58-59)

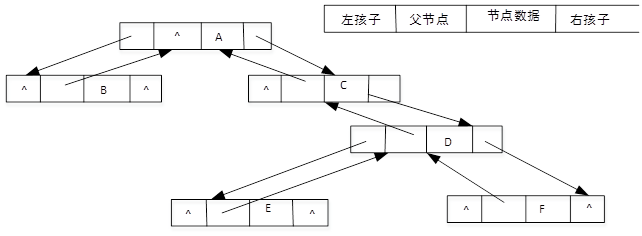
二叉树如右图所示,若进行顺序存储（即用一维数组元素存储该二叉树中的结点且通过下标反映结点间的关系，例如，对于下标为i的结点，其左孩子的下标为2i、右孩子的下标为2i+1），则该数组的大小至少为（）；若釆用三叉链表存储该二叉树（各个结点包括结点的数据、父结点指针、左孩子指针、右孩子指针），则该链表的所有结点中空指针的数目为（）。



（58）A．6  
B.10  
C.12  
D.15  
  
（59）A．6  
B.8  
C.12  
D.14

**试题分析**

用一维数组元素存储该二叉树中的结点且通过下标反映结点间的关系，实际上存储的是这棵二叉树对应的完全二叉树，因此需要的存储空间为2n-1=15(n为二叉树层数)。如下图所示：  
IMG_296  
釆用三叉链表存储该二叉树（各个结点包括结点的数据、父结点指针、左孩子指针、右孩子指针）；如下图所示：

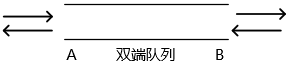


空指针数量为8。

**试题答案**

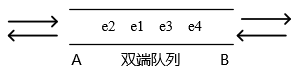
（58）D（59）B

# 试题49(2014年上半年试题60)

某双端队列如下所示，要求元素进出队列必须在同一端口，即从A端进入的元素必须从A端出、从B端进入的元素必须从B端出，则对于4个元素的序列e1、e2、e3、e4，若要求从前2个元素（e1、e2）从A端口按次序全部进入队列，后两个元素(e3、e4)从B端口按次序全部进入队列，则可能得到的出队序列是（）。  


（60）A．e1、e2、e3、e4  
B.e2、e3、e4、e1  
C.e3、e4、e1、e2  
D.e4、e3、e2、e1

**试题分析**

e1、e2从A端口进入，e3、e4从B端口进入，如下图所示：  
  
根据题意：从A端进入的元素必须从A端出、从B端进入的元素必须从B端出；则出队顺序中e2在e1前面，e4在e3前面。  
只有答案D满足。

**试题答案**

（60）D

# 试题50(2014年上半年试题61)

实现二分查找（折半查找）时，要求查找表（）。

（61）A．顺序存储，关键码无序排列  
B.顺序存储，关键码有序排列  
C.双向链表存储，关键码无序排列  
D.双向链表存储，关键码有序排列

**试题分析**

二分查找又称折半查找，优点是比较次数少，查找速度快，平均性能好；其缺点是要求待查表为有序表，且插入删除困难。因此，折半查找方法适用于不经常变动而查找频繁的有序列表。  
算法要求：①必须采用顺序存储结构②必须按关键字大小有序排列。

**试题答案**

（61）B

# 试题51(2014年上半年试题62-63)

在某个算法时间复杂度递归式T(n)=T(n-1)+n，其中n为问题的规模，则该算法的渐进时间复杂度为（），若问题的规模增加了16倍，则运行时间增加（）倍。

（62）A．Θ(n)  
B.Θ(nlgn)  
C.Θ(n2)  
D.Θ(n2lgn)  
  
（63）A．16  
B.64  
C.256  
D.1024

**试题分析**

由于递归式为：T(n)=T(n-1)+n。我们可以把一个规模为n的时间复杂度算出来。

分析过程为：

T(n)=T(n-1)+n；

T(n-1)=T(n-2)+n-1；

T(n-2)=T(n-3)+n-2；

....

T(n)=1+2+..+n-1+n。

这是一个典型的等差数列。用数列求和公式有：((1+n)\*n)/2。这样就求得时间复杂度为：Θ(n2)。

后面一问则有：

当问题规模为n时，时间复杂度Θ(n2)。

当x=16n时，时间复杂度Θ(x2)=Θ((16n)2)=Θ(256n2)。

**试题答案**

（62）C（63）C

# 试题52(2014年上半年试题64-65)

Prim算法和Kruscal算法都是无向连通网的最小生成树的算法，Prim算法从一个顶点开始，每次从剩余的顶点加入一个顶点，该顶点与当前生成树中的顶占的连边权重最小，直到得到最小生成树开始，Kruscal算法从权重最小的边开始，每次从不在当前的生成树顶点之间的边中选择权重最小的边加入，直到得到一颗最小生成树，这两个算法都采用了（）设计策略，且（）。

（64）A．分治  
B.贪心  
C.动态规划  
D.回溯  
  
（65）A．若网较稠密，则Prim算法更好  
B.两个算法得到的最小生成树是一样的  
C.Prim算法比Kruscal算法效率更高  
D.Kruscal算法比Prim算法效率更高

**试题分析**

本题考查算法设计与分析的基础知识。  
Prim算法从扩展顶点开始，每次总是"贪心的"选择与当前顶点集合中距离最短的顶点，而Kruscal算法从扩展边开始，每次总是"贪心的"选择剩余的边中最小权重的边，因此两个算法都是基于贪心策略进行的。  
Prim算法的时间复杂度为O(n2)，其中n为图的顶点数，该算法的计算时间与图中的边数无关，因此该算法适合于求边稠密的图的最小生成树；Kruscal算法的时间复杂度为O(mlgm)，其中m为图的边数，该算法的计算时间与图中的顶点数无关，因此该算法适合于求边稀疏的图的最小生成树。当图稠密时，用Prim算法效率更高。但若事先没有关于图的拓扑特征信息时，无法判断两者的优劣。由于一个图的最小生成树可能有多棵，因此不能保证用这两种算法得到的是同一棵最小生成树。

**试题答案**

（64）B（65）A

# 试题53(2013年下半年试题57)

以下关于线性表存储结构的叙述，正确的是（）。

（57）A．线性表采用顺序存储结构时，访问表中任意一个指定序号元素的时间复杂度为常量级  
B.线性表采用顺序存储结构时，在表中任意位置插入新元素的运算时间复杂度为常量级  
C.线性表采用链式存储结构时，访问表中任意一个指定序号元素的时间复杂度为常量级  
D.线性表采用链式存储结构时，在表中任意位置插入新元素的运算时间复杂度为常量级

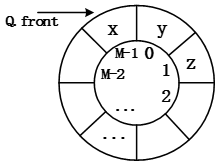
**试题分析**

线性表采用顺序存储结构时，访问表中任意一个指定序号元素的时间复杂度为常量级，因为顺序存储结构访问元素时，能直接定位元素，这样，操作的时间复杂度为O（1）。

**试题答案**

（57）A

# 试题54(2013年下半年试题58)

设循环队列Q的定义中有front和size两个域变量，其中front表示队头元素的指针，size表示队列的长度，如下图所示（队列长度为3，队头元素为x，队尾元素为z）。设队列的存储空间容量为M，则队尾元素的指针为（）。  


（58）A．（Q.front+Q.size-1）  
B.（Q.front+Q.size-1+M）%M  
C.（Q.front-Q.size）  
D.（Q.front-Q.size+M）%M

**试题分析**

本题考查循环队列队尾指针的计算方法。  
从图示可以看出，要得到z的值可进行Q.front+Q.size-1操作，但在此不容忽视的一个问题是，循环队列在进行了多次入队出队操作之后，Q.front+Q.size-1有可能大于M，如Q.front指向M-1空间时，Q.front+Q.size-1=M+1，这已超出队列长度，所以需要让其与M进行求模操作，修正位置号。

**试题答案**

（58）B

# 试题55(2013年下半年试题59)

在一个有向图G的拓扑序列中，顶点Vi排列在Vj之前，说明图G中（）。

（59）A．一定存在弧<vi,vj>  
B.一定存在弧<vj,vi>  
C.可能存在vi到vj的路径，而不可能存在vj到vi的路径  
D.可能存在vj到vi的路径，而不可能存在vi到vj的路径

**试题分析**

拓扑序列是拓扑排序的产出物。对一个有向无环图G进行拓扑排序，是将G中所有顶点排成一个线性序列，使得图中任意一对顶点u和v，若边(u,v)∈E(G)，则u在线性序列中出现在v之前。由此可见，如果Vi排列在Vj之前，说明可能存在vi到vj的路径，而不可能存在vj到vi的路径。

**试题答案**

（59）C

# 试题56(2013年下半年试题60)

以下关于哈夫曼树的叙述，正确的是（）。

（60）A．哈夫曼树一定是满二叉树，其每层结点数都达到最大值  
B.哈夫曼树一定是平衡二叉树，其每个结点左右子树的高度差为-1、0或1  
C.哈夫曼树中左孩子结点的权值小于父节点、右孩子节点的权值大于父节点  
D.哈夫曼树中叶子节点的权值越小则距离树根越远、叶子结点的权值越大则距离树根越近

**试题分析**

给定n个权值作为n个叶子结点，构造一棵二叉树，若带权路径长度达到最小，称这样的二叉树为最优二叉树，也称为哈夫曼树。哈夫曼树是带权路径长度最短的树，权值较大的结点离根较近。所以D选项的说法正确。

**试题答案**

（60）D

# 试题57(2013年下半年试题61)

某哈希表（散列表）的长度为n，改散列函数为H（Key）=Keymodp,采用线性探测法解决冲突。以下关于P值的叙述中，正确的是（）。

（61）A．p的值一般为不大于n且最接近n的质数  
B.p的值一般为大于n的任意整数  
C.p的值必须为小于n的合数  
D.p的值必须等于n

**试题分析**

在采用散列表进行数据存储时，散列函数中p的取值是非常重要的，因为该取值直接影响冲突发生率，所以p的值一般会取接近于元素个数n但是要小于n的质数。  
例如你n取20，那么P最好是19。

**试题答案**

（61）A

# 试题58(2013年下半年试题62-63)

对n个基本有序的整数进行排序，若采用插入排序算法，则时间和空间复杂度分别为（）；若采用快速排序算法，则时间和空间复杂度分别为（）。

（62）A．O（n2）和O（n）  
B.O（n）和O（n）  
C.O（n2）和O（1）  
D.O（n）和O（1）  
  
（63）A．O（n2）和O（n）  
B.O（nlgn）和O（n）  
C.O（n2）和O（1）  
D.O（nlgn）和O（1）

**试题分析**

本题考查算法分析的基础知识。排序和查找是基本的计算问题。存在很多相关的算法，不同的算法适用于不同的场合。不同的数据输入特点相同的算法也有不同的计算时间。若数据基本有序，对插入排序算法而言，则可以在近似线性时间内完成排序。即  
O(n)；而对于快速排序而言，则是其最坏情况，需要二次时间才能完成排序，即O(n2)。两个算法在排序时仅需要一个额外的存储空间，即空间复杂度为常数O(1)。

