6.0元 数据结构

重庆邮电大学 2014-2015 学年第 1 学期

数据结构课程试卷（期末）（A 卷）（闭卷）

题　号	一	二	三	四	总 分
得　分					
评卷人					

注意：请仔细阅读题目，按要求答题，答案一律写在后面的答题纸上，并请保持字迹清楚，容易阅读。

一、选择题（本大题共 18 小题，每小题 2 分，共 36 分）

1. 一个栈的输入序列为 1 2 3 4 5，则下列序列中是栈的输出序列的是（　　）。

A. 5 4 1 3 2　　B. 5 4 3 2 1　　C. 3 1 2 4 5　　D. 1 4 2 5 3

2. 判定一个循环队列 Q（有 m 个位置）为满队列的条件是（　　）。

A. Q.front == Q.rear　　B. Q.front != Q.rear

C. Q.front == (Q.rear+1)%m　　D. Q.front != (Q.rear+1)%m

3. 高度为 k 的完全二叉树至少有（　　）个结点。（空树高度为 0）

A. 2^{k-1}　　B. 2^k　　C. 2^k-1　　D. k

4. 在解决计算机主机与打印机之间速度不匹配问题时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，而打印机则依相同次序从该缓冲区中取出数据打印。该缓冲区作为数据结构是一个（　）结构。

A. 栈　　B. 队列　　C. 哈希表（Hash Table）　　D. 线性表

5. 在线性表的下列运算中，不改变数据元素之间结构关系的运算是（　　）

A. 插入　　B. 删除

C. 排序　　D. 查找

6. 下面程序段的时间复杂度是（　　）。

```
for ( i=0; i<m; i++ )
    for ( j=1; j<n; j++ )
        A[i][j]=0;
```

A. O(n)　　B. O(m+n+1)　　C. O(m+n)　　D. O(m*n)

7. 计算机算法指的是解决问题的有限运算序列，它必须具备输入、输出和（　　）等 5 个特性。

A. 可执行性、可移植性和可扩充性　　B. 可行性、确定性和有穷性

C. 确定性、有穷性和稳定性　　D. 易读性、稳定性和安全性

8. 链表不具有的特点是（　　）。

A.可随机访问任一元素　　B.插入、删除不需要移动元素

C.不必事先估计存储空间　　D.所需空间与线性表长度成正比

9. 数组 A 中，每个元素需占用 2 字节，行下标从 0 到 7，列下标从 0 到 9，从首地址开始连续存放在存储器中，则存放该数组至少需要的存储字节数是（　　）

A. 63　　B. 80

C. 126　　D. 160

10. 折半查找 (Binary Search)要求查找表中各元素的关键字值必须是（　　）排列。

A. 递增或递减　　B.递增

C.递减　　D.无序

11. 一棵含有 16 个结点的二叉树的高度至少为（　　）。(空树高度为 0)

A. 3　　B. 4

C. 5　　D. 6

12. 一个具有 n 个顶点的无向图最多有（　　）条边。

A. n　　B. n(n-1)/2

C. n(n-1)　　D. 2n

13. 时间复杂度不受待排序序列初始状态的影响，总是 $O(n^2)$的是(　　)。

A 简单选择排序　B 直接插入排序

C 归并排序　　D 快速排序

14. 下列序列中，(　　) 是执行第一趟快速排序后得到的序列（排序的关键字类型是字符串）。

A.[da,ax,eb,de,bb] ff [ha,gc]　　B.[cd,eb,ax,da] ff [ha,gc,bb]

C.[gc,ax,eb,cd,bb] ff [da,ha]　　D.[ax,bb,cd,da] ff [eb,gc,ha]

15. 设结点 x 和结点 y 是二叉树 T 中的任意两个结点，若在先根序列中 x 在 y 之前，而在后根序列中 x 在 y 之后，则 x 和 y 的关系是（　　）

A. x 是 y 的左兄弟　　B. x 是 y 的右兄弟

C. x 是 y 的祖先　　D. x 是 y 的后代

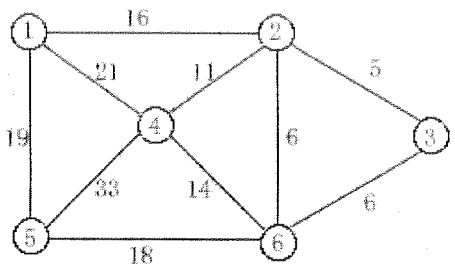
16. 应用 Dijkstra's 算法 (单源最短路径)求 5 号顶点到 3 号顶点的最短路径，最先求出的最短路径是（ ？ ）

A.　5 号顶点到 1 号顶点

B.　5 号顶点到 2 号顶点

C.　5 号顶点到 4 号顶点

D.　5 号顶点到 6 号顶点



17. 有 n 个球队参加的某联赛按单循环方式进行比赛，那么共需要进行(　　) 场比赛。

A. n(n-1) /2　　B.　n

C. n(n-1)　　D.　n+1

18. 设高度为 h 的二叉树上只有度为 0 和度为 2 的结点，则此类二叉树中所包含的结点数至少为（　）。

A.2h　　B.2h-1

C.2h+1　　D.h+1

二、填空题（本大题共 10 小题，每小题 2 分，共 20 分）

1. 设栈 S 和队列 Q 的初始状态皆为空，元素 a1，a2，a3，a4，a5 和 a6 依次通过一个栈，一个元素出栈后即进入队列 Q，若 6 个元素出队列的顺序是 a4，a5，a3，a6，a1，a2 则栈 S 至少应该容纳（　　）个元素。
2. 设一循环队列 Q 中，rear 指针指向队尾元素的下一个位置，front 指针指向队首元素，则判断队列中元素为空的条件是（　　）。
3. 采用特殊字符作为结尾的顺序存储方式存储，则串“My_year2009new”最少需要（　　）个字符的存储空间。
4. 已知完全二叉树的第 4 层（根结点为第 1 层）总共只有 2 个结点，则其叶子结点数是（　　）。
5. 具有 20 个结点的完全二叉树的高度（空树高度为 0）为（　　）。
6. 在一棵二叉树中，度为 0 的结点的个数为 5，度为 1 的结点的个数为 3，则共有（　　）个结点。
7. 具有 n 个顶点的有向完全图有（　　）条边。
8. 若待排序序列已经按关键字基本有序，则（　　）排序方法快。
9. n 个顶点的连通图的邻接矩阵中至少有(　　)个非零元素。
10. 在长度为 n 顺序实现的线性表的第 i（0≤i≤n）个位置插入一个元素，需要后移（　　）个元素。

三、问答题（本大题共 6 小题，每小题 6 分，共 36 分）

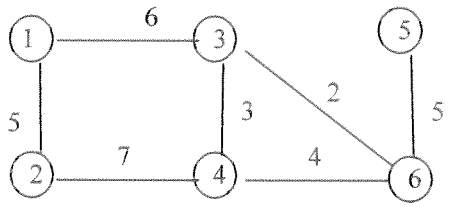
1. 已知初始序列（38，12，63，44，91，48），写出直接插入排序的各趟排序的结果。

2. 设 Hash 函数为 H（K）= K mod 7，哈希表的地址空间为 0，1，2，3，4，5，6，开始时哈希表为空，用二次探测法解决冲突，请画出依次插入键值 23，14，9，6，30，12 后的哈希表。

3. 已知某二叉树的先序遍历序列是 ABCEDFG，中序遍历序列是 CBEADFG，请写出其后序遍历序列。

4. 某森林的孩子－兄弟存储结构刚好构成一棵完全二叉树，二叉树按层次编号次序依次排列为A、B、C、D、E、F。请画出原来的森林。

5. 图G各顶点的连接关系及相应权值如下图所示。（1）画出图的邻接矩阵存储图示；（2）从顶点 2 开始对图进行广度优先遍历，写出遍历结果；



6. 使用Kruskal算法求上图的最小生成树，给出形成过程。

四、算法应用与实现题（本大题共 1 小题，每小题 8 分，共 8 分）

1. 某二叉树以二叉链表的方式存储，其结点的数据域为一个结构，其中包含一个为整数的子项。请编写一个函数，将所有结点的数据域中整数类型的那一个子项全部加上 100。（自定义数据类型）

重庆邮电大学《数据结构》试题

2013-2014 第 1 学期期末综合复习题及答案

（含金量高!!!如答案有误，请与leidj@cqupt.edu.cn联系）

一、单项选择题

(1) 一个算法应该是（ B ）。

A) 程序 B) 问题求解步骤的描述 C) 要满足五个基本属性 D) A 和 C

(2) 算法指的是（ D ）。

A) 计算机程序 B) 解决问题的计算方法

C) 排序算法 D) 解决问题的有限运算序列。

(3) 与数据元素本身的形式、内容、相对位置、个数无关的是数据的（ B ）。

A) 存储结构 B) 逻辑结构 C) 算法 D)操作

(4) 从逻辑上可以把数据结构分为（ C ）两大类。

A) 动态结构、静态结构 B) 顺序结构、链式结构

C) 线性结构、非线性结构 D) 初等结构、构造型结构

(5) 下列叙述中正确的是(D)。

A）一个逻辑数据结构只能有一种存储结构

B）数据的逻辑结构属于线性结构，存储结构属于非线性结构

C）一个逻辑数据结构可以有多种存储结构，且各种存储结构不影响数据处理的效率

D）一个逻辑数据结构可以有多种存储结构，且各种存储结构影响数据处理的效率

(6) 数据的基本单位是（ A ）

A) 数据项 B) 数据类型 C) 数据元素 D) 数据变量

(7) 下列程序的时间复杂度为（ A ）

```
i=0；s=0；
while（s<n）
{ i++；s=s+i；}
```

A) O（$\sqrt{n}$） B) O（$\sqrt{2n}$） C) O（n） D) O（n^2）

(8) 下列程序段的渐进时间复杂度为（ C ）。

```
for( int i=1;i<=n;i++)
    for( int j=1;j<= m; j++)
        A[i][j] = i*j ;
```

A）O(m2) B）O(n2) C）O(m*n) D）(m+n)

(9) 程序段如下：

```
sum=0；
for(i=1;i<=n;i++)
    for(j=1;j<=n;j++)
        sum++；
```

其中 n 为正整数，则最后一行的语句频度在最坏情况下是（ C ）。

A）O(n） B) O(nlogn) C) O(n^3) D) O(n^2)

(10) 在下面的程序段中，对 x 的赋值语句的频度为（ C ）。

```
for  ( i=1;  i>=n ;  i++)
        for  ( j=1;  j>=n ;  j++)
        x:=x+1;
```

A) O(2n) B)O(n) C) O(n^2) D) O($\log_2{}^n$)

(11) 程序段

```
for  ( i:=n-1;  i<=1;  i--）
    for  ( j:=1;  j>=i ;  j++)
        if  (a[j]>a[j+1] )
{  t=a[j];  a[j]= a[j+1];  a[j+1]= t;  }
```

其中 n 为正整数，则最后一行的语句频度在最坏情况下是（ D ）。

A) O(n） B) O(nlogn) C) $O(n^3)$ D) $O(n^2)$

(12) 设有一个递归算法如下：

```
int fact(int n)
{  /*  大于等于 0  */
    if ( n<=0 )  return  1  ;
    else  return  n*fact (n-1)  ;
}
```

则计算 fact(n)需要调用该函数的次数为（ B ）。

A) n B) n+1 C) n+2 D) n-1

(13) 下述程序段中语句①的频度是（ C ）。

```
s=0;
for(i=1;i<m;i++)
for(j=0;j<=i;j++)
        s+=j;
```

A) $\frac{(m+1)(m-1)}{2}$ B) $\frac{m(m-1)}{2}$ C) $\frac{(m+2)(m-1)}{2}$ D) $\frac{m(m+1)}{2}$

(14) 若某线性表中最常用的操作是在最后一个元素之后插入一个元素和删除第一个元素，则最节省运算时间的存储方式是（ D ）。

A)单链表 B)仅有头指针的单循环链表

C)双链表 D)仅有尾指针的单循环链表

(1) 求循环链表中当前结点的后继和前驱的时间复杂度分别是（ C ）。

A) O(n)和 O(1) B) O(1)和 O(1) C) O(1)和 O(n) D) O(n)和 O(n)

(15) 求单链表中当前结点的后继和前驱的时间复杂度分别是（ C ）。

A) O（n）和 O（1） B) O（1）和 O（1）

C) O（1）和 O（n） D) O(n)和 O（n）

(16) 非空的单循环链表的头指针为 head,尾指针为 rear，则下列条件成立的是（ A ）。

A) rear->next= =head B) rear->next->next= =head

C) head->next= =rear D) head->next->next= =rear

(17) 从一个长度为 n 的顺序表中删除第 i 个元素（1≤i≤n）时，需向前移动的元素的个数是（ A ）。

A)n-i B)n-i+1 C)n-i-1 D)i

(18) 已知一个有序表为（13，18，24，35，47，50，62，83，90，115，134），当二分检索值为 90 的元素时，检索成功需比较的次数是（ B ）。

A)1 B)2 C)3 D)4

(19) 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序存储，a[0][0]为第一个元素，其存储地址为 0，每个元素占有 1 个存储地址空间，则 a[4][5]的地址为（ A ）。