**试题答案**

（62）D（63）C

# 试题59(2013年下半年试题64-65)

在求解某问题时，经过分析发现该问题具有最优子结构性质，求解过程中子问题被重复求解，则采用（）算法设计策略；若定义问题的解空间，以深度优先的方式搜索解空间，则采用（）算法设计策略。

（64）A．分治  
B.动态规划  
C.贪心  
D.回溯  
  
（65）A．动态规划  
B.贪心  
C.回溯  
D.分支限界

**试题分析**

分治法的设计思想是将一个难以直接解决的大问题分解成一些规模较少的相同问题以便各个击破，分而治之。  
动态规划法与分治法类似，其基本思想也是将待求解问题分解成若干个子问题，先求解子问题，然后从这些子问题的解得到原问题的解。与分治法不同的是，适合于用动态规划法求解的问题，经分解得到的子问题往往不是独立的。若用分治法来解这类问题，则相同的子问题会被求解多次，以至于最后解决原问题需要耗费指数级时间。  
贪心法经常用于解决最优化问题，但他的最优往往是从局部最优来考虑的，每一步都选最优的方案，但这种方案不一定能得到整体上的最优解。  
回溯法是一种既带有系统性又带有跳跃性的搜索算法。它在包含问题的所有解的解空间树中，按照深度优先的策略，从根节点出发搜索解空间树。  
题目描述中提到，需要解决的问题具有最优子结构性质，且求解过程中子问题被重复求解，这种情况下如果采用分治法，效率会很低，所以应采用动态规划法。而“以深度优先的方式搜索解空间”则明显是在采用回溯法。

**试题答案**

（64）B（65）C

# 试题60(2013年上半年试题51)

采用顺序表和单链表存储长度为n的线性序列，根据序号查找元素，其时间复杂度分别为（）。

（51）A．O(1)O(1)  
B.O(1)O(N)  
C.O(N)O(1)  
D.O(N)O(N)

**试题分析**

顺序表是在计算机内存中以数组的形式保存的线性表，是指用一组地址连续的存储单元依次存储数据元素的线性结构。顺序存储结构的主要优点是节省存储空间，因为分配给数据的存储单元全用存放结点的数据，结点之间的逻辑关系没有占用额外的存储空间。采用这种方法时，可实现对结点的随机存取，即每一个结点对应一个序号，由该序号可以直接计算出来结点的存储地址。  
链表是一种物理存储单元上非连续、非顺序的存储结构，数据元素的逻辑顺序是通过链表中的指针链接次序实现的。链表由一系列结点（链表中每一个元素称为结点）组成，结点可以在运行时动态生成。每个结点包括两个部分：一个是存储数据元素的数据域，另一个是存储下一个结点地址的指针域。  
链表(Linkedlist）是一种常见的基础数据结构，是一种线性表，但是并不会按线性的顺序存储数据，而是在每一个节点里存到下一个节点的指针(Pointer）。由于不必按顺序存储，链表在插入的时候可以达到O⑴的复杂度，比另一种线性表：顺序表快得多，但是查找一个节点或者访问特定编号的节点则需要O(n）的时间，而顺序表相应的时间复杂度分别是O(logn）和O⑴。

**试题答案**

（51）B

# 试题61(2013年上半年试题52)

设元素序列a，b，c，d，e，f经过初始为空的栈S后，得到出栈序列cedfba，则栈S的最小容量为（）。

（52）A．3  
B.4  
C.5  
D.6

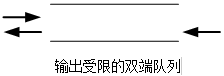
**试题分析**

此题考查栈的用法，根据题中出栈的顺序，当元素c出栈后，栈中有元素a、b，当元素e出栈之前，栈中由元素a、b、d、e，此时栈中的元素达到最多。因此栈S中最小容量为4。

**试题答案**

（52）B

# 试题62(2013年上半年试题53)

输出受限的双端队列是指元素可以从队列的两端输入，但只能从队列的一端输出，如下图所示，若有e1，e2，e3，e4依次进入输出受限的双端队列，则得不到输出序列（）。  


（53）A．e4，e3，e2，e1  
B.e4，e2，e1，e3  
C.e4，e3，e1，e2  
D.e4，e2，e3，e1

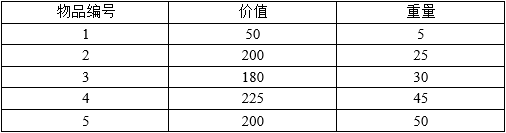
**试题分析**

此题考查队列的用法，题中给出的受限双端队列，两端都可以进，一端出。假设分A和B端，B端可以进出，由D选项出序列，可以看出e1、e2、e3按顺序从A端进入，而e4从B端进入，当e4从B端出来后，无法将后面的e2出队列。

**试题答案**

（53）D

# 试题63(2013年上半年试题60-61)

考虑下述背包问题的实例。有5件物品，背包容量为100，每件物品的价值和重量如下表所示，并已经按照物品的单位重量价值从大到小排好序，根据物品单位重量价值大优先的策略装入背包中，则采用了（）设计策略。考虑0/1背包问题（每件物品或者全部放入或者全部不装入背包）和部分背包问题（物品可以部分装入背包），求解该实例，得到的最大价值分别为（）。  


（60）A．分治  
B.贪心  
C.动态规划  
D.回溯  
  
（61）A．605和630  
B.605和605  
C.430和630  
D.630和430

**试题分析**

本题考查贪心算法和背包问题的知识点。  
贪心算法（又称贪婪算法）是指，在对问题求解时，总是做出在当前看来是最好的选择。也就是说，不从整体最优上加以考虑，他所做出的仅是在某种意义上的局部最优解。贪心算法不是对所有问题都能得到整体最优解，但对范围相当广泛的许多问题他能产生整体最优解或者是整体最优解的近似解。  
0/1背包考虑该问题时，只能放入1、2、3号物品，故总价值为430，采用部分背包问题可以将物品拆分，故放1、2、3号物品后还可以放入部分4号物品，故总容量为630。

**试题答案**

（60）B（61）C

# 试题64(2013年上半年试题62-63)

给定n个整数构成的数组A={a1,a2,…,an}和整数x,判断A中是否存在两个元素ai和aj，使得ai+aj=x。为了求解该问题，首先用归并排序算法对数组A进行从小到大排序；然后判断是否存在ai+aj=x，具体如下列伪代码所示，则求解该问题时排序算法应用了（）算法设计策略，整个算法的时间复杂度为（）  
i=1;j=n  
whilei<j  
ifai+aj=xreturntrue  
elseifai+aj>x  
j--;  
else  
i++;  
returnfalse;

（62）A．分治  
B.贪心  
C.动态规划  
D.回溯  
  
（63）A．O(n)  
B.O(nlgn)  
C.O(n2)  
D.O(nlg2n)

**试题分析**

分治算法的基本思想是将一个规模为N的问题分解为K个规模较小的子问题，这些子问题相互独立且与原问题性质相同。求出子问题的解，就可得到原问题的解。

**试题答案**

（62）A（63）B

# 试题65(2013年上半年试题64)

一个高度为h的满二叉树的结点总数为2h-1，从根结点开始，自上而下、同层次结点从左至右，对结点按照顺序依次编号，即根结点编号为1，其左、右孩子结点编号分别为2和3，再下一层从左到右的编号为4、5、6、7，依此类推。那么，在一棵满二叉树中，对于编号为m和n的两个结点，若n=2m+1,则（）。

（64）A．m是n的左孩子  
B.m是n的右孩子  
C.n是m的左孩子  
D.n是m的右孩子

**试题分析**

由于该二叉树为满二叉树，除最后一层无任何子节点外，每一层上的所有结点都有两个子结点（最后一层上的无子结点的结点为叶子结点）。满二叉树的性质可知父结点m和右孩子之间的关系式n=2m+1。

**试题答案**

（64）D

# 试题66(2013年上半年试题65)

以下关于哈希(Hash,散列)查找叙述中，正确的是（）。

（65）A．哈希函数应尽可能复杂些，以消除冲突  
B.构造哈希函数时应尽量使关键字的所有组成部分都能起作用  
C.进行哈希查找时，不再需要与查找表中的元素进行比较  
D.在哈希表中只能添加元素不能删除元素

**试题分析**

哈希表根据设定的哈希函数H(key）和所选中的处理冲突的方法，将一组关键字映象到一个有限的、地址连续的地址集（区间）上并以关键字在地址集中的“象”作为相应记录在表中的存储位置。所以在构造哈希函数使应尽量使关键字的所有组成部分起作用。

**试题答案**

（65）B

# 试题67(2012年下半年试题57)

在字符串的模式匹配过程中，如果模式串的每个字符依次和主事中一个连续的字符序列相等，则称为匹配成功。如果不能在主串中找到与模式串相同的子串，则称为匹配失败。在布鲁特—福斯模式匹配算法（朴素的或基本的模式匹配）中，若主串和模式串的长度分别为n和m（且n远大于m），且恰好在主串末尾的m个字符处匹配成功，则在上述的模式匹配过程中，字符的比较次数最多为（）。

（57）A．n\*m  
B.（n-m+1）\*m  
C.（n-m-1）\*m  
D.（n-m）\*n

**试题分析**

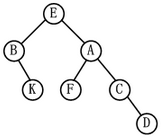
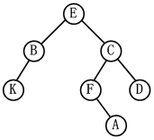
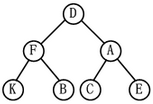
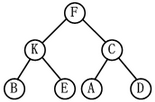
本题主要考查字符串的匹配。在本题的描述中，告诉我们是在主串末尾的m个字符处匹配成功，那么在这之前，从左到右依次匹配了n-m次，且都失败了，最坏的情况，就是每次匹配都是匹配到最后一个字符不符合，因此每次匹配的比较次数就是子串的长度，即m。而匹配成功时，一共也比较了m次。所以字符的比较次数最多为（n-m+1）\*m次。

**试题答案**

（57）B

# 试题68(2012年下半年试题58)

若某二叉树的后序遍历序列为KBFDCAE，中序遍历序列为BKEFACD，则该二叉树为（）。

（58）A．  
B.  
C.  
D.

**试题分析**

本题考查二叉树的遍历。  
二叉树的主要遍历方式有：前序遍历、中序遍历、后序遍历、层次遍历。如果已知中序遍历，并知道前序遍历与后序遍历中的任意一个，便可得到一棵唯一的二叉树。  
具体是怎么做的呢？  
利用的是遍历的特点。中序遍历的顺序是：左、根、右。而后序遍历的顺序是：左、右、根。  
回到题目里面来，从“后序遍历序列为KBFDCAE”，可以得知，二叉树的根结点为：E（此时已经可以排除选项C与选项D了）。继续分析，由“中序遍历序列为BKEFACD”，可以得知，二叉树的左子树包括结点：BK。右子树包括结点：FACD。  
重复上面的步骤，对左子树与左子树看成独立的两棵树进行分析。在后序遍历中，左子树的结点BK的顺序为“KB”，所以B是根结点；右子树的结点FACD的顺序为“FDCA”，所以右子树的根结点为A。当分析到这一步时，已经可以得到本题答案为A。

**试题答案**

（58）A

# 试题69(2012年下半年试题59)

在13个元素构成的有序表M[1．．13]中进行折半查找（向下取整），若找到的元素为M[4]，则被比较的元素依次为（）。

（59）A．M[7]、M[3]、M[5]、M[4]  
B.M[7]、M[5]、M[4]  
C.M[7]、M[6]、M[4]  
D.M[7]、M[4]

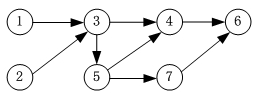
**试题分析**

整个查找的过程为：  
（1+13）/2=7，因此首先与第7元素比较，由于要查找的元素在其前面，因此用（1+7-1）/2=3，然后与第3个元素比较，由于待查找在其后面，因此用（3+1+6）/2=5，因此接下来与第5个元素进行比较，最后再与第4个元素比较，找到了M[4]。

**试题答案**

（59）A

# 试题70(2012年下半年试题60)

拓扑排序是将有向图中所有顶点排成一个线性序列的过程，并且该序列满足：若在AOV网中从顶点Vi到Vj有一条路径，则顶点Vì必然在顶点Vj之前。对于下面所示的有向图，（）是其拓扑序列。  


（60）A．1234576  
B.1235467  
C.2135476  
D.2134567

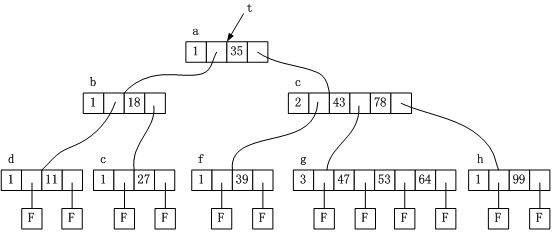
**试题分析**

本题考查数据结构中的拓扑排序。  
拓扑排序通俗一点来讲，其实就是依次遍历没有前驱结点的结点。而某一时刻没有前驱结点的结点有可能存在多个，所以一个图的拓扑排序可能有多个。  
以本题为例，1号结点与2号结点都没有前驱结点，所以拓扑排序的第一个元素可以是1，也可以是2。当1与2都访问完了，便可访问3号结点，3号结点访问完了，便可访问5号结点，访问完5号结点，可访问4号，或是7号结点。  
所以拓扑排序结果为：（12）35（47）6。括号中有多个数字，则代表在这多个数字的顺序可以变化。  
这样，具体的拓扑排序结果为：1235476、1235746、2135476、2135746。

**试题答案**

（60）C

# 试题71(2012年下半年试题61)

下图所示为一棵M阶B-树，M最有可能的值为（）。  


（61）A．1  
B.2  
C.3  
D.4

**试题分析**

本题主要考查B-树的概念。  
一棵m阶的B-树，或者为空树，或为满足下列特性的m叉树：  
（1）树中每个结点至多有m棵子树；  
（2）若根结点不是终端结点，则至少有2棵子树；  
（3）除根结点之外的所有非终端结点至少有[m/2]棵子树；  
（4）所有的非终端结点中包含信息数据（n，P0，K1，P1，K2，P2，...，Kn，Pn），其中：  
Ki(1≤i≤n)是关键字，并且Ki<ki+1(1≤i≤n-1)；  
Pi(0≤i≤n)是指向子树根结点的指针，而且指针Pi-1所指子树中所有结点的关键字均小于关键字Ki(1≤i≤n)，并且均大于关键字Ki-1(2≤i≤n)；第一个指针P0所指子树中所有结点的关键字均小于K1，最后一个指针Pn所指子树中所有结点的关键字均大于Kn；  
n是结点中关键字的个数，有[m/2]-1≤n≤m-1。  
（5）所有的叶子结点都出现在同一层次上，并且不带信息。这些结点实际上并不存在，如果查找进入叶子结点，则说明查找失败。  
从题目给出的图来看，最多一个节点有4棵子树，最少一个节点有2棵子树，因此这个B-树最有可能是一棵4阶的B-树。

**试题答案**

（61）D

# 试题72(2012年下半年试题62-63)

将数组{1，1，2，4，7，5}从小到大排序，若采用（）排序算法，则元素之间需要进行的比较次数最少，共需要进行（）次元素之间的比较。

（62）A．直接插入  
B.归并  
C.堆  
D.快速  
  
（63）A．5  
B.6  
C.7  
D.8

**试题分析**

本题主要考查排序算法。  
本题给出的数组如果采用直接插入排序，那么其排序过程如下：首先1和1比较找到合适的插入位置，然后2和1比较，找到合适的插入位置；然后4和2比较，找到4的合适插入位置，然后7和4比较，找到7的合适插入位置，然后5和7比较，因为5比7小，因此要与4比较，然后就找到了5的合适位置，整个排序过程结束。总的比较次数为1+1+1+1+2=6次。  
归并排序的算法思想是将两个相邻的有序子序列归并为一个有序序列，然后再将新产生的相邻序列进行归并，当只剩下一个有序序列时算法结束。其过程如下：1和1比较，然后归并，2和4比较，然后归并，7和5比较，然后归并，解析来将再将[1，1]和[2，4]归并，用2分别与两个1比较得到[1，1，2，4]，然后再用[1，1，2，4]与[5，7]归并。这时，用5与[1，1，2，4]中每个元素分别比较一次，最后即可得到整个有序序列。总的比较次数为：1+1+1+2+4=9次。  
堆排序的基本思想是先将序列建立堆，然后输出堆顶元素，再将剩下的序列建立堆，然后再输出堆顶元素，依此类推，直到所有元素均输出为止。因此在堆排序过程中，最重要的就是建堆。本题中给出的数组序列就是一个小顶堆，然后输出堆顶，将剩下的部分调整为小顶堆，调整的过程为，首先将最后一个元素5置换到堆顶，然后用5与左孩子结点比较，由于大于左孩子，因此与其置换位置，然后值为5的结点仍然大于其左孩子结点，再置换位置，这样就得到了新的小顶堆，这个过程总共比较2次。后面的排序过程是同样的道理。本题采用堆排序算法总共的比较次数为7次。  
快速排序的基本思想是：  
（1）以某个元素为支点（通常是第一个元素），通过比较关键码和交换记录，将待排序的序列分成两个区间。其中左区间中所有元素的关键字均不大于支点元素的关键字，而右区间中所有元素的关键字均不小于支点元素的关键字。称此过程为一次划分；  
（2）分别对左右区间的待排序序列，再按照以上方法进行划分，直到整个序列按关键字有序为止。  
由于本题给出的例子基本是从小到大有序，不适合采用快速排序发，其总共需要的比较次数为15次。

**试题答案**

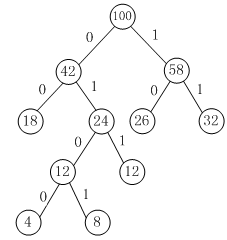
（62）A（63）B

# 试题73(2012年下半年试题64-65)

霍夫曼编码将频繁出现的字符采用短编码，出现频率较低的字符采用长编码。具体的操作过程为：i）以每个字符的出现频率作为关键字构建最小优先级队列；ii）取出关键字最小的两个结点生成子树，根节点的关键字为孩子节点关键字之和，并将根节点插入到最小优先级队列中，直至得到一颗最优编码树。  
霍夫曼编码方案是基于（）策略的。用该方案对包含a到f六个字符的文件进行编码，文件包含100000个字符，每个字符的出现频率（用百分比表示）如下表所示，则与固定长度编码相比，该编码方案节省了（）存储空间。  
IMG_309

（64）A．分治  
B.贪心  
C.动态规划  
D.回溯  
  
（65）A．21%  
B.27%  
C.18%  
D.36%

**试题分析**

根据题目对霍夫曼编码的描述，我们不难知道，每次都是选择当前最小的情况，这符合贪心算法总是找当前看来最优的情况，因此属于贪心策略。  
如果对包含100，000个字符，且这些字符都属于a到f。那么如果采用固定长度的编码，针对于每个字符需要3位来编码（因为有6个不同的字符，至少需要3位才能表示6种不同的变化）。那么对100000个字符编码，其编码长度为300000。  
如果采用霍夫曼编码，那么首先我们就要根据字符出现的频率构造出其霍夫曼树。首先选择出现频率最低的4和8，生成子树，其父节点为12，然后放入出现频率队列中，后面的采用同样的道理，以此类推。构造出的霍夫曼树如下图所示：  
  
由图可以知道，a的编码为00，b的编码为11，c的编码为0100，d的编码为0101，e的编码为011，f的编码为10。因此总的编码长度为（2\*18%+2\*32%+4\*4%+4\*8%+3\*12%+2\*26%）\*100000=236000，因此节省的存储空间大小为300000-236000=64000。因此节省的存储空间为比例为64000/300000=21%。

**试题答案**

（64）B（65）A

# 试题74(2012年上半年试题21)

对于二维数组a[1..N，1..N]中的一个元素a[i，j]（1≤i，j≤N），存储在a[i，j]之前的元素个数（）。

（21）A．与按行存储或按列存储方式无关  
B.在i=j时与按行存储或按列存储方式无关  
C.在按行存储方式下比按列存储方式下要多  
D.在按行存储方式下比按列存储方式下要少

**试题分析**

按行存储即先存储完第一行元素，再开始存储第二行元素，依次类推。按列存储即先存储完第一列元素，再开始存储第二列元素，依次类推。  
因为按行存储和按列存储是两种不同的存储方式，因此在本题中，存储在a[i，j]之前的元素个数与存储方式的选择有关。另外，由于二维数组a的行和列都是n，是相等的，那么在i=j（即行等于列）的情况下，存储元素的个数也是与存储方式无关的。

**试题答案**

（21）B

# 试题75(2012年上半年试题57)

对于一个长度大于1且不存在重复元素的序列，令其所有元素依次通过一个初始为空的队列后，再通过一个初始为空的栈。设队列和栈的容量都足够大，一个序列通过队列（栈）的含义是序列的每个元素都入队列（栈）且出队列（栈）一次且仅一次。对于该序列在上述队列和栈上的操作，正确的叙述是（）。

（57）A．出队序列和出栈序列一定相同  
B.出队序列和出栈序列一定互为逆序  
C.入队序列与出队序列一定相同，入栈序列与出栈序列不一定相同  
D.入栈序列与出栈序列一定互为逆序，入队序列与出队序列不一定互为逆序

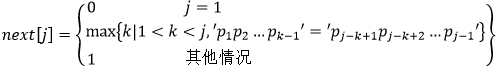
**试题分析**

本题主要考查队列和栈的特性。队列具有先进先出的特点，而栈具有后进先出的特点。因此我们可以知道入队序列与出队序列一定相同，但入栈序列与出栈序列不一定相同。比如a，b，c这样一个序列，那么按照a，b，c的顺序入队列，那么其出队列的次序一定是a，b，c。而按照a，b，c的顺序入栈，那么可能是a入栈后就出栈，然后b入栈又出栈，然后C入栈出栈。也可能是等a，b，c都入栈后再出栈，那么出栈序列就是c，b，a。