A) 13 B) 35 C) 17 D) 36

(20) 线性表采用链式存储时，节点的存储的地址（ B ）。

A) 必须是不连续的 B) 连续与否均可 C) 必须是连续的 D) 和头节点的存储地址相连续

(21) 用链表表示线性表的优点是（D ）。

A) 便于随机存取 B) 花费的存储空间比顺序表少

C) 数据元素的物理顺序与逻辑顺序相同 D) 便于插入与删除

(22) 链表不具有的特点是（ B ） 。

A) 插入、删除不需要移动元素 B) 可随机访问任一元素

C) 不必事先估计存储空间 D) 所需空间与线性长度成正比

(23) 在长度为 n 的顺序表中删除第 i 个元素(1≤i≤n)时，元素移动的次数为(D)。

A) n-i+1 B) i C) i+1 D) n-i

(24) 采用顺序搜索方法查找长度为 n 的顺序表示，搜索成功的平均搜索长度为（ D ）。

A) n B) n/2 C) (n-1)/2 D) (n+1)/2

(25) 将长度为 n 的单链表链接在长度为 m 的单链表之后的算法的时间复杂度为（ B ）。
A) O(1) B) O(n) C) O(m) D) O(m+n)
(26) 若不带头结点的单链表的头指针为 head，则该链表为空的判定条件是(A)。
A) head==NULL B) head->next==NULL C) head!=NULL D) head->next==head
(27) 某线性表中最常用的操作是在最后一个元素之后插入一个元素和删除第一个元素，则采用（ D ）存储方式最节省运算时间。
A) 单链表 B) 仅有头指针的单循环链表 C) 双链表 D) 仅有尾指针的单循环链表
(28) 若允许表达式内多种括号混合嵌套，则为检查表达式中括号是否正确配对的算法，通常选用的辅助结构是（ A ）。
A) 栈 B) 线性表 C) 队列 D) 二叉排序树
(29) 顺序栈 S 中 top 为栈顶指针，指向栈顶元素所在的位置，elem 为存放栈的数组，则元素 e 进栈操作的主要语句为（ D ）。
A) s.elem［top］=e； s.top=s.top+1； B) s.elem［top+1］=e；s.top=s.top+1；
C) s.top=s.top+1； s.elem［top+1］=e； D) s.top=s.top+1；s.elem［top］=e；
(30) 循环队列 sq 中，用数组 elem［0‥25］存放数据元素，sq.front 指示队头元素的前一个位置，sq.rear 指示队尾元素的当前位置，设当前 sq.front 为 20，sq.rear 为 12，则当前队列中的元素个数为（ C ）。
A) 8 B) 16 C) 17 D) 18
(31) 链式栈与顺序栈相比，一个比较明显的优点是（ B ）。
A) 插入操作更加方便 B) 通常不会出现栈满的情况
C) 不会出现栈空的情况 D) 删除操作更加方便
(32) 一个递归的定义可以用递归过程求解，也可以用非递归过程求解，但单从运行时间来看，通常递归过程比非递归过程（ B)。
A) 较快 B) 较慢 C) 相同 D) 不定
(33) 若已知一个栈的入栈序列是 1,2,3,4……n，其输出序列为 p1,p2,p3,……pn，若 p1= =n,则 pi 为(C)。
A) i B) n= =i C) n-i+1 D) 不确定
(34) 一个栈的入栈序列是 a,b,c,d,e，则栈的不可能的输出序列是(C) 。
A) edcba B) decba C) dceab D) abcde
(35) 若进栈序列为 1，2，3，4，5，6，且进栈和出栈可以穿插进行，则不可能出现的出栈序列是(D)。
A) 2，4，3，1，5，6 B) 3，2，4，1，6，5
C) 4，3，2，1，5，6 D) 2，3，5，1，6，4
(36) 对于栈操作数据的原则是（ B ）。
A) 先进先出 B) 后进先出 C) 后进后出 D) 不分顺序
(37) 栈和队列的共同点是（ C ）。
A) 都是先进先出 B) 都是先进后出
C) 只允许在端点处插入和删除元素 D) 没有共同点
(38) 一个队列的入队序列是 1，2，3，4，则队列的输出序列是（ B ）。
A) 4,3,2,1 B) 1,2,3,4 C)1,4,3,2 D) 3,2,4,1
(39) 设数组 data[m]作为循环队列 SQ 的存储空间，front 为队头指针，rear 为队尾指针，则执行出对操作后其头指针 front 值为（ D ）。
A) front=front+1 B) front=(front+1)%(m-1) C) front=(front-1)%m D) front=(front+1)%m
(40) 引起循环队列队头位置发生变化的操作是(A)。
A) 出队 B) 入队 C) 取队头元素 D) 取队尾元素
(2) 设以数组 A[m]存放循环队列的元素，其头尾指针分别为 front 和 rear，则当前队列中的元素个数为（ A ）。
A）(rear-front+m)%m B）rear-front+1 C）(front-rear+m)%m D）(rear-front)%m
(41) 二维数组 A[12][18]采用列优先的存储方法，若每个元素各占 3 个存储单元，且 A[0][0]地址为 150，则元素 A[9][7]的地址为(A)。
A) 429 B) 432 C) 435 D) 438
(42) 设有一个 10 阶的对称矩阵 A[10][10]，采用压缩方式按行将矩阵中下三角部分的元素存入一维数组

A) 32　　B) 33　　C) 41　　D) 65

(43)　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，则在 B 中确定 aij（i<j）的位置 k 的关系为(　A　)。

A) i*(i-1)/2+j　　B) j*(j-1)/2+i　　C) i*(i+1)/2+j　　D) j*(j+1)/2+i

(44)　对稀疏矩阵进行压缩存储目的是（　C　）。

A)　便于进行矩阵运算　　B)　便于输入和输出　C)　节省存储空间　　D)　降低运算的时间复杂度

(45)　对广义表 L=((a,b),(c,d),(e,f))执行操作 tail(tail(L))的结果是(B)。

A) (e,f)　　B) ((e,f))　　C) (f)　　D) ( )

(46)　设广义表 L=（(a,b,c)），则 L 的长度和深度分别为（C　　）。

A) 1 和 1　　B) 1 和 3　　C) 1 和 2　　D) 2 和 3

(47)　树中所有结点的度之和等于所有结点数加（　C　）。

A) 0　　B)1　　C) -1　　D) 2

(48)　在一棵具有 n 个结点的二叉链表中，所有结点的空域个数等于（C　　）。

A) n　　B) n-1　　C) n+1　　D) 2*n

(49)　某二叉树的先序序列和后序序列正好相反，则该二叉树一定是（B　　）的二叉树。

A)　空或只有一个结点　　B)　高度等于其节点数 C)　任一结点无左孩子　　D)　任一结点无右孩子

(50)　含有 10 个结点的二叉树中，度为 0 的结点数为 4，则度为 2 的结点数为（A　　）

A)3　　B)4　　C)5　　D)6

(51)　除第一层外，满二叉树中每一层结点个数是上一层结点个数的（　　C　）

A)1/2 倍　　B)1 倍　　C) 2 倍　　D) 3 倍

(52)　对一棵有 100 个结点的完全二叉树按层编号，则编号为 49 的结点，它的父结点的编号为（　A　　）

A)24　　B)25　　C)98　　D)99

(53)　可以惟一地转化成一棵一般树的二叉树的特点是（　　B　）

A)根结点无左孩子　　B)根结点无右孩子　　C)根结点有两个孩子　D)根结点没有孩子

(54)　设高度为 h 的二叉树上只有度为 0 和度为 2 的结点，则此类二叉树中所包含的结点数至少为(　B　)。

A) 2h　　B) 2h-1　　C) 2h+1　　D) h+1

(55)　在一棵度为 3 的树中，度为 3 的节点个数为 2，度为 2 的节点个数为 1，则度为 0 的节点个数为(　C　)。

A) 4　　B) 5　　C) 6　　D) 7

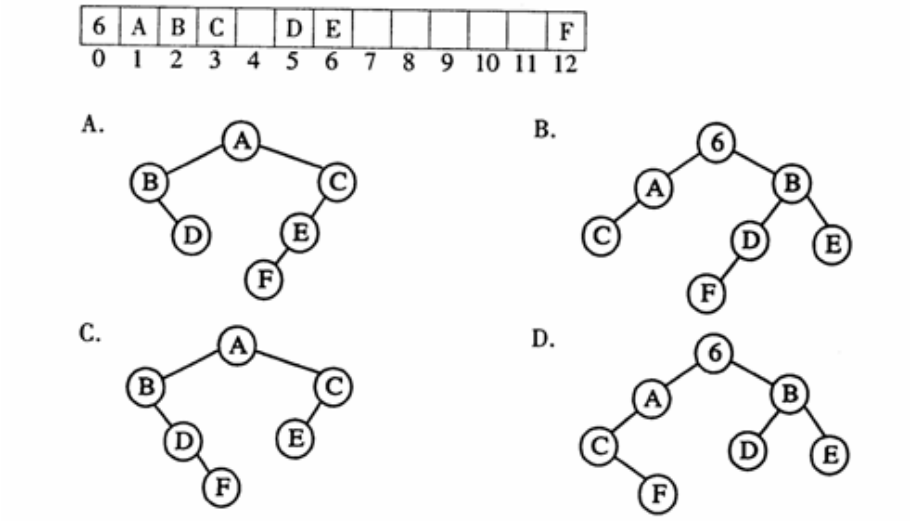
(56)　设森林 F 对应的二叉树为 B，它有 m 个结点，B 的根为 p，p 的右子树结点个数为 n,森林 F 中第一棵　子树的结点个数是（　A　）。

A)m-n　　B)m-n-1　　C) n+1　　D)　条件不足，无法确定

(57)　将一株有 100 个节点的完全二叉树从上到下，从左到右依次进行编号，根节点的编号为 1，则编号为 49 的节点的 左孩子编号为 A（）。

A) 98　　B) 89　　C) 50　　D)　没有孩子

(58)　下列图示的顺序存储结构表示的二叉树是(A)



(59) 树最适合用来表示（ C ）。

A) 有序数据元素　　B) 无序数据元素

C) 元素之间具有分支层次关系的数据　　D) 元素之间无联系的数据

(60) 在一个非空二叉树的中序遍历序列中，根结点的右边（ A ）。

A) 只有右子树上的所有结点　　B) 只有右子树上的部分结点

C) 只有左子树的上的部分结点　　D) 只有左子树上的所有结点

(61) 任何一棵二叉树的叶结点在先序、中序和后序遍历序列中相对次序（D ）。

A) 不发生改变　　B) 发生改变　　C) 不能确定　　D) 以上都不对

(62) 在有 n 个叶子结点的哈夫曼树中，其结点总数为（ D ）。

A) 不确定　　B) 2n　　C) 2n+1　　D) 2n-1

(63) 权值为{1,2,6,8}的四个结点构成的哈夫曼树的带权路径长度是（ D ）。

A) 18　　B) 28　　C) 19　　D) 29

(64) 对一个满二叉树，m 个树叶，k 个分枝结点，n 个结点，则（ D ）。

A) n=m+1　　B) m+1=2n　　C) m=k-1　　D) n=2k+1

(65) 在含有 n 个顶点和 e 条边的无向图的邻接矩阵中，零元素的个数为（ D ）。

A) e　　B) 2e　　C) n^2-e　　D) n^2-2e

(66) 若采用邻接矩阵翻存储一个 n 个顶点的无向图，则该邻接矩阵是一个（D ）。

A) 上三角矩阵　　B) 稀疏矩阵　　C) 对角矩阵　　D) 对称矩阵

(67) 在一个图中，所有顶点的度数之和等于所有边数的（ C ）倍。

A) 1/2　　B) 1　　C) 2　　D) 4

(68) 在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和的（ B ）倍。

A) 1/2　　B) 1　　C) 2　　D) 4

(69) 对于含 n 个顶点和 e 条边的图，采用邻接矩阵表示的空间复杂度为（ C ）。

A) O（n）　　B) O(e)　　C) O(n+e)　　D) O(n^2)

(70) 如果求一个连通图中以某个顶点为根的高度最小的生成树，应采用（ B ）。

A) 深度优先搜索算法　　B) 广度优先搜索算法

C) 求最小生成树的 prim 算法　　D) 拓扑排序算法

(71) n 个顶点的连通图至少中含有(A)。

A) n-1　　B) n　　C) n+1　　D) 0

(72) n 个顶点的完全有向图中含有(D)。

A) n-1 条有向边　　B) n 条有向边　　C) n(n-1)/2 条有向边　D) n(n-1)条有向边

(73) 假设一个有 n 个顶点和 e 条弧的有向图用邻接表表示，则删除预某个顶点 vi 相关的所有弧的时间复杂度是（ C ）。

A) O(n)　　B) O(e)　　C) O(n+e)　　D) O(n*e)

(74) 在无向图中定义顶点 Vi 域 Vj 之间的路径为从 Vi 到达 Vj 的一个（ A ）。

A) 顶点序列　　B) 边序列　　C) 权值总和　　D) 边的条数

(75) 无向图 G=(V,E),其中：V={a,b,c,d,e,f}, E={(a,b),(a,e),(a,c),(b,e),(c,f), (f,d),(e,d)}，对该图进行深度优先遍历，得到的顶点序列正确的是（ D ）。

A) a,b,e,c,d,f　　B) a,c,f,e,b,d　　C) a,e,b,c,f,d　　D) a,e,d,f,c,b

(76) 下面哪一方法可以判断出一个有向图是否有环（回路）。B

A) 求节点的度　　B) 拓扑排序　　C) 求最短路径　　D) 求关键路径

(77) 图的广度优先搜索类似于树的（ B ）次序遍历。

A) 先根　　B) 中根　　C) 后根　　D) 层次

(78) 在图采用邻接表存储时，求最小生成树的 Prim 算法的时间复杂度为(B)。

A) O(n)　　B) O(n+e)　　C) O(n^2)　　D) O(n^3)

(79) 已知有向图 G=(V,E)，其中 V={V1,V2,V3,V4,V5,V6,V7}，E={<V1,V2>,<V1,V3>,<V1,V4>,<V2,V5>,<V3,V5>,<V3,V6>,<V4,V6>,<V5,V7>,<V6,V7>},G 的拓扑序列是（A ）。

A) $V_1,V_3,V_4,V_6,V_2,V_5,V_7$　　B) $V_1,V_3,V_2,V_6,V_4,V_5,V_7$

C) $V_1,V_3,V_4,V_5,V_2,V_6,V_7$　　D) $V_1,V_2,V_5,V_3,V_4,V_6,V_7$

(80) 关键路径是事件结点网络中（ A ）。

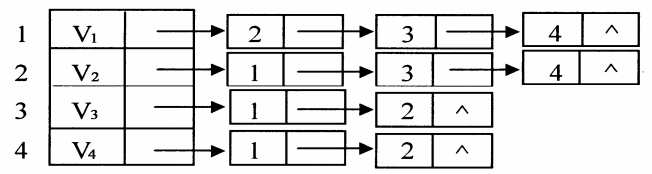
A) 从源点到汇点的最长路径　　B) 从源点到汇点的最短路径

C) 最长回路　　D) 最短回路

(81) 有 n 个结点的有向完全图的弧数是（ C ）。

A) n^2　　B) 2n　　C) n(n-1)　　D) 2n(n+1)

(82) 设图的邻接链表如题 12 图所示，则该图的边的数目是（ B ）。



83 题图

A) 4　　B) 5　　C) 10　　D) 20

(83) 在一个图中，所有顶点的度数之和等于图的边数的（ B ）倍。

A) 1/2　　B) 1　　C) 2　　D) 4

(84) 在一个有向图中，所有顶点的入度之和等于所有顶点的出度之和的（ B ）倍。

A) 1/2　　B) 1　　C) 2　　D) 4

(85) 有 8 个结点的无向图最多有（ B ）条边。

A) 14　　B) 28　　C) 56　　D) 112

(86) 有 8 个结点的无向连通图最少有（ C ）条边。

A) 5　　B) 6　　C) 7　　D) 8

(87) 有 8 个结点的有向完全图有（ C）条边。

A) 14　　B) 28　　C) 56　　D) 112

(88) 用邻接表表示图进行广度优先遍历时，通常是采用（ B ）来实现算法的。

A) 栈　　B) 队列　　C) 树　　D) 图

(89) 用邻接表表示图进行深度优先遍历时，通常是采用（ A ）来实现算法的。

A) 栈　　B) 队列　　C) 树　　D) 图

(90) 已知图的邻接矩阵，根据算法思想，则从顶点 0 出发按深度优先遍历的结点序列是（ C ）。

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

A) 0 2 4 3 1 5 6　　B)0 1 3 6 5 4 2　　C) 0 4 2 3 1 6 5　　D) 0 3 6 1 5 4 2

建议：0 1 3 4 2 5 6（根据序号小的优先结果）

(91) 已知图的邻接矩阵同上题，根据算法，则从顶点 0 出发，按深度优先遍历的结点序列是（ D ）。

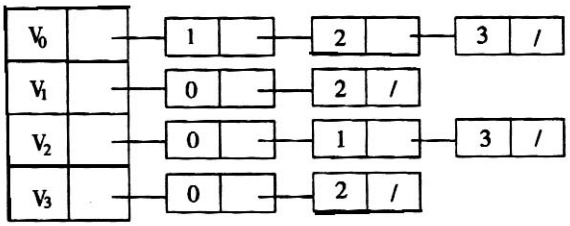
A) 0 2 4 3 1 5 6　　B) 0 1 3 5 6 4 2　　C) 0 4 2 3 1 6 5　　D) 0 1 3 4 2 5 6