**试题答案**

（57）C

# 试题76(2012年上半年试题58)

在字符串的KMP模式匹配锋法中，需要求解模式串p的next函数值，其定义如下所示。若模式串p为“aaabaaa”，则其next函数值为（）。  


（58）A．0123123  
B.0123210  
C.0123432  
D.0123456

**试题分析**

KMP模式匹配算法通俗点说就是一种在一个字符串中定位另一个串的高效算法。其实我们在做这个题目时，也可以不需要知道KMP模式匹配算法，可以根据题目给出的定义式来求解。  
当j=1时，很显然next[1]=0。  
当j=2时，由于1<k<j，因此k无法取到合适值，因此next[2]=1。  
当j=3时，k的取值为2，那么等号左边的‘P1P2…PK-1’字符串就是P1，为字符串中的第一个字符a，而右边就是P2，即字符串中的第二个字符a，显然，它们相等，因此next[3]=k=2。  
当j=4时，k可以取值2或者3，取值为2时，等号左边为第一个字符a，而等号右边为P3，也是字符a，因此相等，但这个时候我们还要判定当k取值为3时，等号左边为第一与第二个字符，即‘aa’，而右边为‘aa’，也相等，因此next[4]=max{2，3}=3。  
当j=5时，k可以取值2、3或者4.，当k取值为2时，等号左边为第一个字符a，而等号右边为P4，也是字符b，不相等。当k取值为3时，等号左边为第一与第二个字符，即‘aa’，而右边为‘ab’，也不相等。当k取值为4时，等号左边为“aaa”，而等号右边为“aab”，也不相等，因此next[5]=1。  
同理我们可以求得当j=6，j=7的结果，本题正确答案选A。

**试题答案**

（58）A

# 试题77(2012年上半年试题59)

若n2、n1、n0分别表示一个二叉树中度为2、度为1和叶子结点的数目（结点的度定义为结点的子树数目），则对于任何一个非空的二叉树，（）。

（59）A．n2一定大于n1  
B.n1一定大于n0  
C.n2一定大于n0  
D.n0一定大于n2

**试题分析**

根据二叉树的性质，我们知道n0=n2+1，因此在一棵二叉树中，叶子结点的数目一定是大于度为2的结点的个数。

**试题答案**

（59）D

# 试题78(2012年上半年试题60)

从存储空间的利用率角度来看，以下关于数据结构中图的存储的叙述，正确的是（）。

（60）A．有向图适合采用邻接矩阵存储，无向图适合采用邻接表存储  
B.无向图适合采用邻接矩阵存储，有向图适合采用邻接表存储  
C.完全图适合采用邻接矩阵存储  
D.完全图适合采用邻接表存储

**试题分析**

本题主要考查图的存储结构，常见的图的存储结构有邻接矩阵存储和邻接表存储，其中在邻接矩阵存储方式中，矩阵中每个元素的值都表示两个点之间的边的信息，如果每两个点之间都有变的信息，那么矩阵中的所有元素都是有效元素，那么从存储空间的利用率角度来看，其利用率是100%，而采用邻接表存储其存储空间利用率肯定低于100%，因为采用邻接表存储，不仅要存储边的信息，还要存储节点信息，指针信息等。  
这种情况下，这个图很显然是一个完全图，因此从存储空间的利用率角度来看，完全图适合采用邻接矩阵存储。

**试题答案**

（60）C

# 试题79(2012年上半年试题61)

递增序列A（a1，a2，…，an）和B（b1，b2，…，bn）的元素互不相同，若需将它们合并为一个长度为2n的递增序列，则当最终的排列结果为（）时，归并过程中元素的比较次数最多。

（61）A．a1，a2，…，an，b1，b2，…，bn  
B.b1，b2，…，bn，a1，a2，…，an  
C.a1，b1，a2，b2，…，ai，bi，…，an，bn  
D.a1，a2，…，ai/2，b1，b2，…，bi/2，ai/2+1，ai/2+2，…，an，bi/2+1，…，bn

**试题分析**

要将两个有序序列归并为一个有序序列时，当一个序列的最大值小于另一个序列的最小值时，这时需要比较的次数最小。当获得新序列后，两个序列的元素交替的情况（如选项C），这种情况下需比较的次数最多。

**试题答案**

（61）C

# 试题80(2012年上半年试题62)

以下关于渐近符号的表示中，不正确的是（）。

（62）A．IMG_312  
B.IMG_313  
C.IMG_314  
D.IMG_315

**试题分析**

很明显，在本题中错误的是C选项的描述。

**试题答案**

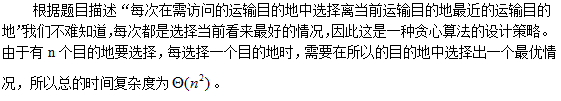
（62）C

# 试题81(2012年上半年试题63-64)

某货车运输公司有一个中央仓库和n个运输目的地，每天要从中央仓库将货物运输到所有运输目的地，到达每个运输目的地一次且仅一次，最后回到中央仓库。在两个地点i和j之间运输货物存在费用Cij。为求解旅行费用总和最小的运输路径，设计如下算法：首先选择离中央仓库最近的运输目的地1，然后选择离运输目的地1最近的运输目的地2，…，每次在需访问的运输目的地中选择离当前运输目的地最近的运输目的地，最后回到中央仓库。刚该算法采用了（）算法设计策略，其时间复杂度为（）。

（63）A．分治  
B.动态规划  
C.贪心  
D.回溯  
  
（64）A．IMG_316  
B.IMG_317  
C.IMG_318  
D.IMG_319

**试题分析**



**试题答案**

（63）C（64）A

# 试题82(2012年上半年试题65)

现要对n个实数（仅包含正实数和负实数）组成的数组A进行重新排列，使得其中所有的负实数都位于正实数之前。求解该问题的算法的伪代码如下所示，则该算法的时间和空间更杂度分别为（）。  
i=0;j=n-1;  
whilei<jdo  
whileA[i]<0do  
i=i+1;  
whileA[j]>0do  
j=j-1;  
ifi<jdo  
交换A[i]和A[j];

（65）A．IMG_321  
B.IMG_322  
C.IMG_323  
D.IMG_324

**试题分析**

根据程序不难看出，要将负实数位于正实数之前，其实就是对所有元素进行了一次遍历，正实数和负实数互换位置即可，因此其时间复杂度为O(n)，由于元素A[i]和A[j]互换时，需要一个临时存储空间来存放元素，因此其空间复杂度为O(1)。

**试题答案**

（65）C

# 试题83(2011年下半年试题4)

【说明】  
设某一机器由n个部件组成，每一个部件都可以从m个不同的供应商处购得。供应商j供应的部件i具有重量wij和价格cij。设计一个算法，求解总价格不超过上限cc的最小重量的机器组成。  
采用回溯法来求解该问题：  
首先定义解空间。解空间由长度为n的向量组成，其中每个分量取值来自集合{1，2，…，m}，将解空间用树形结构表示。  
接着从根结点开始，以深度优先的方式搜索整个解空间。从根结点开始，根结点成为活结点，同时也成为当前的扩展结点。向纵深方向考虑第一个部件从第一个供应商处购买，得到一个新结点。判断当前的机器价格（c11）是否超过上限（cc），重量（w11）是否比当前已知的解（最小重量）大，若是，应回溯至最近的一个活结点；若否，则该新结点成为活结点，同时也成为当前的扩展结点，根结点不再是扩展结点。继续向纵深方向考虑第二个部件从第一个供应商处购买，得到一个新结点。同样判断当前的机器价格（c11+c21）是否超过上限（cc），重量（w11+w21）是否比当前已知的解（最小重量）大。若是，应回溯至最近的一个活结点；若否，则该新结点成为活结点，同时也成为当前的扩展结点，原来的结点不再是扩展结点。以这种方式递归地在解空间中搜索，直到找到所要求的解或者解空间中已无活结点为止。

【C代码】  
下面是该算法的C语言实现。  
（1）变量说明  
n：机器的部件数  
m：供应商数  
cc：价格上限  
w[][]：二维数组，w[i][j]表示第j个供应商供应的第i个部件的重量  
c[][]：二维数组，c[i][j]表示第j个供应商供应的第i个部件的价格  
bestW：满足价格上限约束条件的最小机器重量  
bestC：最小重量机器的价格  
bestX[]：最优解，一维数组，bestX[i]表示第i个部件来自哪个供应商  
cw：搜索过程中机器的重量  
cp：搜索过程中机器的价格  
x[]：搜索过程中产生的解，x[i]表示第i个部件来自哪个供应商  
i：当前考虑的部件，从0到n-1  
j：循环变量  
（2）函数backtrack  
intn=3;  
intm=3;  
intcc=4;  
intw[3][3]={{1，2，3}，{3，2，1}，{2，2，2}};  
intc[3][3]={{1，2，3}，{3，2，1}，{2，2，2}};  
intbestW=8;  
intbestC=0;  
intbestX[3]={0，0，0};  
intcw=0;  
intcp=0;  
intx[3]={0，0，0};  
intbacktrack(inti){  
intj=0;  
intfound=0;  
if(i>n-1){/\*得到问题解\*/  
bestW=cw;  
bestC=cp;  
for(j=0;j<n;j++){  
（1）;  
}  
return1;  
}  
if(cp<=cc){/\*有解\*/  
found=1;  
}  
for(j=0;（2）;j++){  
/\*第i个部件从第j个供应商购买\*/  
（3）;  
cw=cw+w[i][j];  
cp=cp+c[i][j];  
if(cp<=cc&&（4）){/\*深度搜索，扩展当前结点\*/  
if(backtrack(i+1)){found=1;}  
}  
/\*回溯\*/  
cw=cw-w[i][j];  
（5）;  
}  
returnfound;  
}

**试题分析**

本题考查回溯法的应用。  
在题目的描述中告诉了我们回溯法的基本思想。其实回溯法主要有两个过程，一个是向前探索，只要在当前满足设定的判定条件时，才向前探索，而另外一个就是回溯，在两种情况下，需要回溯，其分别是当不满足设定条件时和求的一个解的时候。  
下面我们来具体分析本试题。根据题目给出的注释，我们知道第（1）空所处的位置是得到问题的一个解时，我们该怎么办，根据题目描述，应该是将这个解记录下来，存放到bestX数组当中，而求得的解是保存在x数组当中的，因此这里需要循环将x数组中的元素值赋给bestX数组，因此第（1）空答案为bestX[j]=x[j]。  
第（2）空是for循环中的循环判定条件，根据题目注释我们知道该循环的作用是确定第i个部件从第j个供应商购买，那么在确定第i个部件到底是从哪个供应商购买时，需要比较从各供应商购买的情况，因此循环的次数为供应商数，因此第（2）空答案是j<m。结合这个循环体当中的语句和我们对回溯法的理解，我们可以发现循环下面的语句是要考虑将第i个部件从供应商j当中购买，也就是j是当前解的一部分，因此需要将j记录到解当中来，所以第（3）空应该是x[i]=j。  
第（4）空是if语句中的一个条件，根据题目注释，我们可以知道如果该if语句表达式的计算结果为真，需要进行深度搜索，扩展当前结点，那么如果要继续向前探索，就需要满足设定的条件，也就是当前总重量要小于bestW，而当前总价格要小于等于cc，因此第（4）空的答案应该填cw<bestW。  
根据题目注释，第（5）空是在回溯下面的语句，根据回溯的原则我们可以知道，回溯时，要将当前考虑的结点的重量和价格从总重量和总价格中减去，因此第（5）的答案是cp=cp-c[i][j]。

**试题答案**

（4）

（1）bestX[j]=x[j]  
（2）j<m  
（3）x[i]=j  
（4）cw<bestW  
（5）cp=cp-c[i][j]