(92) 已知图的邻接矩阵同上题，根据算法，则从顶点 0 出发，按广度优先遍历的结点序列是（ B ）。

A) 0 2 4 3 6 5 1　　B) 0 1 3 6 4 2 5　　C) 0 4 2 3 1 5 6　　D) 0 1 3 4 2 5 6

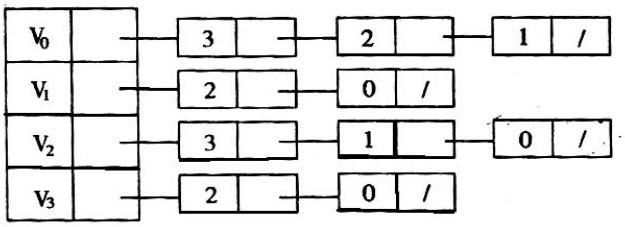
（建议：0 1 2 3 4 5 6）（根据序号小的优先结果）

(93) 已知图的邻接表如下所示，根据算法，则从顶点 0 出发按深度优先遍历的结点序列是（ D ）。



A) 1 3 2　　B) 0 2 3 1　　C) 0 3 2 1　　D) 0 1 2 3

(94) 已知图的邻接表如下所示，根据算法，则从顶点 0 出发按广度优先遍历的结点序列是（ A ）。



A) 0 3 2 1　　B) 0 1 2 3　　C) 0 1 3 2　　D) 0 3 1 2

(95) 深度优先遍历类似于二叉树的（ A ）。

A) 先序遍历　　B) 中序遍历　　C) 后序遍历　　D) 层次遍历

(96) 广度优先遍历类似于二叉树的（ D ）。

A) 先序遍历　　B) 中序遍历　　C) 后序遍历　　D) 层次遍历

(97) 任何一个无向连通图的最小生成树（ B ）。

A) 只有一棵　　B) 一棵或多棵　　C) 一定有多棵　　D) 可能不存在

(98) 在分析折半查找的性能时常常加入失败节点，即外节点，从而形成扩充的二叉树。若设失败节点 i 所在层次为 Li，那么查找失败到达失败点时所做的数据比较次数是（ D ）。

A) Li+1　　B) Li+2　　C) Li-1　　D) Li

(99) 向一个有 127 个元素原顺序表中插入一个新元素并保存原来顺序不变，平均要移动（B ）个元素。

A) 8　　B) 63.5　　C) 63　　D) 7

(100) 由同一组关键字集合构造的各棵二叉排序树(B)。

A) 其形态不一定相同，但平均查找长度相同

B) 其形态不一定相同，平均查找长度也不一定相同

C) 其形态均相同，但平均查找长度不一定相同

D) 其形态均相同，平均查找长度也都相同

(101) 衡量查找算法效率的主要标准是（C ）。

A) 元素的个数　　B) 所需的存储量　　C) 平均查找长度　　D) 算法难易程度

(102) 适合对动态查找表进行高效率查找的组织结构是（C ）。

A) 有序表　　B) 分块有序表　　C) 二叉排序树　　D) 快速排序

(3) 能进行二分查找的线性表,必须以（A ）。

A) 顺序方式存储，且元素按关键字有序

B) 链式方式存储，且元素按关键字有序

C) 顺序方式存储，且元素按关键字分块有序

D) 链式方式存储，且元素按关键字分块有序

(103) 为使平均查找长度达到最小,当由关键字集合{05,11,21,25,37,40,41,62,84}构建二叉排序树时,第一个插入的关键字应为（ B ）

A) 5　　B)37　　C) 41　　D) 62

(104) 用某种排序方法对关键字序列{35,84,21,47,15,27,68,25,20}进行排序时，序列的变化情况如下：

20,15,21,25,47,27,68,35,84

15,20,21,25,35,27,47,68,84

15,20,21,25,27,35,47,68,84

则采用的方法是（ D ）。

A) 直接选择排序　　B) 希尔排序　　C) 堆排序　　D) 快速排序

(105) 一组记录的排序码为(46,79,56,38,40,84)，则利用快速排序的方法，以第一个记录为基准得到的第一次划分结果为（ C ）。

A) 38,40,46,56,79,84　　B) 40,38,46,79,56,84　　C) 40,38,46,56,79,84　　D) 40,38,46,84,56,79

(106) 快速排序在最坏情况下的时间复杂度是（ B ）

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

(107) 下列排序算法中不稳定的是（　D　）。

A) 直接选择排序　　B) 折半插入排序　　C) 冒泡排序　　D) 快速排序

(108) 对待排序的元素序列进行划分，将其分为左、右两个子序列，再对两个子序列进行同样的排序操作，直到子序列为空或只剩下一个元素为止。这样的排序方法是（　C　）。

A) 直接选择排序　　B) 直接插入排序　　C) 快速排序　　D) 冒泡排序

(109) 将 5 个不同的数据进行排序，至多需要比较（ C　）次。

A)　8　　B)　9　　C)　10　　D)　25

(110) 排序算法中，第一趟排序后，任一元素都不能确定其最终位置的算法是（　D　　）。

A)选择排序　　B)快速排序　　C)冒泡排序　　D)插入排序

(111) 排序算法中，不稳定的排序是（　　C　）。

A)直接插入排序　　B)冒泡排序　　C)堆排序　　D)选择排序

(112) 排序方法中，从未排序序列中依次取出元素与已排序序列（初始时为空）中的元素进行比较，将其放入已排序序列的正确位置上的方法，称为（　C ）.

A)　希尔排序　　B)　冒泡排序　　C)　插入排序　　D)　选择排序

(113) 从未排序序列中挑选元素，并将其依次插入已排序序列（初始时为空）的一端的方法，称为（ D　　）。

A)　希尔排序　　B)　归并排序　　C)　插入排序　　D)　选择排序

(114) 对 n 个不同的排序码进行冒泡排序，在下列哪种情况下比较的次数最多。（ B　　）

A)　从小到大排列好的　　B)　从大到小排列好的

C)　元素无序　　D)　元素基本有序

(115) 对 n 个不同的排序码进行冒泡排序，在元素无序的情况下比较的次数为（　D　）。

A)　n+1　　B)　n　　C)　n-1　　D)　n(n-1)/2

(116) 快速排序在下列哪种情况下最易发挥其长处。（ C　　）

A)　被排序的数据中含有多个相同排序码

B)　被排序的数据已基本有序

C)　被排序的数据完全无序

D)　被排序的数据中的最大值和最小值相差悬殊

(117) 对有 n 个记录的表作快速排序，在最坏情况下，算法的时间复杂度是（ B　　）。

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

(118) 若一组记录的排序码为（46, 79, 56, 38, 40, 84），则利用快速排序的方法，以第一个记录为基准得到的一次划分结果为（　C　）。

A)　　38,　40,　46,　56,　79,　84　　B)　　40, 38,　46 ,　79,　56,　84

C)　　40,　38，46,　56,　79,　84　　D)　　40,　38, 46,　84,　56,　79

(119) 快速排序方法在（　D ）情况下最不利于发挥其长处。

A) 要排序的数据量太大　　B) 要排序的数据中含有多个相同值

C) 要排序的数据个数为奇数　　D) 要排序的数据已基本有序

(120) 对关键码序列 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)26(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)

(121) 对下列关键字序列用快速排序法进行排序时，速度最快的情形是（　　）。

A) {21,25,5,17,9,23,30}　　B){25,23,30,17,21,5,9}

C) {21,9,17,30,25,23,5}　　D) {5,9,17,21,23,25,30}

二、填空题

(122)数据结构是一门研究非数值计算的程序设计问题中计算机的 操作对象 以及它们之间的 关系 和运算等的学科。

(123)数据结构被形式地定义为（D, R），其中D是 数据元素 的有限集合，R是D上的 关系 有限集合。

(124)数据结构包括数据的 逻辑结构 、数据的 存储结构 和数据的 运算 这三个方面的内容。

(125)数据结构按逻辑结构可分为两大类，它们分别是 线性结构 和 非线性结构 。

(126)线性结构中元素之间存在一对一关系，树形结构中元素之间存在一对多关系，图形结构中元素之间存在多对多关系。

(127)在线性结构中，第一个结点 没有 前驱结点，其余每个结点有且只有 1 个前驱结点；最后一个结点 没有 后续结点，其余每个结点有且只有 1 个后续结点。

(128)在树形结构中，树根结点没有 前驱 结点，其余每个结点有且只有 1 个前驱结点；叶子结点没有 后续 结点，其余每个结点的后续结点数可以任意多个 。

(129)在图形结构中，每个结点的前驱结点数和后续结点数可以 任意多个 。

(130)数据的存储结构可用四种基本的存储方法表示，它们分别是顺序 、 链式 、 索引 和 散列 。

(131)数据的运算最常用的有 5 种，它们分别是插入 、 删除、修改、 查找 、排序。

(132)一个算法的效率可分为 时间 效率和 空间 效率。

(133)对于给定的n个元素,可以构造出的逻辑结构有集合，线性表，树，图四种。

(134)顺序映象的特点是借助元素在存储器中的相对位置来表示数据元素之间的逻辑关系。非顺序映象的特点是借助是指示元素存储地址的 指针表示数据元素之间的逻辑关系。任何一个算法的设计取决于选定逻辑结构，而算法的实现依赖于采用的存储结构。

(135)数据类型是一组 性质相同 的值集合以及定义在这个值集合上的一组操作的总称。

(136)数据对象是性质相同的数据元素的集合，是数据的一个子集。

(137)如果操作不改变原逻辑结构的“值”，而只是从中提取某些信息作为运算结果，则称该类运算为 引用 型运算。

(138)算法的 健壮特性是指做为一个好的算法，当输入的数据非法时，也能适当地做出正确反应或进行相应的处理，而不会产生一些莫名其妙的输出结果。

(139)算法分析不是针对实际执行时间的精确的算出算法执行具体时间的分析，而是针对算法中语句的 执行次数做出估计，从中得到算法执行时间的信息。

(140)T(n)=O(f(n))，它表示随问题规模n的增大算法的执行时间的增长率和f(n)的增长率 相同，称作算法的渐进时间复杂度，简称时间复杂度。

(141)若算法执行时所需要的辅助空间相对于输入数据量而言是个常数，则称这个算法为 原地工作，辅助空间为O(1)。

(142)在带有头结点的单链表中L中，第一个元素结点的指针是 L->next 。

(143)在一个带头节点的单循环链表中，p指向尾结点的直接前驱，则指向头结点的指针head可用p表示为head= p->next->next 。

(144)设单链表的结点结构为(data,next)，next 为指针域，已知指针 px 指向单链表中 data 为 x 的结点，指针 py 指向 data 为 y 的新结点 ，若将结点 y 插入结点 x 之后，则需要执行以下语句: py->next=px->next; px->next=py。

(145) 对于栈操作数据的原则是 后进先出 。

(146) 设以数组A[m]存放循环队列的元素，其头尾指针分别为front和rear，则当前队列中的元素个数为 (rear-front+m)%m 。

(147)若已知一个栈的入栈序列是 1,2,3,4……n，其输出序列为p1,p2,p3,……pn，若p1= =n,则pi为 n-i+1 。

(148) **队列** 是被限定为只能在表的一端进行插入运算，在表的另一端进行删除运算的线性表。

(149)通常程序在调用另一个程序时，都需要使用一个 栈 来保存被调用程序内分配的局部变量。形式参数的存储空间以及返回地址。

(150)栈下溢是指在 栈空 时进行出栈操作。

(151)用P表示入栈操作，D表示出栈操作，若元素入栈的顺序为 1234，为了得到 1342 出栈顺序，相应的P和D的操作串为 PDPPDPDD 。

(152)在具有n个单元的循环队列中，队满共有 n-1 个元素。

(153) **队列** 是被限定为只能在表的一端进行插入运算，在表的另一端进行删除运算的线性表。

(154)循环队列的引入，目的是为了克服 假溢出 。

(155)所谓稀疏矩阵指的是 非零元 很少(t<<m*n)且分布没有规律 。

(156)在稀疏矩阵表示所对应的三元组线性表中，每个三元组元素按 行 为主序， 列号为辅序的次序排列。

(157)二位数组$A_{m\times n}$按行优先顺序存储在内存中，元素a_{00}地址为$loc(a_{00})$,每个元素在内存中占d个字节，元素a_{ij}的地址计算公式为$loc(a_{ij})$= $loc(a_{00})+(i*n+j)*d$ 。

(158)去除广义表LS=(a1,a2,a3,……，an)中第 1 个元素，由其余元素构成的广义表称为LS的 表尾 。

(159)树内个结点的度 最大值 称为树的度。

(160)一个二叉树第 5 层节点最多有 16 个。

(161)已知完全二叉树T的第 5 层只有 7 个结点，则该树共有 11 个叶子结点。

(162)在一棵二叉树中，度为零的结点的个数为N0,度为 2 的结点的个数为N2,则有N0 = N2+1 。

(163) 假设用于通信的电文由 8 个字母组成，其频率分别为 7,19,2,6,32,3,27,10。设计哈夫曼编码，其中字母的编码长度最大是 5 位。

(164) 一棵具有 257 个结点的完全二叉树，它的深度为 9 。

(165) 图的深度优先遍历序列 不是 惟一的。

(166) 在图中，任何两个结点之间都可能存在关系，因此图的数据元素之间时一种 多对多 的关系。

(167) 在有向图中，以顶点v为终点的边的数目称为v的 入度 。

(168) 一个无向图有n个顶点，e条边，则所以顶点的度数之和为 2e 。

(169) 图有 邻接矩阵 、 邻接表 等存储结构，遍历图有 深度优先遍历 、 广度优先遍历 等方法。

(170)

(171) 有向图G用邻接表矩阵存储，其第i行的所有非零元素之和等于顶点i的 出度 。

(172) 如果n个顶点的图是一个环，则它有 n 棵生成树。（以任意一顶点为起点，得到n-1 条边）

(173) n个顶点e条边的图，若采用邻接矩阵存储，则空间复杂度为 O(n2) 。

(174) n个顶点e条边的图，若采用邻接表存储，则空间复杂度为 O(n+e) 。

(175) 设有一稀疏图G，则G采用 邻接表 存储较省空间。

(176) 设有一稠密图G，则G采用 邻接矩阵 存储较省空间。

(177) 图的逆邻接表存储结构只适用于 有向 图。

(178) 已知一个图的邻接矩阵表示，删除所有从第i个顶点出发的方法是 将邻接矩阵的第i行全部置0 。

(179) 图的深度优先遍历序列 不是 惟一的。

(180) n个顶点e条边的图采用邻接矩阵存储，深度优先遍历算法的时间复杂度为 O(n2) ；若采用邻接表存储时，该算法的时间复杂度为 O(n+e) 。

(181) n个顶点e条边的图采用邻接矩阵存储，广度优先遍历算法的时间复杂度为 O(n2)；若采用邻接表存储，该算法的时间复杂度为 O(n+e) 。

(182) 图的BFS生成树的树高比DFS生成树的树高 小或相等 。

(183) 用普里姆(Prim)算法求具有n个顶点e条边的图的最小生成树的时间复杂度为 O(n2)；用克鲁斯卡尔(Kruskal)算法的时间复杂度是 O(elog2e) 。