# 试题84(2011年下半年试题21)

若二维数组arr[1..M，1..N]的首地址为base，数组元素按列存储且每个元素占用K个存储单元，则元素arr[i，j]在该数组空间的地址为（）。

（21）A．base+((i-1)\*M+j-1)\*K  
B.base+((i-1)\*N+j-1)\*K  
C.base+((j-1)\*M+i-1)\*K  
D.base+((j-1)\*N+i-1)\*K

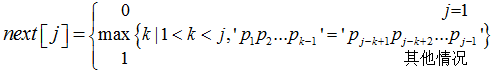
**试题分析**

题目告诉我们是按列存储，那么在存储元素arr[i，j]以前，应该存放了j-1列，而每一列中有M个元素（即数组的行数），那么应该有（j-1）\*M个元素，而在第j列中，存放元素arr[i，j]以前，应该有i-1个元素被存放，因此，在存放元素arr[i，j]以前总共有（j-1）\*M+i-1个元素被存放，而每个元素占用K个存储单元，因此本题答案选C。

**试题答案**

（21）C

# 试题85(2011年下半年试题57)

在KMP模式匹配算法中，需要求解模式串p的next函数值，其定义如下（其中，j为模式串中字符的序号）。对于模式串“abaabaca”，其next函数值序列为（）。  


（57）A．01111111  
B.01122341  
C.01234567  
D.01122334

**试题分析**

KMP模式匹配算法通俗点说就是一种在一个字符串中定位另一个串的高效算法。其实我们在做这个题目时，也可以不需要知道KMP模式匹配算法，可以根据题目给出的定义式来求解。

当j=1时，很显然next[1]=0。

当j=2时，由于1<k<j，因此k无法取到合适值，因此next[2]=1。

当j=3时，k的取值为2，那么等号左边的‘P1P2…PK-1’字符串就是P1，为字符串中的第一个字符a，而右边就是P2，即字符串中的第二个字符b，显然，它们不相等，因此next[3]=1。

当j=4时，k可以取值2或者3，取值为2时，等号左边为第一个字符a，而等号右边为P3，也是字符a，因此相等，但这个时候我们还要判定当k取值为3时，等号左边为第一与第二个字符，即‘ab’，而右边为‘ba’，显然不相等，因此next[4]=2。

同理我们可以求得当j=5，j=6的结果，本题正确答案选B。

**试题答案**

（57）B

# 试题86(2011年下半年试题58)

对于线性表（由n个同类元素构成的线性序列），采用单向循环链表存储的特点之一是（）。

（58）A．从表中任意结点出发都能遍历整个链表  
B.对表中的任意结点可以进行随机访问  
C.对于表中的任意一个结点，访问其直接前驱和直接后继结点所用时间相同  
D.第一个结点必须是头结点

**试题分析**

采用单向循环链表存储的特点之一是从表中任意结点出发都能遍历整个链表，另外便于元素的元素节点的删除与插入。如需要对表中的任意节点进行随机访问需采用顺序存储结构。

**试题答案**

（58）A

# 试题87(2011年下半年试题59)

无向图中一个顶点的度是指图中与该顶点相邻接的顶点数。若无向图G中的顶点数为n，边数为e，则所有顶点的度数之和为（）。

（59）A．n\*e  
B.ne  
C.2n  
D.2e

**试题分析**

在无向图中，一条边连接两个顶点，即如果存在一条边，那么与这条边相关的两个顶点的度都为加1，那么总的度就应该加2，因此，如果图中有n条边，那么所有顶点的度数之和就应该为2e。

**试题答案**

（59）D

# 试题88(2011年下半年试题60)

一棵满二叉树，其每一层结点个数都达到最大值，对其中的结点从l开始顺序编号，即根结点编号为1，其左、右孩子结点编号分别为2和3，再下一层从左到右的编号为4、5、6、7，依此类推，每一层都从左到右依次编号，直到最后的叶子结点层为止，则用（）可判定编号为m和n的两个结点是否在同一层。

（60）A．IMG_326  
B.IMG_327  
C.IMG_328  
D.IMG_329

**试题分析**

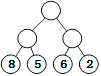
如果是满二叉树，那么其第n层的结点数应该是第n-1层结点数的两倍，从根（第一层）开始，各层的结点数应分别是2n-1个，其中n为当前的层次，因此一颗m层的满二叉树，其总的结点数位2m-1个。而如果知道结点编号x，我们可以用log2x+1来求取该结点属于那一层。

**试题答案**

（60）B

# 试题89(2011年下半年试题61)

（）一是由权值集合{8，5，6，2}构造的哈夫曼树（最优二叉树）。

（61）A．  
B.  
C.  
D.

**试题分析**

构造哈夫曼树的过程是首先从给出的权值集合中找出最小的两个权值，即2和5，用它们作为子结点构建一个父结点，其权值为7，然后将7放入权值集合中并将2和5去掉，再在集合中找出两个最小权值，即6和7，而7已经在我们构造的树种，然后用6和7作为子结点构建一个父结点，其权值为6+7=13，然后同样将13放入权值集合中并将6和7去掉，最好集合中只有8和13，将它们作为子结点构建一个父结点，就得到了这棵哈夫曼树。

**试题答案**

（61）C

# 试题90(2011年下半年试题62)

迪杰斯特拉(Dijkstra)算法用于求解图上的单源点最短路径。该算法按路径长度递增次序产生最短路径，本质上说，该算法是一种基于（）策略的算法。

（62）A．分治  
B.动态规划  
C.贪心  
D.回溯

**试题分析**

分治法：对于一个规模为n的问题，若该问题可以容易地解决（比如说规模n较小）则直接解决；否则将其分解为k个规模较小的子问题，这些子问题互相独立且与原问题形式相同，递归地解这些子问题，然后将各子问题的解合并得到原问题的解。

动态规划法：这种算法也用到了分治思想，它的做法是将问题实例分解为更小的、相似的子问题，并存储子问题的解而避免计算重复的子问题。

贪心算法：它是一种不追求最优解，只希望得到较为满意解的方法。贪心算法一般可以快速得到满意的解，因为它省去了为找到最优解而穷尽所有可能所必须耗费的大量时间。贪心算法常以当前情况为基础做最优选择，而不考虑各种可能的整体情况，所以贪心算法不要回溯。

回溯算法（试探法）：它是一种系统地搜索问题的解的方法。回溯算法的基本思想是：从一条路往前走，能进则进，不能进则退回来，换一条路再试。其实现一般要用到递归和堆栈。

针对单源最短路径问题，由Dijkstra提出了一种按路径长度递增的次序产生各顶点最短路径的算法。若按长度递增的次序生成从源点s到其他顶点的最短路径，则当前正在生成的最短路径上除终点以外，其余顶点的最短路径均已生成（将源点的最短路径看做是已生成的源点到其自身的长度为0的路径）。这是一种典型的贪心策略，就是每递增一次，经对所有可能的源点、目标点的路径都要计算，得出最优。

带权图的最短路径问题即求两个顶点间长度最短的路径。其中：路径长度不是指路径上边数的总和，而是指路径上各边的权值总和。

**试题答案**

（62）C

# 试题91(2011年下半年试题63)

在有n个无序无重复元素值的数组中查找第i小的数的算法描述如下：任意取一个元素r，用划分操作确定其在数组中的位置，假设元素r为第k小的数。若i等于k，则返回该元素值；若i小于k，则在划分的前半部分递归进行划分操作找第i小的数；否则在划分的后半部分递归进行划分操作找第k-i小的数。该算法是一种基于（）策略的算法。

（63）A．分治  
B.动态规划  
C.贪心  
D.回溯

**试题分析**

分治法：对于一个规模为n的问题，若该问题可以容易地解决（比如说规模n较小）则直接解决；否则将其分解为k个规模较小的子问题，这些子问题互相独立且与原问题形式相同，递归地解这些子问题，然后将各子问题的解合并得到原问题的解。  
动态规划法：这种算法也用到了分治思想，它的做法是将问题实例分解为更小的、相似的子问题，并存储子问题的解而避免计算重复的子问题。  
贪心算法：它是一种不追求最优解，只希望得到较为满意解的方法。贪心算法一般可以快速得到满意的解，因为它省去了为找到最优解而穷尽所有可能所必须耗费的大量时间。贪心算法常以当前情况为基础做最优选择，而不考虑各种可能的整体情况，所以贪心算法不要回溯。  
回溯算法（试探法）：它是一种系统地搜索问题的解的方法。回溯算法的基本思想是：从一条路往前走，能进则进，不能进则退回来，换一条路再试。其实现一般要用到递归和堆栈。

**试题答案**

（63）A

# 试题92(2011年下半年试题64)

对n个元素值分别为-1、0或1的整型数组A进行升序排序的算法描述如下：统计A中-1、0和1的个数，设分别为n1、n2和n3，然后将A中的前n1个元素赋值为-1，第n1+l到nl+n2个元素赋值为0，最后n3个元素赋值为1。该算法的时间复杂度和空间复杂度分别为（）。

（64）A．IMG_334  
B.IMG_335  
C.IMG_336  
D.IMG_337

**试题分析**

时间复杂度是指程序运行从开始到结束所需要的时间。通常分析时间复杂度的方法是从算法中选取一种对于所研究的问题来说是基本运算的操作，以该操作重复执行的次数作为算法的时间度量。一般来说，算法中原操作重复执行的次数是规模n的某个函数T(n)。由于许多情况下要精确计算T(n)是困难的，因此引入了渐进时间复杂度在数量上估计一个算法的执行时间。其定义如下：  
如果存在两个常数c和m，对于所有的n，当n≥m时有f(n)≤cg(n)，则有f(n)=O(g(n))。也就是说，随着n的增大，f(n)渐进地不大于g(n)。例如，一个程序的实际执行时间为T(n)=3n3+2n2+n，则T(n)=O(n3)。  
在本题中，根据题目的描述，我们可以知道，遍历完整个数组中的元素，和修改数组中各元素的值都需要时间n，因此是2n，那么该算法的时间复杂度为O(n)。  
空间复杂度是指程序运行从开始到结束所需的辅助存储量，在本题中，只需要辅助存储量来存储统计的元素个数，因此其空间复杂度为O(1)。

**试题答案**

（64）A

# 试题93(2011年下半年试题65)

设算法A的时间复杂度可用递归式IMG_338表示，算法B的时间复杂度可用递归式IMG_339表示，若要使得算法B渐进地快于算法A，则a的最大整数为（）。

（65）A．48  
B.49  
C.13  
D.14

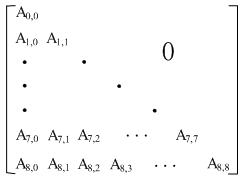
**试题分析**

题目要求使得算法B渐进地快于算法A，即B的时间复杂度小于A的时间复杂度。  
通过简单的化简我们可以得到算法A的T(n)=49T(n/4)+2n2，而算法B的T(n)=aT(n/4)+n2，而根据渐进的规则，算法A的时间复杂度应该为49T(n/4)，而算法B的时间复杂度应该为aT(n/4)，因此a的取值应该要小于49。