(184) 若要求一个稀疏图G的最小生成树，最好用 克鲁斯卡尔(Kruskal) 算法来求解。

(185) 若要求一个稠密图G的最小生成树，最好用 普里姆(Prim) 算法来求解。

(186) 用Dijkstra算法求某一顶点到其余各顶点间的最短路径是按路径长度 递增 的次序来得到最短路径的。

(187) 拓扑排序算法是通过重复选择具有 0 个前驱顶点的过程来完成的。

(188) 在各种查找方法中，平均查找长度与结点个数n无关的查找方法是 散列查找 。

(189) 散列法存储的基本思想是由 关键字的值 决定数据的存储地址。

(190) 大多数排序算法都有两个基本的操作： 比较和移动 。

(191) 由于查找算法的基本运算是关键字之间的比较操作，所以可用 平均查找长度 来衡量查找算法的性能。

(192) 查找有静态查找和动态查找，当查找不成功时动态查找会 将查找关键字插入在表中 。

(193) 顺序查找法中设置监视哨，可以起到 防止越界 的作用。

(194) 假设列表长度为n，那么查找第i个数据元素时需进行 n-i+1 次比较。

(195) 假设查找每个数据元素的概率相等，即 Pi=1/n，则顺序查找算法的平均查找长度为： ASL=(n+1)/2 。

(196) 折半查找法又称为二分法查找法，这种方法要求待查找的列表必须是 按关键字大小有序排列 的顺序表。

(197) 在有序表(12,24,36,48,60,72,84)中二分查找关键字 72 时所需进行的关键字比较次数为 2 。

(198) 折半查找有序表（4,6,12,20,28,38,50,70,88,100），若查找表中元素 20，它将依次与表中元素 28，6，12，20 比较大小。

(199) 在各种查找方法中，平均查找长度与结点个数n无关的查找方法是 散列查找 。

(200) 散列法存储的基本思想是由 关键字的值 决定数据的存储地址。

(201) 当关键字集合很大时，关键字值不同的元素可能会映象到哈希表的同一地址上，即 k1≠k2 ，但 H（k1）=H（k2），这种现象称为 冲突 .

(202) 产生冲突现象的两个关键字称为该散列函数的 同义词 。

(203) 在散列函数 H(key)=key MOD p 中，p应取 素数 。

(204) 设哈希表长m=14, 哈希函数H(key)=key MOD11.表中已有 4 个结点；addr(15)=4, addr(38)=5, addr(61)=6, addr(84)=7, 其余地址为空。如用二次探测再散列处理冲突，关键字为 49 的结点的地址是 9 。

(205) 希尔排序是属于 插入 排序的改进方法。

(206) 给出一组关键字T＝（20,4,34,5,16,33,18,29,2,40,7），要求从下到大进行排序，试给出快速排序（选一个记录为枢纽）第一趟排序结果 7,4,2,85,16,18,20,,29,33,40,34 。

(207) 大多数排序算法都有两个基本的操作： 比较和移动 。

(208) 在对一组记录（54,38,96,23,15,72,60,45,83）进行直接插入排序时，当把第 7 个记录 60 插入到有序表时，为寻找插入位置至少需比较 6 次。

(209) 在插入和选择排序中，若初始数据基本正序，则选用 插入 ；若初始数据基本反序，则选用 选择 。

(210) 在堆排序和快速排序中，若初始记录接近正序或反序，则选用 堆排序 ；若初始记录基本无序，则最好选用 快速排序 。

(211) 对于n个记录的集合进行冒泡排序，在最坏的情况下所需要的时间是 $O(n^2)$ 。若对其进行快速排序，在最坏的情况下所需要的时间是 $O(n^2)$

（222）对于n个记录的集合进行归并排序，所需要的平均时间是 O(nlog2n) ，所需要的附加空间是 O(n) 。

(223) 对于n个记录的表进行 2 路归并排序，整个归并排序需进行 ⌈log2n⌉ 趟（遍）。

(224) 设要将序列（Q, H, C, Y, P, A, M, S, R, D, F, X）中的关键码按字母序的升序重新排列，则：
冒泡排序一趟扫描的结果是 H C Q P A M S R D F X Y ；
快速排序一趟扫描的结果是 F H C D P A M Q R S Y X ；

(225) 在堆排序、快速排序和归并排序中，
若只从存储空间考虑，则应首先选取 方法，其次选取 快速排序方法，最后选取归并排序方法；
若只从排序结果的稳定性考虑，则应 选取 归并排序 方法；
若只从平均情况下最快考虑，则应选取 堆排序、快速排序和归并排序 方法；
若只从最坏情况下最快并且要节省内存考虑，则应选取 堆排序 方法。

三、程序填空题

(226) 以下程序的功能是实现带附加头结点的单链表数据结点逆序连接，请填空完善之。

```
void reverse(pointer h)
 /* h 为附加头结点指针；*/
```

```
{ pointer p,q;
    p=h->next;   h->next=NULL;
  while((1)_____)
    {q=p; p=p->next; q->next=h->next; h->next=(2)_____;    }
  }
```

(1)p!=null　　//链表未到尾就一直作

(2)q　　//将当前结点作为头结点后的第一元素结点插入

(227)下面是用 c 语言编写的对不带头结点的单链表进行就地逆置的算法，该算法用 L 返回逆置后的链表的头指针，试在空缺处填入适当的语句。

```
void  reverse（linklist &L）{
        p=null；q=L；
        while（q！=null）
        { (1)_____;    q->next=p；p=q；(2)___ ; }
          (3)_____;
       }
```

(1) L=L->next;　//暂存后继

(2)q=L;　　//待逆置结点

(3)L=p;　　//头指针仍为 L

(228)以下算法的功能是用头插法建立单链表的算法，请填空完善之。

```
Linklist  CreateFromHead(   )
{      LinkList   L;
       Node   *s;
       char     c;
       L=(Linklist)malloc(sizeof(Node));                    /*为头结点分配存储空间*/
       _____ L->next=NULL   ;
       While( (   c=getchar())   !='*'  )
       {    s=(Node*)malloc(sizeof(Node));                  /*为读入的字符分配存储空间*/
            s->data=c;
            _____ s->next=L->next   ;
            _____ L->next=s    ;
       }
       return L;
}
```

(229)以下算法的功能是尾插法创建链表，请填空完善之。

```
typedef struct Node   /*结点类型定义*/
{    char data;
     struct Node   * next;
}  Node, *LinkList;   /* LinkList 为结构指针类型*/
Linklist  CreateFromTail(   )                          /*将新增的字符追加到链表的末尾*/
{      LinkList L;
       Node *r, *s;
       char   c;
       L=(Node * )malloc(sizeof(Node));                  /*为头结点分配存储空间*/
       L->next=NULL;
       r=L;          /*r 指针指向链表的当前表尾，以便于做尾插入，其初值指向头结点*/
      while(  (  _____c=getchar()) !='$'   )                    /*当输入'$'
时，建表结束*/
       {     s=(Node*)malloc(sizeof(Node));    s->data=c;
             _____；r->next=s;    r=s;
```

```
        }
        _____;     r->next=NULL          /*将最后一个结点的 next 链域置为空，表示链表的结束*/
        return L;
    }   /*CreateFromTail*/
```

(230)下列算法在顺序表 L 中依次存放着线性表中的元素，在表中查找与 e 相等的元素，若 L.elem[i]=e, 则找到该元素，并返回 i+1，若找不到，则返回“-1” ，请填空完善之。

```
    int  Locate(SeqList L，int e)
    {     i=0 ;                    /*i 为扫描计数器，初值为 0，即从第一个元素开始比较*/
          while ((i<=L.last)&&(L.elem[i]!=e) )  _____  i++;
                                   /*顺序扫描表，直到找到值为 key 的元素,或扫描到表尾而没找到*/
          if  ( _____ i<=L.last  )      return(i+1);          /*若找到值为 e 的元素，则返回其序号*/
          else          return(-1);                          /*若没找到，则返回空序号*/
     }
```

(231)下列算法在顺序表 L 中第 i 个数据元素之前插入一个元素 e。 插入前表长 n=L->last+1，i 的合法取值范围是 1≤i≤L->last+2，请填空完善之。

```
    void    InsList(SeqList *L, int i,   int e)
    {   int k;
        if((i<1) || (i>L->last+2))     printf(“插入位置 i 值不合法”);
        if(L->last>=maxsize-1)      printf(“表已满无法插入”);
        for(k=L->last;k>=i-1;k--)          /*为插入元素而移动位置*/
                  _____L->elem[k+1]=L->elem[k] ;
        _____L->elem[i-1]=e ;      /*在 C 语言数组中，第 i 个元素的下标为 i-1*/
        _____L->last++ ;
    }
```

(232)下列算法是在顺序表 L 中删除第 i 个数据元素，并用指针参数 e 返回其值。i 的合法取值为 1≤i≤L.last+1，请填空完善之。

```
        int  DelList(SeqList *L, int i, int *e)
        {      int k;
               if((i<1)||(i>_____  L->last+1  ))     printf(“删除位置不合法！”);
                        _____*e= L->elem[i-1] ;       /*  将删除的元素存放到 e 所指向的变量中*/
               for(k=i;i<=L->last;k++)
_____ L->elem[k-1]= L->elem[k]  ;          /*将后面的元素依次前移*/
               _____ L->last--  ;
        }
```

四、解答题

(233)假设以数组 seqn［m］存放循环队列的元素，设变量 rear 和 quelen 分别指示循环队列中队尾元素的位置和元素的个数。

(1) 写出队满的条件表达式；

(2) 写出队空的条件表达式；

(3) 设 m=40,rear=13,quelen=19,求队头元素的位置；

(4) 写出一般情况下队头元素位置的表达式。

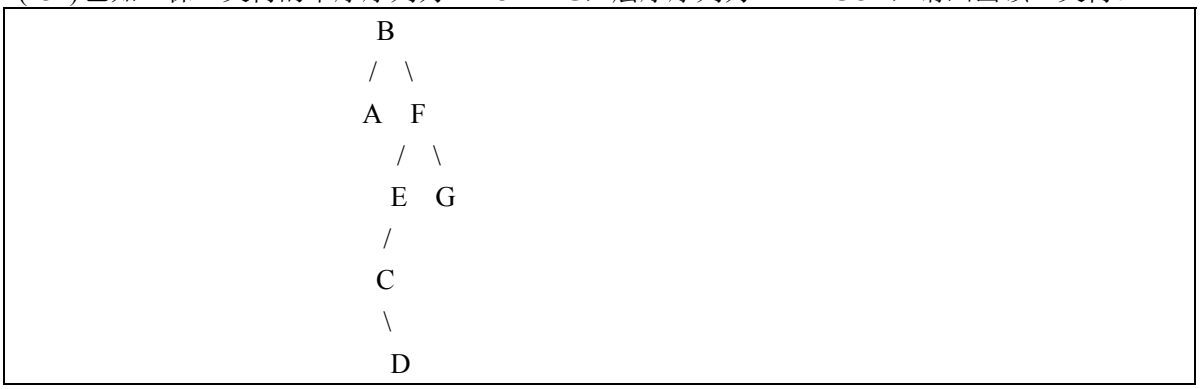
(1) quelen==m

(2) quelen == 0

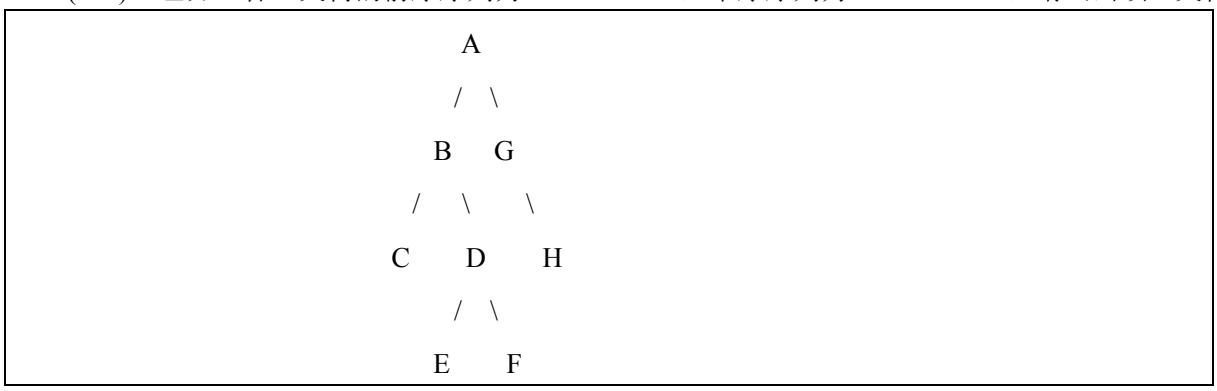
(3) (13 - 19 + 40) % 40 = 34

(4) (rear - quelen + m) % m

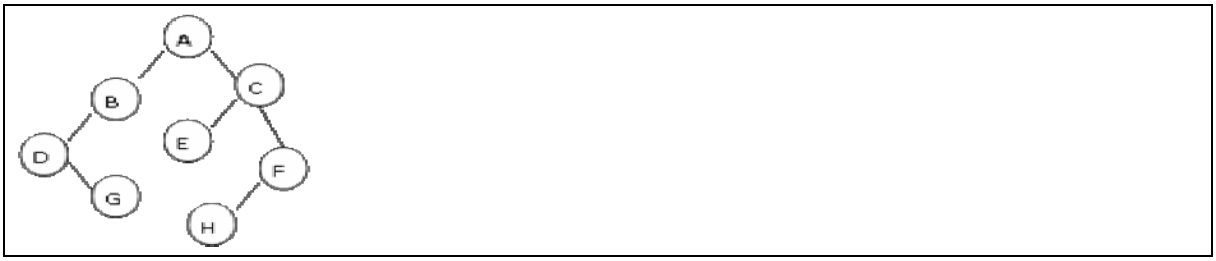
(234) 已知一棵二叉树的中序序列为 ABCDEFG，层序序列为 BAFEGCD，请画出该二叉树。



(235) 已知一棵二叉树的前序序列为 ABCDEFGH，中序序列为 CBEDFAGH，请画出该二叉树。



(236) 已知一棵二叉树如图所示。请分别写出按前序、中序、后序和层次遍历是得到的顶点序列。



前序：A,B,D,G,C,E,F,H

中序：D,G,B,A,E,C,H,F

后序：G,D,B,E,H,F,C,A

层次：A,B,C,D,E,F,G,H

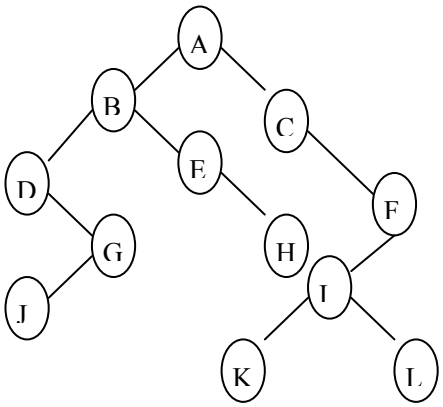
(237)