**试题答案**

（65）A

# 试题94(2011年上半年试题57)

设下三角矩阵（上三角部分的元素值都为0）A[0．．n，0．．n]如下所示，将该三角矩阵的所有非零元素（即行下标不小于列下标的元素）按行优先压缩存储在容量足够大的数组M[]中（下标从1开始），则元素A[i，j](O≤i≤n，j≤i)存储在数组M的（）中。  


（57）A．IMG_341  
B.IMG_342  
C.IMG_343  
D.IMG_344

**试题分析**

对于这个题目，我们可以这样理解，题目要求按行优先，其含义就是先存储完第一行，在开始存储第二行，然后再存储第三行的非0元素，依次类推。这样的话我们就发现了一个规律，第1行只有一个元素，  
第二行2个元素，  
第三行3个元素，  
第n行n个元素。  
这个规律是一个递增数列。那么元素A[i，j]是第几行第几列的元素呢？由于下标是从0开始的（这个要特别注意），那么下标为i的应该就是第i+1行，因此在存储下标为i的这行之前，应该存放了i行元素，其中第i行的元素个数为i个，那么在存放第i+1行之前，应该存放的元素个数总和为i(i+1)/2，。那么当存放到第i+1行时，在存放下标为j的元素前，同样的道理应该存放了j个元素，因此在存放元素A[i，j]之前，总共存放了的元素个数总和为i(i+1)/2+j，因此元素A[i，j]应该是第i(i+1)/2+j+1个要存放的元素，由于存放的数组M是从下标为1开始的。因此元素A[i，j]存储在数组M的M[i(i+1)/2+j+1]中。

**试题答案**

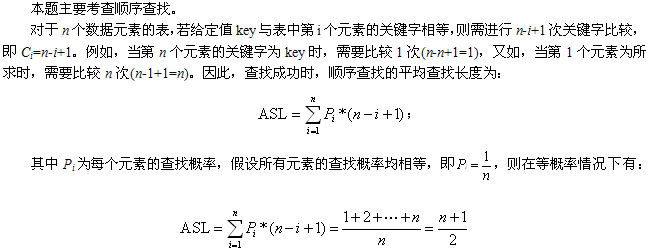
（57）A

# 试题95(2011年上半年试题58)

对n个元素的有序表A[1．．n]进行顺序查找，其成功查找的平均查找长度（即在查找表中找到指定关键码的元素时，所进行比较的表中元素个数的期望值）为（）。

（58）A．n  
B.(n+1)／2  
C.log2n  
D.n2

**试题分析**



**试题答案**

（58）B

# 试题96(2011年上半年试题59)

在（）中，任意一个结点的左、右子树的高度之差的绝对值不超过1。

（59）A．完全二叉树  
B.二叉排序树  
C.线索二叉树  
D.最优二叉树

**试题分析**

本题主要考查一些特殊二叉树的性质。  
若二叉树中最多只有最下面两层的结点度数可以小于2，并且最下面一层的叶子结点都依次排列在该层最左边的位置上，则这样的二叉树称为完全二叉树，因此在完全二叉树中，任意一个结点的左、右子树的高度之差的绝对值不超过1。  
二叉排序树的递归定义如下：二叉排序树或者是一棵空树；或者是具有下列性质的二叉树：  
(1)若左子树不空，则左子树上所有结点的值均小于根结点的值；  
(2)若右子树不空，则右子树上所有结点的值均大于根结点的值；  
(3)左右子树也都是二叉排序树。  
在n个结点的二叉树链式存储中存在n+1个空指针，造成了巨大的空间浪费，为了充分利用存储资源，可以将这些空链域存放指向结点在遍历过程中的直接前驱或直接后继的指针，这种空链域就称为线索，含有线索的二叉树就是线索二叉树。  
最优二叉树即哈夫曼树。

**试题答案**

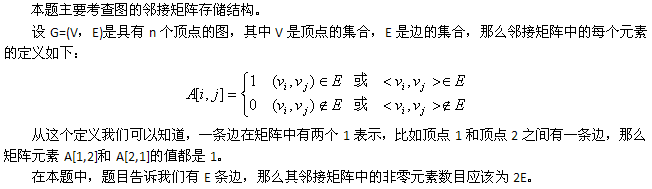
（59）A

# 试题97(2011年上半年试题60)

设一个包含N个顶点、E条边的简单无向图采用邻接矩阵存储结构（矩阵元素A[i][j]等于1/0分别表示顶点i与顶点j之间有／无边），则该矩阵中的非零元素数目为（）。

（60）A．N  
B.E  
C.2E  
D.N+E

**试题分析**



**试题答案**

（60）C

# 试题98(2011年上半年试题61)

对于关键字序列(26，25，72，38，8，18，59)，采用散列函数H(Key)=Keymod13构造散列表（哈希表）。若采用线性探测的开放定址法解决冲突（顺序地探查可用存储单元），则关键字59所在散列表中的地址为（）。

（61）A．6  
B.7  
C.8  
D.9

**试题分析**

根据题目给出的散列函数我们可以分别计算出关键字(26，25，72，38，8，18，59)对应的散列地址分别为(0，12，7，12，8，5，7)。  
开放定址处理冲突的基本思路是为发生冲突的关键字在散列表中寻找另一个尚未占用的位置，其解决冲突能力的关键取决于探测序列，在本题中，题目告诉我们采用顺序探查法，即增量为1的线性探测法，在该线性探测法中，设Hi(1≤i<m)为第i次在散列表中探测的位置，其中增量序列为{1，2，3，4，5，…，m-1}则有：  
Hi=(H(Key)+i)%m  
其中H(Key)为散列函数，m为散列表长度，i为增量序列。而本题中m=13。因此本题的散列表构造过程如下：  
（1）关键字26，25，72由散列函数H(key)得到没有冲突的散列地址而直接存入散列表中。  
（2）计算关键38的散列地址为12，发生冲突（与关键字25冲突），其第一次线性探测地址为（12+1）%13=0，但仍然发生冲突（与关键字26冲突），因此需要进行第二次线性探测，其地址为（12+2）%13=1，这时没有发生冲突，即将38存入地址为1的空间。  
（3）接着将关键字8，18计算其散列地址，由于没有冲突，即分别存入散列地址为8和5的空间中。  
（4）计算关键59的散列地址为7，发生冲突（与关键字72冲突），其第一次线性探测地址（7+1）%13=8，但仍然发生冲突（与关键字8冲突），因此需要进行第二次线性探测，其地址为（7+2）%13=9，这时没有发生冲突，即将59存入地址为9的存储空间。  
因此本题的答案选D。

**试题答案**

（61）D

# 试题99(2011年上半年试题62)

要在8\*8的棋盘上摆放8个“皇后”，要求“皇后”之间不能发生冲突，即任何两个“皇后”不能在同一行、同一列和相同的对角线上，则一般采用（）来实现。

（62）A．分治法  
B.动态规划法  
C.贪心法  
D.回溯法

**试题分析**

回溯法是一种选优搜索法，按选优条件向前搜索，以达到目标。但当探索到某一步时，发现原先选择并不优或达不到目标，就退回一步重新选择，这种走不通就退回再走的技术为回溯法。回溯法求解的过程其实是搜索整个解空间，来找到最优的解。而“皇后”问题是一个典型的用回溯法求解的问题。

**试题答案**

（62）D

# 试题100(2011年上半年试题63)

分治算法设计技术（）。

（63）A．一般由三个步骤组成：问题划分、递归求解、合并解  
B.一定是用递归技术来实现  
C.将问题划分为k个规模相等的子问题  
D.划分代价很小而合并代价很大

**试题分析**

分治的基本思想就是：对于一个规模为n的问题，若该问题可以容易地解决（比如说规模n较小）则直接解决，否则将其分解为k个规模较小的子问题，这些子问题互相独立且与原问题形式相同，递归地解这些子问题，然后将各子问题的解合并得到原问题的解。  
所以分治算法设计技术主要包括三个步骤，分别是问题划分、递归求解、合并解。

**试题答案**

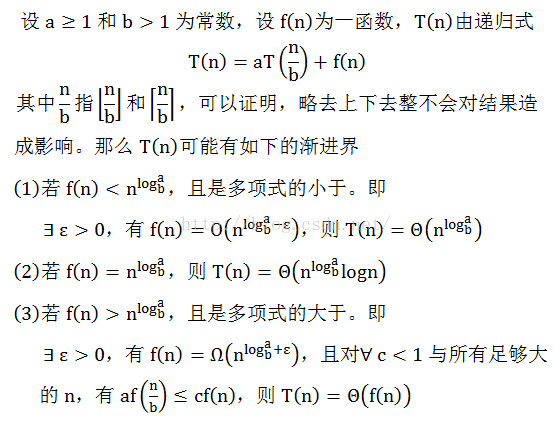
（63）A

# 试题101(2011年上半年试题64)

某算法的时间复杂度可用递归式IMG_347表示，若用IMG_348表示，则正确的是（）。

（64）A．IMG_349  
B.IMG_350  
C.IMG_351  
D.IMG_352

**试题分析**

本题采用公式法求解，以下是相关公式。  
  
本题中，a=6，b=5，f(n)=n。所以套用公式（1），得到A选项为正确答案。

**试题答案**

（64）A

# 试题102(2011年上半年试题65)

用插入排序和归并排序算法对数组<3，1，4，1，5，9，6，5>进行从小到大排序，则分别需要进行（）次数组元素之间的比较。

（65）A．12，14  
B.10，14  
C.12，16  
D.10，16

**试题分析**

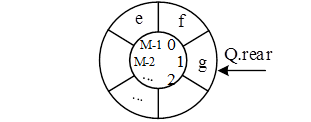
插入排序的基本思想是逐个将待排序元素插入到已排序的有序表中。假设n个待排序元素存储在数组R[n+1]中(R[0]预留)，则：  
(1)初始时数组R[1..1]中只包含元素R[1]，则数组R[1..1]必定有序；  
(2)从i=2到n，执行步骤3；  
(3)此时，数组R被划分成两个子区间，分别是有序区间R[1..i-1]和无序区间R[i..n]，将当前无序区间的第1个记录R[i]插入到有序区间R[1..i]中适当的位置上，使R[1..i]变为新的有序区间。  
在实现的过程中，设置监视哨R[0]，并从R[i-1]到R[0]查找元素R[i]的插入位置  
那么用插入排序对数组<3，1，4，1，5，9，6，5>进行排序的过程为：  
原元素序列：监视哨（3），1，4，1，5，9，6，5  
第一趟排序：3（1，3），4，1，5，9，6，53插入时与1比较1次  
第二趟排序：4（1，3，4），1，5，9，6，54插入时与3比较1次  
第三趟排序：1（1，1，3，4），5，9，6，51插入时比较3次  
第四趟排序：5（1，1，3，4，5），9，6，55插入时与4比较1次  
第五趟排序：9（1，1，3，4，5，9），6，59插入时与5比较1次  
第六趟排序：6（1，1，3，4，5，6，9），56插入时与9和5分别比较1次  
第七趟排序：5（1，1，3，4，5，5，6，9）5插入时与9，6，5分别比较1次  
那么整个排序过程需要比较的次数为12次。  
归并排序的思想是将两个相邻的有序子序列归并为一个有序序列，然后再将新产生的相邻序列进行归并，当只剩下一个有序序列时算法结束。其基本步骤如下：  
(1)将n个元素的待排序序列中每个元素看成有序子序列，对相邻子序列两两合并，则将生成个子有序序列，这些子序列中除了最后一个子序列长度可能小于2外，其他的序列长度都等于2；  
(2)对上述个长度为2的子序列再进行相邻子序列的两两合并，则产生个子有序序列，同理，只有最后一个子序列的长度可能小于4；  
(3)第i趟归并排序为，对上述个长度为i的子序列两两合并，产生个长度为2i的子有序序列；  
(4)重复执行此步骤，直到生成长度为n的序列为止。  
那么用归并排序对数组<3，1，4，1，5，9，6，5>进行排序的过程为：  
原元素序列：3，1，4，1，5，9，6，5  
第一趟排序：[1，3]，[1，4]，[5，9]，[5，6]比较4次  
第二趟排序：[1，1，3，4]，[5，5，6，9]前半部分比较3次，后半部分比较3次  
第三趟排序：[1，1，3，4，5，5，6，9]5分别与1，1，3，4比较一次  
所以整个排序过程需要比较的次数为14次。