(238) 已知一棵二叉树的前序序列为：A,B,D,G,J,E,H,C,F,I,K,L 中序序列：D,J,G,B,E,H, A,C,K,I,L,F。

(1) 写出该二叉树的后序序列；

(2) 画出该二叉树；

(3) 求该二叉树的高度(假定空树的高度为－1)和度为 2、度为 1、及度为 0 的结点个数。



该二叉树的后序序列为：J,G,D,H,E,B,K,L,I,F,C,A。

该二叉树的形式如图所示：

该二叉树高度为：5。

度为 2 的结点的个数为：3。

度为 1 的结点的个数为：5。

度为 0 的结点个数为：4。

(239)有一份电文中共使用 6 个字符:a,b,c,d,e,f,它们的出现频率依次为 2,3,4,7,8,9，试构造一棵哈夫曼树，并求其加权路径长度 WPL，字符 c 的编码。

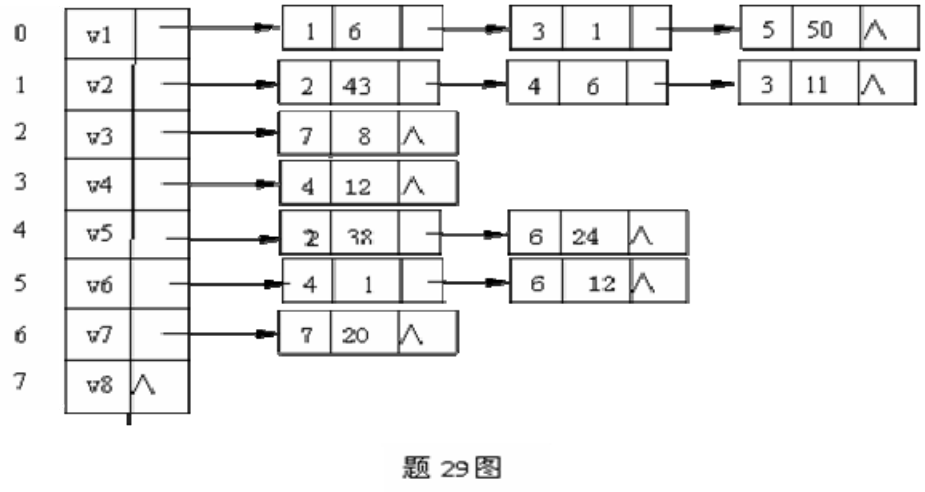
WPL＝80 字符 c：001（不唯一）

(240)下图是带权的有向图 G 的邻接表表示法。从结点 V1 出发，求出：

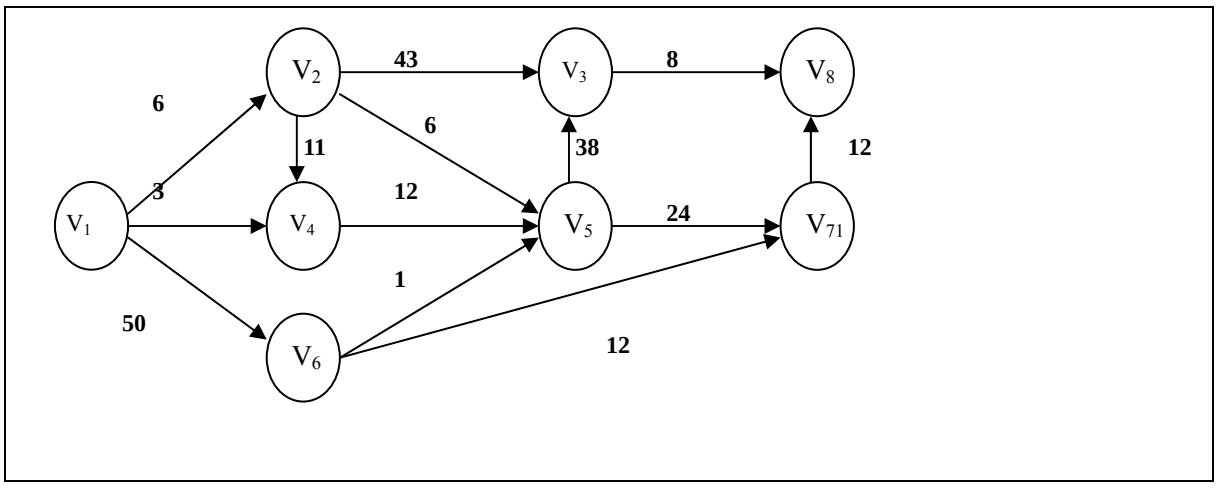
a) 深度遍历图 G；

b) G 的一个拓扑序列；

c) 从结点 V1 到结点 V8 的最短路径。



题 29 图



参考答案：v1,v2,v3,v8,v4,v5,v7,v6 (2 分)

v1,v2,v4,v6,v5,v3,v7,v8 (2 分)

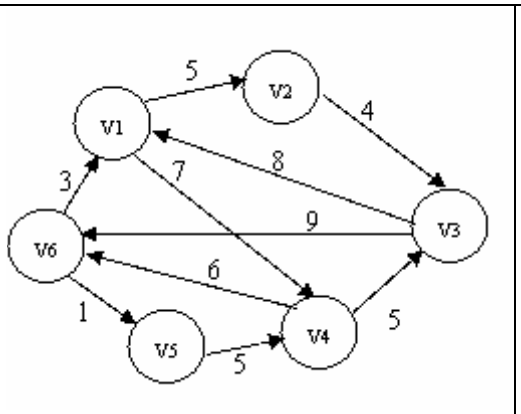
V1 到结点 V8 的最短路径为：v1,v2,v3,v8 (2 分)

(241)已知如图所示的有向图，请给出该图的：

（1） 每个顶点的入度、出度；

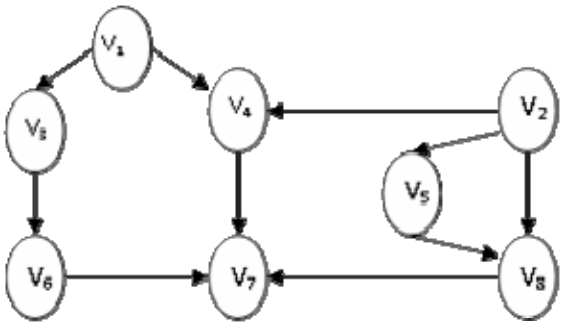
（2） 邻接矩阵；

（3） 邻接表；



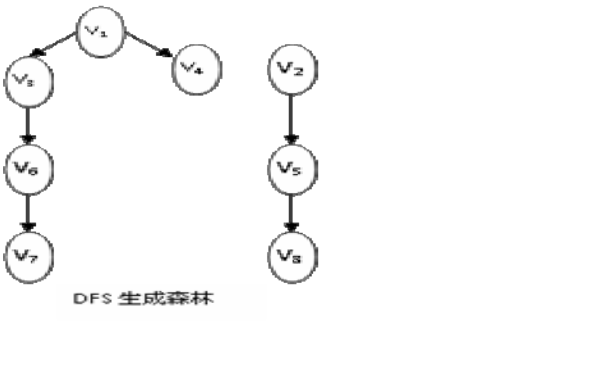
	v1	v2	v3	v4	v5	v6
v1	∞	5	∞	7	∞	∞
v2	∞	∞	4	∞	∞	∞
v3	8	∞	∞	∞	∞	9
v4	∞	∞	5	∞	∞	6
v5	∞	∞	∞	5	∞	∞
v6	3	∞	∞	∞	1	∞

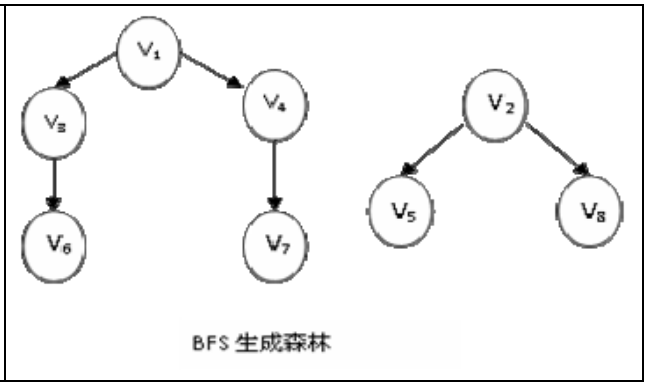
(242) 对下面的有向图，从顶点 V1 开始进行遍历，试画出遍历得到的 DFS 生成森林和 BFS 生成森林。



图全对给 4 分，错一个顶点扣 1 分，扣完为止。

遍历得到的 DFS 生成森林和 BFS 生成森林如下图：





(243) 采用哈希函数 H(k)=3*k mod 13 并用线性探测开放地址法处理冲突，在数列地址空间［0..12］中对关键字序列 22,41,53,46,30,13,1,67,51

（1）构造哈希表（画示意图）；

（2）装填因子；等概率下

（3）成功的和

（4）不成功的平均查找长度

（1）答：

散列地址	0	1	2	3	4	5	6	7	8	9	10	11	12
关键字	13	22		53	1		41	67	46		51		30
比较次数	1	1		1	2		1	2	1		1		1

（2）装填因子=9/13=0.7

（3）ASL_{succ} =11/9

(244) 设有一组关键字{9,01,23,14,55,20,84,27}，采用哈希函数：H（key）=key mod 7 ，表长为 10，用开放地址法的二次探测再散列方法 Hi=(H(key)+di) mod 10(di=12,22,32,…,)解决冲突。要求：对该关键字序列构造哈希表，并计算查找成功的平均查找长度。

散列地址	0	1	2	3	4	5	6	7	8	9
关键字	14	01	9	23	84	27	55	20		
比较次数	1	1	1	2	3	4	1	2		

平均查找长度：ASLsucc=（1+1+1+2+3+4+1+2）/8=15/8

以关键字 27 为例：H（27）=27%7=6（冲突） H1=（6+1）%10=7（冲突）

H2=（6+22）%10=0（冲突） H3=（6+33）%10=5 所以比较了 4 次。

(245) 对于给定的一组记录的关键字{ 23,13,17,21,30,60,58,28,30,90},试分别写出冒泡排序、快速排序、堆排序、归并排序第一趟排序后的结果。

冒泡排序 13,23,17,21,,28,30,60,58,30*, 90

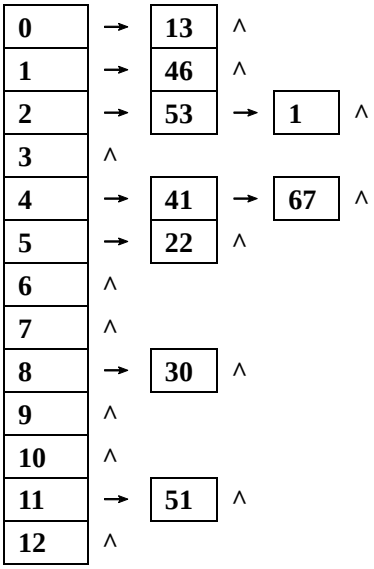
快速排序：(21,13,17,) 13,(30,60,58,28,30*,90)

(246) 采用哈希函数 H(k)=2*k mod 13 并用链地址法处理冲突，在数列地址空间［0..12］中对关键字序列 22,41,53,46,30,13,1,67,51 进行下列工作：

（a）构造哈希表（画示意图）；

（b）等概率下成功的和不成功的平均查找长度。

参考答案：链地址表全对给 8 分。错一个结点扣 1 分，扣完为止。



ASLsucc=(7+4)/13=11/13（1 分）

(247)　设哈希（Hash）表的地址范围为 0～17，哈希函数为：H（K）＝K　MOD　16。

K 为关键字，用线性探测法再散列法处理冲突，输入关键字序列：

（10，24，32，17，31，30，46，47，40，63，49）

造出 Hash 表，试回答下列问题：

（1）　画出哈希表的示意图；

（2）　若查找关键字 63，需要依次与哪些关键字进行比较？

（3）　若查找关键字 60，需要依次与哪些关键字比较？

（4）　假定每个关键字的查找概率相等，求查找成功时的平均查找长度。

解：　（1）画表如下：

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
32	17	**63**	**49**					24	**40**	10				30	31	**46**	**47**

（2）　查找 63, 首先要与 H(63)=63%16=15 号单元内容比较，即 63 vs 31 , no;

然后顺移，与 46, 47, 32, 17, 63 相比，一共比较了 6 次！

（3）查找 60,首先要与 H(60)=60%16=12 号单元内容比较，但因为 12 号单元为空（应当有空标记），所以应当只比较这一次即可。

（4）　对于黑色数据元素，各比较 1 次；共 6 次；

对红色元素则各不相同，要统计移位的位数。“**63**”需要 6 次，“**49**”需要 3 次，“**40**”需要 2 次，“**46**”需要 3 次，“**47**”需要 3 次，

所以 ASL=1/11（6＋2＋3×3）＝17/11=1.5454545454≈1.55

四、算法设计题(10 分)

(248)阅读下列递归算法，写出非递归方法实现相同功能的 C 程序。

```
void   test(int   &sum)
{ int   x;
   scanf(x);
   if(x=0) sum=0 else {test(sum); sum+=x;}
   printf(sum);
}
```

```
#include<stdio.h>     (1 分)
void main()     (1 分)
{   int x,sum=0,top=0,s[];     (1 分)
    scanf("%d",&x)
    while (x<>0)
          { s[++top]:=a; scanf("%d",&x); }(3 分)
    while (top)
          sum+=s[top--]; (3 分)
    printf("%d",sum); (1 分)
 }
```

(249)试写出把图的邻接矩阵表示转换为邻接表表示的算法。

设图的邻接矩阵为 g[n][n]（针对无向图），定义邻接表节点的类型为

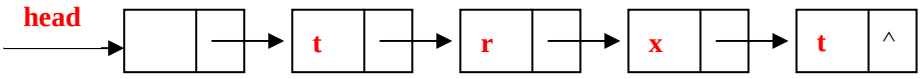
```
struct edgenode
{   int adjvex;
    edgenode   next;
}
typedef   edgenode   *adjlist[n];
void matritolist (int g[][], adjlist gl, int n )
```

```
{       edgenode *p, *q;
        for (int i=0 i<n; i++)    gl[i]=null;
        for (int i=0; i<n; i++)
        for ( int j=0; j<n; j++)
        {     if    (g[i][j]!=0 )
                    p = ( edgenode *) malloc(sizeof (edgenode));
              p->adjvex=j;
              p->next=null;
              if    (gl[i]=null)     {     gl[i]==p;       q=p;     }
              else    (    q->next=p;              q=p;    }
        }
}
```

(250) 阅读算法并根据输入数据画出链表。

```
linklist createlistr1(    )
{       char    ch;
        linklist          head=(listnode*)malloc(sizeof(listnode));
        listnode    *p，*r;
        r=head;
        while((ch=getchar( ))!= ' \n′)
        {         p=(listnode*)malloc(sizeof(listnode));
                  while    (p)    p=(listnode*)malloc(sizeof(listnode));
                  p–>data=ch;             r–>next=p;                r=p;
        }
        r–>next=NULL;
        return(head);
}
```