**试题答案**

（65）A

# 试题103(2010年下半年试题57)

设循环队列Q的定义中有rear和len两个域变量，其中rear表示队尾元素的指针，len表示队列的长度，如下图所示（队列长度为3，队头元素为e）。设队列的存储空间容量为M，则队头元素的指针为（）。



（57）A．(Q.rear+Q.len-1）  
B.(Q.rear+Q.len-1+M)%M  
C.(Q.rear-Q.len+1）  
D.(Q.rear-Q.len+1+M)%M

**试题分析**

对于循环队列，求队头元素的指针的计算公式为：(rear-len+1+M)%M。  
求队列中元素个数公式为：(rear-fear+M)%M。其中fear表示队列的对头指针。

**试题答案**

（57）D

# 试题104(2010年下半年试题58)

下面关于哈夫曼树的叙述中，正确的是（）。

（58）A．哈夫曼树一定是完全二叉树  
B.哈夫曼树一定是平衡二叉树  
C.哈夫曼树中权值最小的两个结点互为兄弟结点  
D.哈夫曼树中左孩子结点小于父结点、右孩子结点大于父结点

**试题分析**

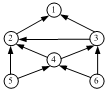
哈夫曼树是一种特殊的二叉树，但它不是完全二叉树，也不是平衡二叉树，给出n个权值{w1，w2，…，wn}构造一棵具有n个叶子结点的哈夫曼树的方法如下：

第一步，构造n个只有根结点的二叉树集合F={T1，T2，…，Tn}，其中每棵二叉树Ti的根结点带权为Wi(1≤k≤n)；  
第二步，在集合F中选取两棵根结点的权值最小的二叉树作为左右子树，构造一棵新的二叉树，令新二叉树根结点的权值为其左、右子树上根结点的权值之和；  
第三步，在F中删除这两棵二叉树，同时将新得到的二叉树加入到F中；  
第四步，重复第二步和第三步，直到F只含有一棵二叉树为止，这棵二叉树便是哈夫曼树。  
综上所述，我们可以知道哈夫曼树中权值最小的两个结点互为兄弟结点。

**试题答案**

（58）C

# 试题105(2010年下半年试题59)

（）是右图的合法拓扑序列。  


（59）A．654321  
B.123456  
C.563421  
D.564213

**试题分析**

本题主要考查拓扑序列。  
在给出拓扑图求拓扑序列时，我们应该掌握一个关键因素，那就是箭头的画出节点在箭头指向节点前，如果一个节点被很多箭头所指，那么应该要在所有这些箭头的画出节点之后才是本节点。拓扑序列的开始节点应该是没有箭头所指的节点，在本题中应该是5或6，这里需要注意它们谁在最前面都可以。那么按照这个原则我们就可以知道本题的拓扑序列应该为654321或者564321。

**试题答案**

（59）A

# 试题106(2010年下半年试题60)

某一维数组中依次存放了数据元素15，23，38，47，55，62，88，95，102，123，采用折半（二分）法查找元素95时，依次与（）进行了比较。

（60）A．62，88，95  
B.62，95  
C.55，88，95  
D.55，95

**试题分析**

本题主要考查折半（二分）法查找算法。这里首先就需要我们能清楚理解该查找算法。  
在本题中，给出数据序列为15，23，38，47，55，62，88，95，102，123，其中有10个元素，那么首先进行比较的应该是第5个元素，即55，由于95大于55，那么应该在后半部分进行查找，这是应该与第8个元素进行比较，刚好是95，查找成功，然后结束。因此比较的元素有55和95。

**试题答案**

（60）D

# 试题107(2010年下半年试题61)

己知一棵度为3的树（一个结点的度是指其子树的数目，树的度是指该树中所有结点的度的最大值）中有5个度为1的结点，4个度为2的结点，2个度为3的结点，那么，该树中的叶子结点数目为（）。

（61）A．10  
B.9  
C.8  
D.7

**试题分析**

由于叶子节点没有子树，因此它的度为0。而除根节点外，其它的节点都应该可以做为子节点，即可以用于计算度。  
在本题中告我有5个度为1的结点，4个度为2的结点，2个度为3的结点，那么树中总的度数为5+8+6=19，因此树中除根节点外，就应该有19个节点，所以树中总的节点数应该为20，那么叶子节点数=20-5-4-2=9。

**试题答案**

（61）B

# 试题108(2010年下半年试题62)

某算法的时间复杂度可用递归式IMG_356表示，若用IMG_357表示该算法的渐进时间复杂度的紧致界，则正确的是（）。

（62）A．IMG_358  
B.IMG_359  
C.IMG_360  
D.IMG_361

**试题分析**

在本题中，我们关键要理解算法的渐进紧致界的概念，举个例子来说吧，假设当N>N0时，函数f(N)在一个常数因子范围内等于g(N)，则称g(n)是f(n)的一个渐近紧致界。  
本题中给出的递归式的渐进紧致界应该是A。

**试题答案**

（62）A

# 试题109(2010年下半年试题63)

用动态规划策略求解矩阵连乘问题M1\*M2\*M3\*M4，其中M1(20\*5)、M2(5\*35)、M3(35\*4)和M4(4\*25)，则最优的计算次序为（）。

（63）A．IMG_362  
B.IMG_363  
C.IMG_364  
D.IMG_365

**试题分析**

这个题目的关键是要求最优的计算次序，也就是要求计算过程中，乘法的次数最小。如果用选项A的次序来计算，需要计算的乘法次数为：20\*5\*35+20\*35\*4+20\*4\*25。同样我们可以求出其它三种方法所需的乘法次数。其中最小的是选项C的5\*35\*4+20\*5\*4+20\*4\*25。

**试题答案**

（63）C

# 试题110(2010年下半年试题65)

（）不能保证求得0-1背包问题的最优解。

（65）A．分支限界法  
B.贪心算法  
C.回溯法  
D.动态规划策略

**试题分析**

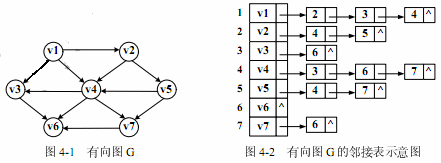
分支限界法一般以广度优先或以最小耗费（最大效益）优先的方式搜索问题的解空间，那么肯定能找出最优解。  
贪心算法的思想是：总是做出在当前来说是最好的选择，而并不从整体上加以考虑，它所做的每步选择只是当前步骤的局部最优选择，但从整体来说不一定是最优的选择。所以用该算法并不能保证求得0-1背包问题的最优解。  
回溯法的思想是：按选优条件向前搜索，以达到目标。但当搜索到某一步时，发现原先选择并不优或达不到目标，就退回一步重新选择。它其实是遍历了整个解空间，所以肯定能找到最优解。  
动态规划法的思想是：在求解问题中，对于每一步决策，列出各种可能的局部解，再依据某种判定条件，舍弃那些肯定不能得到最优解的局部解，在每一步都经过筛选，以每一步都是最优解来保证全局是最优解。它能求得0-1背包问题的最优解。

**试题答案**

（65）B

# 试题111(2010年上半年试题4)

阅读下列说明和C代码，回答问题1至问题3，将解答写在答题纸的对应栏内。  
【说明】  
　　对有向图进行拓扑排序的方法是：  
　　（1）初始时拓扑序列为空；  
　　（2）任意选择一个入度为0的顶点，将其放入拓扑序列中，同时从图中删除该顶点以及从该顶点出发的弧；  
　　（3）重复（2），直到不存在入度为0的顶点为止（若所有顶点都进入拓扑序列则完成拓扑排序，否则由于有向图中存在回路无法完成拓扑排序）。  
函数int\*TopSort(LinkedDigraphG)的功能是对有向图G中的顶点进行拓扑排序，返回拓扑序列中的顶点编号序列，若不能完成拓扑排序，则返回空指针。其中，图G中的顶点从1开始依次编号，顶点序列为v1，v2，…，vn，图G采用邻接表示，其数据类型定义如下：  
　#defineMAXVNUM50/\*最大顶点数\*/  
　　typedefstructArcNode{/\*表结点类型\*/  
　　intadjvex;/\*邻接顶点编号\*/  
　　structArcNode\*nextarc;/\*指示下一个邻接顶点\*/  
　}ArcNode;  
　　typedefstructAdjList{/\*头结点类型\*/  
　　charvdata;/\*顶点的数据信息\*/  
　　ArcNode\*firstarc;/\*指向邻接表的第一个表结点\*/  
　}AdjList;  
　　typedefstructLinkedDigraph{/\*图的类型\*/  
　　intn;/\*图中顶点个数\*/  
　　AdjListVhead[MAXVNUM];/\*所有顶点的头结点数组\*/  
　}LinkedDigraph;  
　　例如，某有向图G如图4-1所示，其邻接表如图4-2所示。



　　函数TopSort中用到了队列结构(Queue的定义省略)，实现队列基本操作的函数原型如下表所示：

|  |  |
| --- | --- |
| 函数原型 | 说明 |
| voidInitQueue（Queue\*Q） | 初始化队列（构造一个空队列） |
| boolIsEmpty（QueueQ） | 判断队列是否为空，若是则返回true，否则返回false |
| voidEnQueue（Queue\*Q,inte） | 元素入队列 |
| voidDeQueue（Queue\*Q,int\*p） | 元素出队列 |

【C代码】  
　int\*TopSort(LinkedDigraphG){  
　　ArcNode\*p;/\*临时指针，指示表结点\*/  
　　QueueQ;/\*临时队列，保存入度为0的顶点编号\*/  
　　intk=0;/\*临时变量，用作数组元素的下标\*/  
　　intj=0，w=0;/\*临时变量，用作顶点编号\*/  
　　int\*topOrder，\*inDegree;  
　　topOrder=(int\*)malloc((G.n+1)\*sizeof(int));/\*存储拓扑序列中的顶点编号\*/  
　　inDegree=(int\*)malloc((G.n+1)\*sizeof(int));/\*存储图G中各顶点的入度\*/  
　　if(!inDegree||!topOrder)returnNULL;  
　　　（1）;/\*构造一个空队列\*/  
　　for(j=1;j<=G.n;j++){/\*初始化\*/  
　　topOrder[j]=0;inDegree[j]=0;  
　}  
　for(j=1;j<=G.n;j++)/\*求图G中各顶点的入度\*/  
　　for(p=G.Vhead[j].firstarc;p;p=p->nextarc)  
　　inDegree[p->adjvex]+=1;  
　for(j=1;j<=G.n;j++)/\*将图G中入度为0的顶点保存在队列中\*/  
　　if(0==inDegree[j])EnQueue(&Q，j);  
　　while(!IsEmpty(Q)){  
　　　（2）;/\*队头顶点出队列并用w保存该顶点的编号\*/  
　　topOrder[k++]=w;  
　/\*将顶点w的所有邻接顶点的入度减1（模拟删除顶点w及从该顶点出发的弧的操作）\*/  
　　　for(p=G.Vhead[w].firstarc;p;p=p->nextarc){  
　　　（3）-=1;  
　　　if(0==（4）)EnQueue(&Q，p->adjvex);  
　　}/\*for\*/  
　}/\*while\*/  
　　　free(inDegree);  
　　　if(（5）)  
　　　returnNULL;  
　　returntopOrder;  
　}/\*TopSort\*/

【问题1】（9分）

　　根据以上说明和C代码，填充C代码中的空（1）～（5）。  
【问题2】（2分）  
　　对于图4-1所示的有向图G，写出函数TopSort执行后得到的拓扑序列。若将函数TopSort中的队列改为栈，写出函数TopSort执行后得到的拓扑序列。  
【问题3】（4分）  
　　设某有向无环图的顶点个数为n、弧数为e，那么用邻接表存储该图时，实现上述拓扑排序算法的函数TopSort的时间复杂度是（6）。  
若有向图采用邻接矩阵表示（例如，图4-1所示有向图的邻接矩阵如图4-3所示），且将函数TopSort中有关邻接表的操作修改为针对邻接矩阵的操作，那么对于有n个顶点、e条弧的有向无环图，实现上述拓扑排序算法的时间复杂度是（7）。



　　从下列的2道试题（试题五和试题六）中任选1道解答。如果解答的试题数超过1道，则题号小的1道解答有效。

**试题分析**

本题考查数据结构和算法中的拓扑排序算法。  
【问题１】  
拓扑排序是将有向无环图中所有顶点排成一个线性序列的过程，并且该序列满足：若在有向图中从顶点Vi到Vj有一条路径，则在该线性序列中，顶点Vi必然在顶点Vj之前。  
对AOE网进行拓扑排序的方法如下：  
①在AOE网中选择一个入度为零（没有前驱）的顶点且输出它：  
②从网中删除该顶点及其与该顶点有关的所有边；  
③重复上述两步，直至网中不存在入度为零的顶点为止。  
在拓扑排序过程中，需要将入度为0的顶点临时存储起来。函数中用一个队列暂存入度为0且没有进入拓扑序列的顶点。显然，空（l）处应填入InitQueue（&Q）。  
进行拓扑排序之前，应先求出网中每个顶点的入度并存入数组inDegree[]中，从而将“从网中删除该顶点及其与该顶点有关的所有边”的操作转换为“相关顶点的入度减1”，一旦发现某个顶点的入度变为0，就将其编号压入堆栈。从而将选择入度为0的顶点操作转化为令队头所代表的顶点出队。  
根据注释，空（2）处应填入DeQueue(&Q，&w)，实现队头元素出队列的处理。  
题中图采用邻接表存储结构，当指针p指向Vi邻接表中的结点时，p->adjvex表示vi的一个邻接顶点，删除vi至顶点p->adjvex的弧的操作实现为顶点p->adjvex的入度减1，因此，空（3）处应填入inDegree[p->adjvex]，当顶点p->adjvex的入度为0时，需要将其加入队列，因此空(4）处也应填入inDegree[p->adjvex]。  
空（5）处判断是否所有顶点都加入了拓扑序列，算法中变量k用于对加入序列的顶点计数，因此，空（5）处应填入“k＜G.n”或“k!=G.n"。  
【问题2】  
使用栈和队列的差别在于拓扑序列中顶点的排列次序可能不同。对于本题中的有向图，在使用队列的方式更下：  
（1）开始时仅顶点V1的入度为0，因此顶点V1入队：  
（2）队头顶点V1出队，并进入拓扑序列，然后删除从顶点V1出发的弧后，仅使顶点v2的入度为0，因此顶点v2入队：  
（3）队头顶点v2出队，并进入拓扑序列，然后删除从顶点v2出发的弧后，仅使顶点v5的入度为0，因止顶点v5入队；  
（4）队头顶点v5出队，并进入拓扑序列，然后删除从顶点v5出发的弧后，仅使顶点v4的入度为0，因此顶点v4入队；  
（5）队头顶点v4出队，并进入拓扑序列，然后删除从顶点v4出发的弧后，仅使顶点v3和v7的入度为0，因此顶点v3和v7依次入队；  
（6）队头顶点v3出队，并进入拓扑序列，然后删除从顶点v3出发的弧后，没有产生新的入度为0的顶点；  
（7）队头顶点v7出队，并进入拓扑序列，然后删除从顶点v7出发的弧后，使顶点v6的入度为0，因此顶点v6入队；  
（8）队头顶点v6出队，并进入拓扑序列，然后删除从顶点v6出发的弧后，没有产生新的入度为0的顶点，队列已空，因此结束拓扑排序过程，得到的拓扑序列为v1v2v5v4v3v7v6。  
使用栈保存入度为0的顶点时，前4步都是一样的，因为每次仅有一个元素进栈，因此出栈序列与入栈序列一致。到第5步时，v3和v7依次入栈后，出栈时的次序为v7和v3，因此得到的拓扑序列为v1v2v5v4v7v3v6。  
【问题3】  
以邻接表为存储结构时，计算各顶点入度的时间复杂度为O(e)，建立零入度顶点队列的时间复杂度为O(n)。在拓扑排序过程中，（图中无环情况下）每个顶点进出队列各1次，入度减l的操作在while循环中共执行e次，所以总的时间复杂度为O(n+e)。  
以邻接矩阵为存储结构时，计算各顶点入度时需要遍历整个矩阵，因此时间复杂度为O（n）。，建立零入度顶点队列的时间复杂度为O（n）。在拓扑排序过程中，（图中无环情况下）每个顶点进出队列各1次，实现入度减1操作时需遍历每个顶点的行向量1遍（时间复杂度为O(n))，所以总的时间复杂度为O(n2）。

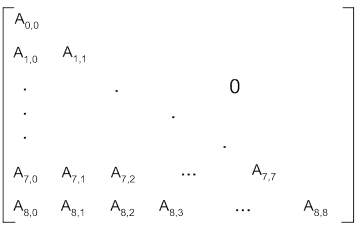
**试题答案**

（4）

【问题1】（9分）　　  
（1）InitQueue(&Q)（1分）注：函数名与参数必须完全正确才可得分  
（2）DeQueue(&Q，&w)（2分）注：函数名与参数必须完全正确才可得分  
（3）inDegree[p->adjvex]及其等价形式（2分）  
（4）inDegree[p->adjvex]及其等价形式（2分）  
（5）k<G.n或k!=G.n（2分）  
【问题2】（2分）  
　　队列方式：v1v2v5v4v3v7v6（或1254376）（1分）  
　栈方式：v1v2v5v4v7v3v6（或1254736）（1分）  
【问题3】（4分）  
　　（6）O(n+e)（2分）  
　　（7）O(n2)（2分）

# 试题112(2010年上半年试题58)

设有如下所示的下三角矩阵A[0..8，0..8]，将该三角矩阵的非零元素（即行下标不小于列下标的所有元素）按行优先压缩存储在数组M[1..m]中，则元素A[i，j]（0<=i<=8，j<=i）存储在数组M的（）中。



（58）A．  
B.IMG_370  
C.IMG_371  
D.IMG_372

**试题分析**

本题考查数据结构基础知识.  
如题图所示，按行方式压缩存储时，A[i，j]之前的元素数目为（1+2+…+i+j）个，数组M的下标从1开始，因此A[i，j]的值存储在MIMG_373中。

**试题答案**

（58）A

# 试题113(2010年上半年试题62)

对于哈希表，如果将装填因子?定义为表中装入的记录数与表的长度之比，那么向表中加入新记录时，（）。

（62）A．的值随冲突次数的增加而递减  
B.越大发生冲突的可能性就越大  
C.等于1时不会再发生冲突  
D.低于0.5时不会发生冲突

**试题分析**

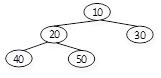
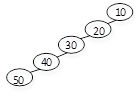
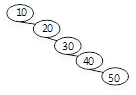
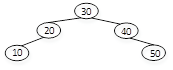
本题考查数据结构基础知识。  
　　装填因子a表示了哈希表的装满程度，显然，a越大发生冲突的可能性就越大。

**试题答案**

（62）B

# 试题114(2010年上半年试题63)

用关键字序列10、20、30、40、50构造的二叉排序树（二叉查找树）为（）。

（63）A．  
B.  
C.  
D.

**试题分析**

本题考查数据结构基础知识。  
　　根据关键字序列构造二叉排序树的基本过程是，若需插入的关键字大于树根，则插入到右子树上，若小于树根，则插入到左子树上，若为空树，则作为树根结点。

**试题答案**

（63）C

# 试题115(2010年上半年试题64)

若某算法在问题规模为n时，其基本操作的重复次数可由下式表示，则该算法的时间复杂度为（）。

IMG_378

（64）A．O(n)  
B.O(n2)  
C.O(logn)  
D.O(nlogn)

**试题分析**

本题考查算法分析与设计基础知识。  
　　根据题中给出的递归定义式进行推导，可得T（n）＝n+n-1+…+2+1，因此时间复杂度为O（n2）。

**试题答案**

（64）B

# 试题116(2010年上半年试题65)

若对一个链表最常用的操作是在末尾插入结点和删除尾结点，则采用仅设尾指针的单向循环链表（不含头结点）时，（）。

（65）A．插入和删除操作的时间复杂度都为O(1)  
B.插入和删除操作的时间复杂度都为O(n)  
C.插入操作的时间复杂度为O(1)，删除操作的时间复杂度为O(n)  
D.插入操作的时间复杂度为O(n)，删除操作的时间复杂度为O(1)

**试题分析**

本题考查数据结构与算法基础知识。  
　　设尾指针的单向循环链表（不含头结点）如下图所示：

IMG_379

　　设结点的指针域为next，新结点的指针为s，则在尾指针所指结点后插入结点的操作为：  
　　　　　　　　　　　　s->next＝t->next；t-＞next＝s；t＝s；  
　　也就是插入操作的时间复杂度为O(1)。  
　　要删除尾指针所指结点，必须通过遍历操作找到尾结点的前驱结点.其操作序列如下：  
　　　　　　If(t->next==t)free(t)；  
　　　　　　else｛  
　　　　　　p=t->next；  
　　　　　　while(P->next！=t)  
　　　　　　p=p->next；  
　　　　　　p->next=t->next;  
　　　　　　free(t);  
　　　　　　t=p；  
　　　　　　}  
　　也就是说，删除操作的时间复杂度为O(n)。

**试题答案**

（65）C