输入数据为：text↵



(251) 阅读算法并指出下列各段程序完成的功能。

```
void    add_poly(Lnode *pa,Lnode *pb)
{   Lnode *p,*q,*u,*pre;      int x;
    p=pa->next;      q=pb->next;      pre=pa;
    while((p!=NULL) && ((q!=NULL))
    {     if      (p->exp < q->exp)
                {     pre=p;    p=p->next;              }
          else        if (p->exp= =q->exp)
                        {    x=p->coef+q->coef;
                             if    (x!=0) { p->coef=x;    pre=p;}
                             else       { pre->next=p->next;    free(p); }
                             p=pre->next;       u=q;       q=q->next;
                             free(u);
                        }
                  else
                        {       u=q->next;q->next=p;pre->next=q;
                                pre=q;    q=u;
                         }
          }
          if (q!=NULL)      pre->next=q;
          free(pb);
}
```

两个多项式相加

(252) 阅读下面的程序，说明程序的具体功能。

```
typedef int elementype
```

```
typedef struct node
{   elemtype   data;
    strunct node   *next;
}linklist;
void function( linklist *head, elemtype x )
{   linklist *q, *p;
    q=head;
    p=q-next;
    while (p !=NULL) && (p->data != x )
    {   q=p;   p=p->next;   }
    if   (q==NULL)     printf ("there is no this node ::\n");
    else   {   q ->next =p ->next ;   free (p);      }
}
```

该程序的功能是：在带头结点的单链表中，删除单链表中枝为 z 的数据元素。

(253) 阅读下面的程序，说明程序的具体功能。

```
void function(   )
{         initstack(s);
          scanf ("%",n);
          while(n)
          {         push(s,n%8);              n=n/8;              }
          while(! Stackempty(s))
          {          pop(s,e);                printf("%d",e);       }
   }
```

该程序的功能是：10 进制数转换为 8 进制

(254) 阅读下面的程序，说明程序的具体功能。

```
void      print(int   w)
{         int   i;
          if ( w!=0)
          {        print(w-1);
                   for(i=1;i<=w;++i)
                            printf("%3d,",w);
                   printf("/n");
          }
}
```

运行结果：

1，

2，2，

3，3，3，

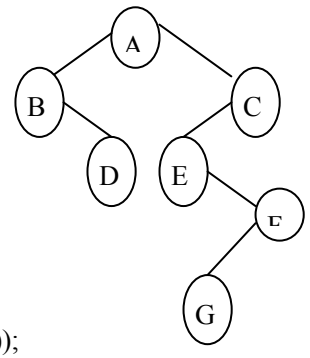
(255) 阅读下面的程序，分别说明程序中四个 for 循环和语句 ++cpot[col];的具体功能。

```
void FastTransposeSMatrix(Matrix M, Matrix &T)
{         T.mu=M.nu;   T.nu=M.mu;   T.tu=M.tu;
          if (T.tu)
               {    for (col=1;col<=M.nu;++col)        num[col]=0;
                    for (t=1;t<=M.tu;++t)              ++num[M. item[t].j];
                    cpot[1]=1;
                    for (col=2;col<=M.nu;++col)
                              cpot[col]=cpot[col-1]+num[col-1];
```

```
                    for (p=1;p<=M.tu;++p)
                    {   col=M. item[p].j;
                          q=cpot[col];
                          T.item[q].i=M.data[p].j;
                           T.item [q].j=M. item[p].i;
                          T.item[q].e=M.data[p].e;
                           ++cpot[col];
                     }
                }
            return OK;
    }
```

第一个 for 循环：初始化每一列中非零元素的个数为 0
第二个 for 循环，计算每一列中非零元素的个数；
第三个 for 循环，计算每一列的第一个元素的首地址；
第四个 for 循环，转置过程；
++cpot[col]：语句的功能是当每一列进行一次转置后，其位置向后加 1。

(256) 已知二叉树的二叉链表存储表示，指出建立如图所示树的结构时输入的字符序列。




```
typedef struct BiTNode
{       char data;
        struct BiTNode *lchild,*rchild;
} BiTNode;
void CreateBiTree( BiTNode *T)
{   char ch;
     scanf(&ch);
     if(ch= =‘ ‘) T=NULL;
     else{        T=(BiTNode *)malloc(sizeof(BiTNode));
                    while (T==NULL)
                          T=(BiTNode *)malloc(sizeof(BiTNode));
                   T–>data=ch;
                   CreateBiTree(T–>lchild);
                   CreateBiTree(T–>rchildd);
              }
}
```

AB␣D␣␣CE␣FG␣␣␣↵

(257) 已知二叉树的二叉链表存储表示，写出中序遍历的递归算法。

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

void inorder( BiTNode    *p)
{   if    (p!=NULL)
        {     inorder(p–>lchild);
              printf(“%c”,p–>data)
              inorder(p–>rchild);
        }
}
```

(258)阅读下面的程序，分别指出程序中三个 for 循环、整个程序以及语句 scanf("%d,%d",&G.vexnum,&G.arcnum);　的功能。

```
void funcgraph(MGraph &G)
{    int i,j,k,w;       char v1,v2;
     printf("Input vexnum & arcnum:");
     scanf("%d,%d",&G.vexnum,&G.arcnum);
     printf("Input Vertices:");
     for (i=0;i<G.vexnum;i++)
          scanf("%c",&G.vexs[i]);
     for (i=0;i<G.vexnum;i++)
          for (j=0;j<G.vexnum;j++)
               G.arcs[i][j]=0;
     for (k=0;k<G.arcnum;k++)
     {    printf("Input Arcs(v1,v2 & w):\n");
          scanf("%c%c,%d",&v1,&v2,&w);
          i=LocateVex(G,v1);
          j=LocateVex(G,v2);
          G.arcs[i][j]=w;     G.arcs[j][i]=w;
     }
}
```

第一个 for 循环：将图中的顶点输入到数组 G.vexs[i]；

第二个 for 循环，初始化邻接矩阵；

第三个 for 循环，将图中边信息存入数组 G.vexs[i]中；

本程序的功能是：创建图的邻接矩阵；

scanf("%d,%d",&G.vexnum,&G.arcnum)；：语句的功能输入定点数和图中的边数。

试题编号：

重庆邮电大学2011－2012学年第　学期

《数据结构》 期末 模拟考试题

题号	一	二	三	四	五	六	总分
分数							
评卷人							

注意：答案写到后面的答题纸上，按要求答题，并请保持字迹清楚，容易阅读。

一、选择题（每题2分，共30分）

1. 栈和队列的共同点有（ ? ）。

A. 都是先进先出　　B. 都是后进先出

C. 不会删除中间的元素　　D. 完全没有共同点

2. 链表不具有的特点是（ ? ）。

A. 可随机访问任一元素　　B. 插入、删除不需要移动元素

C. 不必事先估计存储空间　　D. 所需空间与线性表长度成正比

3. 在解决计算机主机与打印机之间速度不匹配问题时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，而打印机则依相同次序从该缓冲区中取出数据打印。该缓冲区作为数据结构是一个（ ? ）结构。

A. 栈　B. 队列　C. 哈希表（Hash Table）　D. 线性表

4. 设计一个判别表达式中左、右括号是否配对出现的算法，采用（ ? ）数据结构最佳。

A. 栈　B. 队列　C. 顺序结构线性表　D. 链式结构线性表

5. 若某栈的输入序列为1, 2, 3, …, n，输出序列的第一个元素为n，则第2个输出元素为（ ? ）。

A. 1　B. n-1　C. n　D. 都有可能

6. 首先访问某结点的左子树，然后访问该结点，最后访问结点的右子树，这种遍历称为（ ? ）

A. 前序遍历　B. 后序遍历　C. 中序遍历　D. 层次遍历

7. 下列排序算法中，时间复杂度不受数据初始状态影响，恒为$O(n\log_2 n)$的是（ ? ）。

年级：　专业：　班级：　姓名：　学号：

密　封　线

A. 快速排序　　B.冒泡排序　　C.直接选择排序　　D. 堆排序

8. 若长度为 n 的线性表采用顺序存储结构，在其第 i 个位置插入一个元素的时间复杂度为（ ？ ）（1<= i <= n+1 ）。

A. O(0)　　B. O(1)　　C. O(n)　　D. $O(n^2)$

9. 已知数据表 A 中每个元素距其最终位置不远，则采用（ ？ ）排序算法最节省时间。

A.堆排序　　B.直接插入排序　　C.快速排序　　D.简单选择排序

10. 任何一个无向连通图的最小生成树（ ？ ）。

A.只有一棵　　B.有一棵或多棵　　C.一定有多棵　　D.可能不存在

11. 下列序列中，（ ？ ）是执行第一趟快速排序后得到的序列（排序的关键字类型是字符串）。

A.[da,ax,eb,de,bb] ff [ha,gc]　　B.[cd,eb,ax,da] ff [ha,gc,bb]

C.[gc,ax,eb,cd,bb] ff [da,ha]　　D.[ax,bb,cd,da] ff [eb,gc,ha]

12. 对包含 N 个元素的散列表进行查找，平均查找长度 （ ）

A、为 $O(\log_2 N)$　B、为 O(N)　C、不直接依赖于 N　D、三者都不是

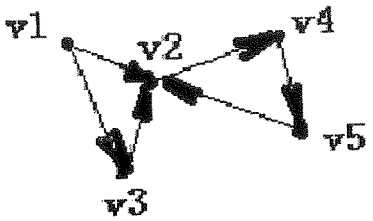
13. 给定下列有向图和初始结点 V_1，按深度优先遍历的结点序列为(？)

A、V1，V3，V4，V5，V2

B、V1，V2，V3，V4，V5

C、V1，V2，V5，V3，V4

D、V1，V2，V4，V5，V3



14. 串是（？ ）。

A.不少于一个字母的序列　　B. 任意个字母的序列

C.不少于一个字符的序列　　D.有限个字符的序列

15. 有 n 个球队参加的某联赛按单循环方式进行比赛，那么共需要进行（ ？ ）场比赛。

A. n(n-1) /2　B. n　C. n(n-1)　D. n+1

装 年级： 密 专业： 班级： 封 姓名： 线 学号： 线 订

二、填空题（每题 2 分，共 20 分）

1. 采用特殊字符作为串的结束，串 S= “WinFilename”需要至少长度为（ ？ ）的字符数组存放。

2. 已知数组 A[1..10,1..10]为对称矩阵，其中每个元素占 5 个单元。现将其下三角部分按行优先次序存储在起始地址为 1000 的连续内存单元中，则元素 A[5，6]对应的地址为（ ？ ）。

3. 已知完全二叉树的第 5 层有 3 个结点(根结点为第 1 层)，则其结点数是(？)

4. 已知二叉树中叶子结点数为 12，仅有一个孩子的结点数为 5，则总结点数是（ ？ ）。

5. 具有 12 个结点的完全二叉树的高度（空树高度为 0）为 （ ？ ）。

6. 高度（空树高度为 0）为 5 的 AVL 树，其结点数最少是 （ ？ ）。

7. 在链式结构的线性表中插入元素的算法复杂度是（ ？ ）。

8. 已知一个无向图的邻接矩阵表示，计算第 j 个结点的度的方法是（ ？ ）。

9. G 为无向图，如果从 G 的某个顶点出发进行一次遍历，即可访问图的每个顶点，则该图一定是（ ？ ）图。

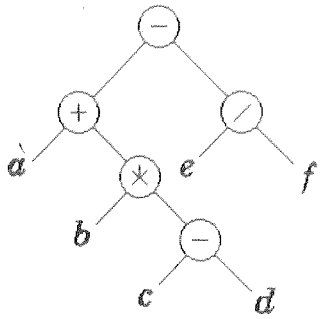
10. 对于键值序列{12,13,11,18,60,15,7,18,25,100}，建里初始堆，必须从键值为（ ？ ）的结点开始对每个结点进行一次堆调整。

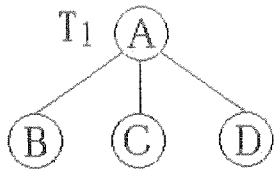
三、问答题。（每题 6 分，共 24 分）

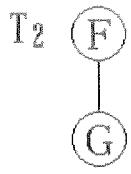
1. 直接选择排序是选出 n 个数据元素中最小的（或最大的），与最左（右）边的数据元素相交换，然后按同样的办法考虑剩下的 n-1 数据元素直到只剩下一个数据元素为止。请分析直接选择排序算法的时间复杂度。

2. 已知关键字序列为 36, 31, 20, 32, 66, 48, 依次将各元素插入到一棵初始为空的二叉排序树，画出对应的二叉排序树。

3. 已知二叉树如左下图，试写出后序遍历结果。



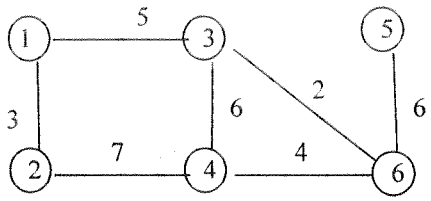




4. 现有森林如右上图，请画出对应的二叉树。

四、算法应用、分析题（共 18 分）

1. 图 G 各顶点的连接关系及相应权值如下图所示。(1) 画出图的邻接表存储图示 (2) 并从顶点 1 开始对图进行广度优先遍历，写出遍历结果；(3) 使用 Kruskal 算法求该图的最小生成树，给出的形成过程。(11 分)



2. 设 Hash 函数为 H(K)=K mod 7，哈希表的地址空间为 0,...，6，开始时哈希表为空，用平方探测法解决冲突，请画出依次插入键值 9，14，10，30，56 后的哈希表。

五、算法设计题。(共 8 分)

已知二叉树用二叉链表存储，试编写一函数实现计算该树的高度。请定义必要的数据结构。

一、选择题

C　A　B　A　B

C　D　C　B　B

A　C　D　D　A

二、填空题

（1）．12　（2）．1095　（3）．18　（4）．28　（5）．4

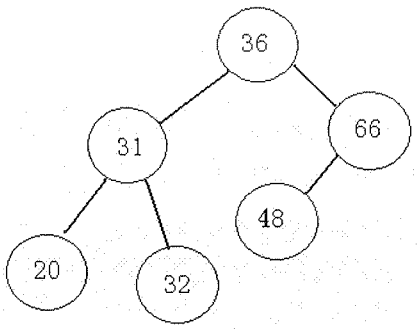
（6）．12　（7）．O（n）　（8）．行/列里 1 的个数

（9）．连通图　（10）．60

三、问答题

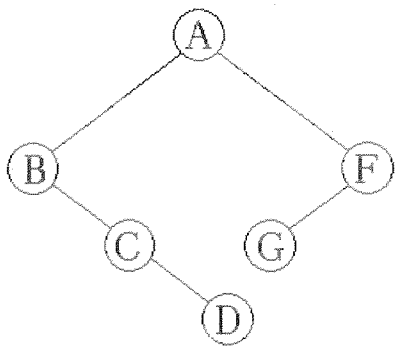
1．从 n 个数据中选择一个最值数据，需要 n－1 次比较，然后从从 n－1 个数据中选择一个最值数据，需要 n－2 次比较，依次类推。其时间复杂度为 O（n2）

2.



3.　a b c d - * + e f / -

4.

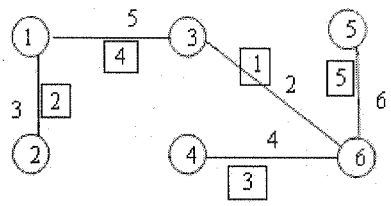


四、 1. (1) 图的邻接表 (略)

1
2
3
4
5
6

(2). 遍历结果 1、2、3、4、6、5

(3). 生成树



2. 最后 hash 表为 [本题答案应唯一。数据每放错一个扣 1 分。]

14	56	9	10			30
0	1	2	3	4	5	6

五 编程

试题编号：

重庆邮电大学2010－2011学年第 二 学期

数据结构（C语言） 期末 考试题

题　号	一	二	三	四	五	六	七	八	总 分
得　分									
评卷人									

一、单项选择题（每小题2分，共24分）

1．计算机识别、存储和加工处理的对象被统称为（ A ）

A.数据　　B.数据元素

C.数据结构　　D.数据类型

2．栈和队列都是（ A ）

A．限制存取位置的线性结构　　B．顺序存储的线性结构

C．链式存储的线性结构　　D．限制存取位置的非线性结构

3．如果以链表作为栈的存储结构,则退栈操作时（ D ）

A.必须判别栈是否满干　　B.对栈不作任何判别

C.判别栈元素的类型　　D.必须判别栈是否空

4．采用两类不同存储结构的字符串可分别简称为（ B ）

A.主串和子串　　B.顺序串和链串

C.目标串和模式串　　D.变量串和常量串

5． 一个向量第一个元素的存储地址是100，每个元素的长度为2，则第5个元素的地址是：B

A. 110　　B .108

C. 100　　D. 120

6．二分查找要求被查找的表是（ C ）

A.键值有序的链接表　　B.链接表但键值不一定有序表

C.键值有序的顺序表　　D.顺序表但键值不一定有序表

7．设高度为h的二叉树上只有度为0和度为2的结点，则此类二叉树中所包含的结点数至少为： C

A. 2h　　B .2h-1

C. 2h+1　　D. h+1

软件开发网

8．树的基本遍历策略可分为先根遍历和后根遍历；二叉树的基本遍历策略可分为先序遍历、中序遍历和后序遍历。这里，我们把 由树转化得到的二叉树叫做这棵树对应的二叉树。下列结论哪个正确？ A

A. 树的先根遍历序列与其对应的二叉树的先序遍历序列相同

B .树的后根遍历序列与其对应的二叉树的后序遍历序列相同

C. 树的先根遍历序列与其对应的二叉树的中序遍历序列相同

D. 以上都不对

9．一个有n个顶点的无向图最多有多少边？C

A. n　　B .n(n-1)
C. n(n-1)/2　　D. 2n

10. 设数组Data[0..m]作为循环队列SQ的存储空间，front为队头指针，rear为队尾指针，则执行出队操作的语句为(A)

A.front=(front+1)%(m+1)　B.front=(front+1)% m
C.rear=(rear+1)% m　　D. front=front+1

11.当在二叉排序树中插入一个新结点时，若树中不存在与待插入结点的关键字相同的结点，且新结点的关键字小于根结点的关键字，则新结点将成为（ A ）

A. 左子树的叶子结点　　B. 左子树的分支结点
C. 右子树的叶子结点　　D. 右子树的分支结点

软件开发网

12. 对于哈希函数H(key)=key%13,被称为同义词的关键字是(D)

A.35和41　　B.23和39
C.15和44　　D.25和51

二、填空题(每空2分，共24分)

1.设r指向单链表最后一个结点，要在最后一个结点之后插入s所指的结点，需执行的三条语句是 r->next=s ; r=s; **r->next=NULL** 。

2.在带头结点单链表L中，表空的条件是 **L->next==NULL**

3.设一个链栈的栈顶指针为ls，栈中结点格式为 info | link ，栈空的条件是 **ls==NULL** 。若栈不空，则退栈操作为 p=ls; **ls=ls->link** ;free(p).

4.已知一棵度为3的树有2个度为1的结点，3个度为2的结点，4个度为3的结点，则该树中有 **12** 个叶子结点。

5.树有三种常用的存储结构，即孩子链表法，孩子兄弟链表法和 **双亲表示法** 。

6.n-1个顶点的连通图的生成树有 **n-2** 条边。

7.一个有向图G中若有弧<Vj,Vi>、<Vi,Vk>和<Vj,Vk>，则在图G的拓朴序列中，顶点Vi,Vj和Vk的相对位置为 **Vj->Vi->Vk** 。

8.设表中元素的初始状态是按键值递增的，分别用堆排序、快速排序、冒泡排序和归并排序

方法对其进行排序(按递增顺序)， **冒泡排序** 最省时间， **快速排序** 最费时间。

9.下面是将键值为X的结点插入到二叉排序树中的算法，请在划线处填上适当的内容。

```
typedef  struct node *pnode
    struct node
    { int  key;
    pnode left,right;
```

```
    }
void searchinsert(int x; pnode t);
    //t为二叉排序树根结点的指针//
{ if( t=NULL )
    { p=malloc(size); p->key=x; p->left=nil; p->right=nil; t=p;}
  else if (x<t->key) searchinsert(x,t->left)
       else  searchinsert(x,t-> right) ;
}
```

三、（10分）下图是带权的有向图G的邻接矩阵表示，请给出：

1、其邻接表存储结构

2、按Floyd算法求所有顶点对之间最短距离的矩阵变化过程。

	V1	V2	V3	V4
V1	0	1	∞	4
V2	∞	0	9	2
V3	3	5	0	8
V4	∞	∞	6	0

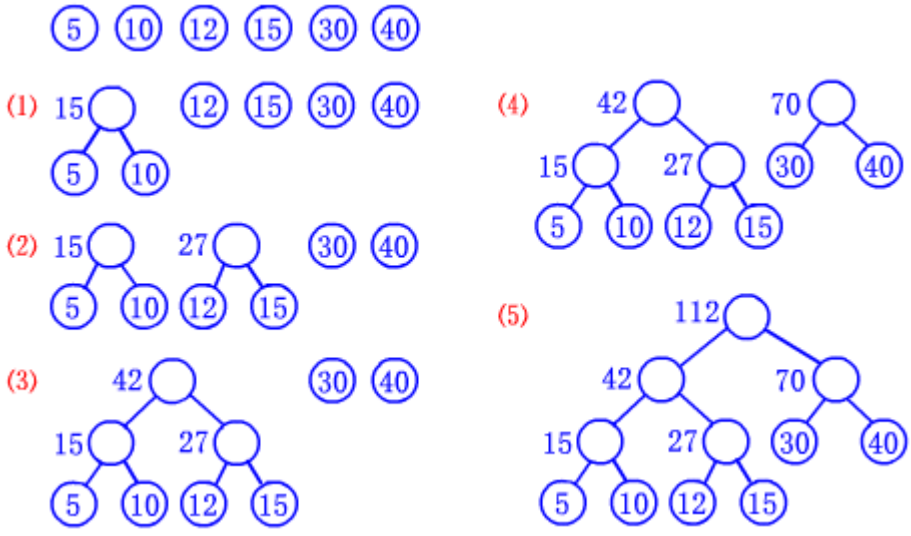
解：Floyd算法执行过程中矩阵的变化情况为（从左到右）：

0	1	∞	4	0	1	10	3	0	1	10	3	0	1	9	3
∞	0	9	2	∞	0	9	2	12	0	9	2	11	0	8	2
3	4	0	7	3	4	0	6	3	4	0	6	3	4	0	6
∞	∞	6	0	∞	∞	6	0	9	10	6	0	9	10	6	0

四、应用题(本题共 28 分)

1.给定权值{5,10,12,15,30,40},构造相应的哈夫曼树，要求写出构造步骤。(4分)

哈夫曼树构造步骤:



学号 姓名 院（系） 班级 线 订 装

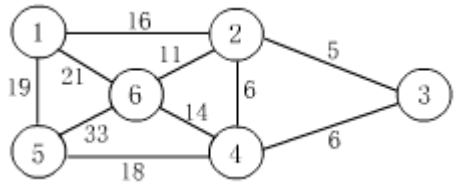
2. 已知一表为(Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec)，按表中顺序依次插入初始为空的二叉排序树，要求：

(1)在右边画出建立的二叉排序树。(4 分)

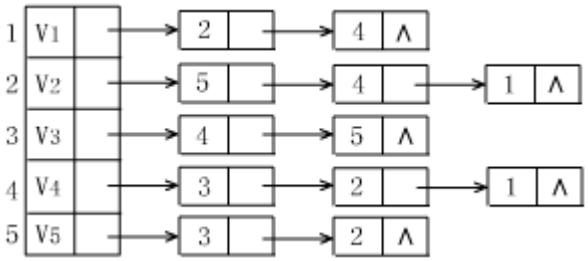
(2)求出在等概率情况下查找成功的平均查找长度。(2 分)

ASL_{SU} =(1+2*2+3*3+4*3+5*2+6)/12=42/12=3.5

3. 下图表示一个地区的交通网，顶点表示城市，边表示连结城市间的公路，边上的权表示修建公路花费的代价。怎样选择能够沟通每个城市且总造价最省的 n-1 条公路，画出所有可能的方案.(4 分)



4. 已知一个无向图的邻接表为:(本题 4 分,每小题 2 分)



(1)画出这个图。

(2)以 V1 为出发点，对图进行广度优先搜索，写出所有可能的访问序列。

V1->V2->V4->V5->V3

V1->V4->V2->V3->V5

5. 设 n 个元素的有序表为 R,K 为一个给定的值，二分查找算法如下：

```
int binsearch(sqlist R; keytype K:);
```

```
{
  l=1; h=n; suc=false;
  while (l<=h)&&(!suc) do
  { mid=(l+h)/2;
    case
      K=R[mid].key: suc=true;
      K<R[mid].key: h=mid-1;
      K>R[mid].key: l=mid+1
    end }
  if (suc) return(mid) else return(0)
}
```

将上述算法中划线语句改为：K<R[mid].key: h=mid.

问改动后，算法能否正常工作？请说明原因。

若能正常工作，请给出一个查找序列和查找某个键值的比较次数.(本题 4 分)

答：（1）若 K 在 R 中或大于 R 中的最大值，则算法能正常运行；

若 K 不在 R 中或小于 R 中的最大值，则算法不能正常运行，会出现死循环；

（2）如：在[2，4，6，8]中，当 K=7 时，算法出现死循环；

当 K=6 时，算法能正常运行，查找成功，比较次数为 2 次。

6.有一组键值 27,84,21,47,15,25,68,35,24,采用快速排序方法由小到大进行排序，请写出每趟的结果，并标明在第一趟排序过程中键值的移动情况。(本题共 6 分)

答：(1)每趟的结果：

(2)第一趟排序键值移动情况：

五、设计题(本题共 14 分)

1.一棵二叉树以二叉链表为存储结构 | lchild | data | rchild | 。设计一个算法，求在前序序列中处于第 K 个位置的结点。(本题 6 分)

类型定义如下:

```
typedef  struct  node  * pointer ;
struct  node
{ datatype  data ;
  pointer lchild, rchild ;
}
typedef  pointer  bitreptr ;
```

算法如下:

```
void  pre ( bitreptr t ; int k; bitreptr p )
{ if ( t!=NULL )
  { i=i+1;
   if ( i==k){ p=t; return(p);}
   pre(t->lchild, k,p ) ;
   pre(t->rchild, k,p ) ;
  }
}
```

2.某单链表 L 的结点结构为 | data | next | ，结点个数至少 3 个，试画出该链表的结构图，

并编写算法判断该链表的元素是否成等差关系，即:设各元素值依次为 a1,a2,...,an, 判断 ai+1-ai=ai-ai-1 是否成立，其中 i 满足 2<=i<=n-1。(8 分)

结构图: (略)

算法如下:

```
int dcsl(lklist L)
{ p=L; q=p->next; r=q->next;
 while ( r!= NULL)
  if ((p->data)-(q->data) != (q->data)-(r->data)) return(0);
  else { p=q; q=r; r=r->next; }
 return(1);
}
```

试题编号：

重庆邮电大学2010－2011学年第一学期

数据结构试卷（期末）（B卷）（闭卷）

题　号	一	二	三	四	五	六	七	八	总 分
得　分									
评卷人									

年级：　专业：　班级：　姓名：　学号：

装　密　订　封　线　线

一、选择题（本大题15小题，第小题2分，共30分）

1、若目标串的长度为 n，模式串的长度为[n/3]，则执行模式匹配算法时，在最坏情况下的时间复杂度是（　）

A. O（3）　B. O（n）　C. O（n^2）　D. O（n^3）

2、树最适合用来表示(　　)。

A.有序数据元素　B.无序数据元素

C.元素之间具有分支层次关系的数据　D.元素之间无联系的数据

3、在一个单链表中,若 q 所指结点是 p 所指结点的前驱结点,若在 q6 与 p 之间插入一个 s 所指的结点,则执行(　　)。

A　s→link=p→link;　p→link=s;　B　p→link=s;　s→link=q;

C　p→link=s→link;　s→link=p;　D　q →link=s;　s→link=p;

4、由权值分别为 11，8，6，2，5 的叶子结点生成一棵哈夫曼树，它的带权路径长度为（　）

A　24　B　71　C　48　D　53

5、在一个长度为 n 的顺序线性表中顺序查找值为 x 的元素时，查找成功时的平均查找长度（即 x 与元素的平均比较次数，假定查找每个元素的概率都相等）为（　　）。

A　n　B　n/2　C　(n+1)/2　D　(n-1)/2

6、线性表采用链式存储时，结点的存储地址（　　）

A．必须是不连续的

B．连续与否均可

C．必须是连续的

D．和头结点的存储地址相连续

7、由两个栈共享一个向量空间的好处是：（　　）

A．减少存取时间，降低下溢发生的机率

B．节省存储空间，降低上溢发生的机率

C．减少存取时间，降低上溢发生的机率

D．节省存储空间，降低下溢发生的机率

8、在一棵度为 3 的树中,度为 3 的结点个数为 2,度为 2 的结点个数为 1,则度为 0 的结点个数为(　　)

A．4　　B．5　　C．6　　D．7

9、用某种排序方法对关键字序列（25，84，21，47，15，27，68，35，20）进行排序时，序列的变化情况如下：

20，15，21，25，47，27，68，35，84

15，20，21，25，35，27，47，68，84

15，20，21，25，27，35，47，68，84

则所采用的排序方法是（　　）

A．选择排序　B．希尔排序　C．归并排序　D．快速排序

10、设某棵二叉树的中序遍历序列为ABCD，前序遍历序列为CABD，则后序遍历该二叉树得到序列为（　）。

(A) BADC　　(B) BCDA　　(C) CDAB　　(D) CBDA

11、设某数据结构的二元组形式表示为A=(D，R)，D={01，02，03，04，05，06，07，08，09}，R={r}，r={<01，02>，<01，03>，<01，04>，<02，05>，<02，06>，<03，07>，<03，08>，<03，09>}，则数据结构A是（　）。

(A) 线性结构　(B) 树型结构　(C) 物理结构　(D) 图型结构

12、设一棵m叉树中有N_1个度数为1的结点，N_2个度数为2的结点，……，N_m个度数为m的结点，则该树中共有（　）个叶子结点。

(A) $\sum_{i=1}^{m}(i-1)N_i$　(B) $\sum_{i=1}^{m}N_i$　(C) $\sum_{i=2}^{m}N_i$　(D) $1+\sum_{i=2}^{m}(i-1)N_i$

13、设F是由T1、T2和T3三棵树组成的森林，与F对应的二叉树为B，T1、T2和T3的结点数分别为N1、N2和N3，则二叉树B的根结点的左子树的结点数为（　）。

(A) N1-1　(B) N2-1　(C) N2+N3　(D) N1+N3

14、设输入序列1、2、3、…、n经过栈作用后，输出序列中的第一个元素是n，则输出序列中的第i个输出元素是（　）。

(A) n-i　(B) n-1-i　(C) n+1 -i　(D) 不能确定

15、设有一组初始记录关键字序列为(34，76，45，18，26，54，92)，则由这组记录关键字生成的二叉排序树的深度为（　）。

(A) 4　(B) 5　(C) 6　(D) 7

二、填空题（本大题共10小题，第小题2分，共20分）

1、数据结构从逻辑上划分为三种基本类型：_____、_____和_____。

2、假定一棵树的广义表表示为A（C，D（E，F，G），H（I，J）），则树中所含的结点数为_____个，树的深度为_____，树的度为_____

3、设一棵二叉树的中序遍历序列为BDCA，后序遍历序列为DBAC，则这棵二叉树的前序序列为_____

4、设指针变量p指向单链表中结点A，则删除结点A的语句序列为：

q=p->next；p->data=q->data；p->next=(_____)；feee(q)；

5、设散列表的长度为8，散列函数H(k)=k % 7，用线性探测法解决冲突，则根据一组初始关键字序列(8，15，16，22，30，32)构造出的散列表的平均查找长度是___

2、已知待散列的线性表为（36，15，40，63，22），散列用的一维地址空间为[0..6]，假定选用的散列函数是 H（K）= K mod 7，若发生冲突采用线性探查法处理，试：

（1）计算出每一个元素的散列地址并在下图中填写出散列表：

0	1	2	3	4	5	6

（2）求出在查找每一个元素概率相等情况下的平均查找长度。

3、设散列表的地址范围是[0..9]，散列函数为 H（key）= （key ²+2）MOD 9，并采用链表处理冲突，请画出元素 7、4、5、3、6、2、8、9 依次插入散列表的存储结构。

6、对一组初始关键字序列（40，50，95，20，15，70，60，45，10）进行冒泡排序，则第一趟需要进行相邻记录的比较的次数为_____，在整个排序过程中最多需要进行_____趟排序才可以完成。

7、设有向图G的二元组形式表示为G=（D，R），D={1，2，3，4，5}，R={r}，r={<1,2>，<2,4>，<4,5>，<1,3>，<3,2>，<3,5>}，则给出该图的一种拓扑排序序列_____。

8、高度为h的完全二叉树中最少有_____个结点，最多有_____个结点。

9、设用于通信的电文仅由8个字母组成，字母在电文中出现的频率分别为7、19、2、6、32、3、21、10，根据这些频率作为权值构造哈夫曼树，则这棵哈夫曼树的高度为_____。

10、设无向图G中有n个顶点e条边，则用邻接矩阵作为图的存储结构进行深度优先或广度优先遍历时的时间复杂度为_____；用邻接表作为图的存储结构进行深度优先或广度优先遍历的时间复杂度为_____。

三、判断题（本大题共10小题，第小题1分，共10分）

1、调用一次深度优先遍历可以访问到图中的所有顶点。（　）

2．分块查找的平均查找长度不仅与索引表的长度有关，而且与块的长度有关。（　）

3．冒泡排序在初始关键字序列为逆序的情况下执行的交换次数最多。（　）

4．满二叉树一定是完全二叉树，完全二叉树不一定是满二叉树。（　）

5．设一棵二叉树的先序序列和后序序列，则能够唯一确定出该二叉树的形状。（　）

6、对连通图进行深度优先遍历可以访问到该图中的所有顶点。（　）

7．先序遍历一棵二叉排序树得到的结点序列不一定是有序的序列。（　）

8．由树转化成二叉树，该二叉树的右子树不一定为空。（　）

9．线性表中的所有元素都有一个前驱元素和后继元素。（　）

10．带权无向图的最小生成树是唯一的。（　）

四、解答题（本大题共3小题，第小题10分，共30分）

1．　已知一个图的顶点集V和边集E分别为：V={1, 2, 3, 4, 5, 6, 7}；

E={(1, 2)3, (1, 3)5, (1, 4)8, (2, 5)10, (2, 3)6, (3, 4)15,

(3, 5)12, (3, 6)9, (4, 6)4, (4, 7)20, (5, 6)18, (6, 7)25}；

用克鲁斯卡尔算法得到最小生成树，试写出在最小生成树中依次得到的各条边。

五、算法阅读题（本大题 10 分）

1. LinkList mynote(LinkList L)

```
{//L 是不带头结点的单链表的头指针
    if(L&&L->next){
        q=L; L=L->next; p=L;
S1:       while(p->next) p=p->next;
S2:       p->next=q; q->next=NULL;
    }
    return L;
}
```

请回答下列问题：

（1）说明语句 S1 的功能；

（2）说明语句组 S2 的功能；

（3）设链表表示的线性表为（$a_1,a_2,\cdots,a_n$），写出算法执行后的返回值所表示的线性表。

2. void ABC(BTNode * BT)

```
{
 if  BT {
   ABC (BT->left);
   ABC (BT->right);
   cout<<BT->data<<' ';
   }
}
```

该算法的功能是：

参考答案

一、选择题

1-5、CCABC　6-10、BBCDA　11-15、BDACA

二、填空题

1、线性结构，树型结构，图型结构

2、**9　3　3**

3、CBDA

4、（q->next）

5、____8/3____

6、__6_____，　_____8____

7、_1,2,3,4,5

8、___2^{h-1} ___2^h-1____。

9、_____6____。

10、$O(n^2)$，　O(n+e)

三、判断题（本大题共 10 小题，第小题 1 分，共 10 分）

1、调用一次深度优先遍历可以访问到图中的所有顶点。（　X）

2．分块查找的平均查找长度不仅与索引表的长度有关，而且与块的长度有关。（V　）

3．冒泡排序在初始关键字序列为逆序的情况下执行的交换次数最多。（V　）

4．满二叉树一定是完全二叉树，完全二叉树不一定是满二叉树。（ V ）

5．设一棵二叉树的先序序列和后序序列，则能够唯一确定出该二叉树的形状。（ X ）

6、对连通图进行深度优先遍历可以访问到该图中的所有顶点。（　V）

7．先序遍历一棵二叉排序树得到的结点序列不一定是有序的序列。（ V ）

8．由树转化成二叉树，该二叉树的右子树不一定为空。（ X ）

9．线性表中的所有元素都有一个前驱元素和后继元素。（　X）

10．带权无向图的最小生成树是唯一的。（X　）

四、解答题（本大题共 3 小题，第小题 10 分，共 30 分）

1、用克鲁斯卡尔算法得到的最小生成树为：

(1,2)3，　(4,6)4，　(1,3)5，　(1,4)8，　(2,5)10，　(4,7)20

2、H(36)=36 mod 7=1;

H(15)=15 mod 7=1;....冲突

H_1(15)=(1+1) mod 7=2;

H(40)=40 mod 7=5;

H(63)=63 mod 7=0;

H(22)=22 mod 7=1; ....冲突

H_1(22)=(1+1) mod 7=2; ....冲突

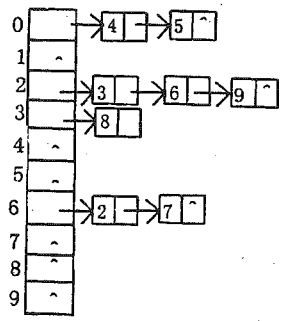
H_2(22)=(2+1) mod 7=3;

（1）

0	1	2	3	4	5	6
63	36	15	22		40	

（2）ASL=$\frac{1+2+1+1+3}{5}=1.6$

3、H(4)=H(5)=0, H(3)=H(6)=H(9)=2, H(8)=3, H(2)=H(7)=6



五、算法阅读题（本大题 10 分）

3. LinkList mynote(LinkList L)

```
{//L 是不带头结点的单链表的头指针
    if(L&&L->next){
        q=L; L=L->next; p=L;
S1:     while(p->next) p=p->next;
S2:     p->next=q; q->next=NULL;
    }
    return L;
}
```

请回答下列问题：

（1）说明语句 S1 的功能；

（2）说明语句组 S2 的功能；

（3）设链表表示的线性表为（$a_1,a_2,\cdots,a_n$），写出算法执行后的返回值所表示的线性表。

4. void ABC(BTNode * BT)

```
{
 if  BT {
   ABC (BT->left);
   ABC (BT->right);
   cout<<BT->data<<' ';
   }
}
```

该算法的功能是：

解：（1）查询链表的尾结点

（2）将第一个结点链接到链表的尾部，作为新的尾结点

（3）返回的线性表为（$a_2,a_3,\cdots,a_n,a_1$）

递归地后序遍历链式存储的二叉树。

2、递归地后序遍历链式存储的二叉树。

装 密 年级： 专业： 班级： 姓名： 学号： 封 订 线 线

重庆邮电大学 2009-2010 学年第 — 学期

数据结构（C 语言描述）考试题

题号	一	二	三	四	五	六	七	八	九	十	总分
分数											
评卷人											

一、单项选择题（在每小题的 4 个备选答案中，选出正确的答案。每小题 1 分，共 10 分）

1、 链表不具有的特点是（ ）

A. 可随机访问任意元素　　B.插入删除不需要移动元素

C. 不必事先估计存储空间　　D.所需空间与线性表长度成正比

2、设计一个判别表达式中左右括号是否配对出现的算法，采用（ ）数据结构最佳。

A. 栈　　B. 线性表的顺序存储结构

C. 队列　　D. 线性表的链式存储结构

3、串是一种特殊的线性表，其特殊性体现在（ ）

A. 可以顺序存储　　B. 数据元素是一个字符　　C. 可以链接存储

D. 数据元素可以是多个字符

4、判定一个循环队列 QU（最多元素为 m0）为满队列的条件是（ ）。

A. QU->front==QU->rear　　B. QU->front==(QU->rear+1) % m0

C. QU->front!=QU->rear　　D. QU->front!=(QU->rear+1) % m0

5、数据结构被形式地定义为(K，R)，其中 K 是数据元素的有限集，R 是 K 上的（ ）有限集。

A.操作　　B.映象　　C. 关系　　D.存储

6、深度为 5（根的层次为 1）的二叉树至多有（ ）结点。

A. 32　　B.16　　C. 31　　D. 15

7、一个具有 n 个顶点的无向完全图的边数为（ ）

A. n(n+1)/2　　B. n(n+1)　　C. n(n-1)/2　　D. n(n-1)

8、若某线性表中最常用的操作是在最后一个结点之后插入一个结点和删除最后一个结点，则采用（　　）存储方式最节省运算时间。

A．容量足够大的顺序表　　B. 双链表　　C.单链表　　D.带头结点的双循环链表

9、快速排序方法在（　　）情况下最不利于发挥其长处。

A．要排序的数据量太大　　B. 要排序的数据已基本有序　　C. 要排序的数据中含有多个相同值　D.要排序的数据个数为奇数

10、AVL 树是一种平衡的二叉排序树，树中任一结点的（　　）

A.左、右子树的高度均相同　　B. 左子树的高度均小于右子树的高度

C. 左、右子树高度差的绝对值不超过 1　　D. 左子树的高度均大于右子树的高度

二、填空题（每空 2 分，共 20 分）

1、数据存储结构主要包括顺序存储和_____结构。

2、对于一个具有 n 个顶点的图，若采用邻接矩阵表示，则矩阵大小为_____。

3、对于一棵完全二叉树，若一个结点的编号为 i，则它的左孩子结点的编号为_____。

4、在以 HL 为表头指针的带表头附加结点的单链表和循环单链表中，链表为空的条件分别为_____和_____。

5、树有三种常用的存储结构，即孩子链表法、孩子兄弟链表法和_____。

6、在一个长度为 n 的顺序表中第 i 个元素（1≤i≤n）之前插入一个元素时，需向后移动元素的个数是_____。

7、n 个顶点的连通图的生成树有_____条边。

8、设 s[1..maxsize]为一个顺序存储的栈，变量 top 指示栈顶位置，栈为空的条件是_____，栈为满的条件是_____。

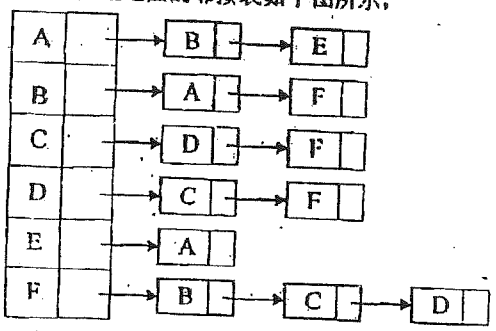
三、应用题（每小题5分，共20分）

1. 有一组关键码序列{8, 9, 5, 3, 7, 2, 1}，分别采用冒泡排序、快速排序、直接选择排序、二路归并排序方法由小到大进行排序，在下面的选项中请选择各种排序第一趟排序的结果。

A. {1, 2, 5, 3, 7, 8, 9}　B. {1, 9, 5, 3, 7, 2, 8}　C. {8, 9, 5, 3, 7, 2, 1}
D. {9, 5, 3, 7, 2, 1, 8}　E. {8, 5, 3, 7, 2, 1, 9}　F. {8, 9, 3, 5, 2, 7, 1}

2. 有七个带权结点，其权值分别为{3, 4, 9, 10, 12, 15, 18}，试以它们为叶子结点构造一棵哈夫曼树，并计算出带权路径长度 WPL。

3. 已知一个无向连通图的邻接表如下图所示，



（1）根据此邻接表写出从 A 点开始按深度优先遍历该图的结果。

（2）画出此无向连通图的邻接矩阵示意图。

4. 假定一个关键字序列为(32,75,29,63,48,94,25,46,18,70)，地址空间为 HT[13]，若采用除留余数法构造哈希函数 H(K) = k %13 和线性探查法处理冲突，试求出每一元素的散列地址，画出最后得到的散列表。

四、算法填空和分析（每空2分，共20分）

2. 下面是将键值为 x 的结点插入到二叉排序树中的算法，请在划线处填上适当的内容。

```
typedef struct pnode
  {int key;
  struct pnode    * lchild , * rchild;
  }pnode;
```

```
void searchinsert(int x, pnode t )
    /*t 为二叉排序树根结点的指针* /

{if (    (1)    )
      {p=( pnode *)malloc(sizeof(pnode));
       p->key=x;p->lchild=null;
       p->rchild=null;t=p;
      }
   else if (x<t->key)searchinsert(x,      (2)      )
                  else    (3)
  }
```

1. 请阅读下列算法并填空。

```
int Partition (Elem R[], int low, int high)
{// 交换记录子序列 R[low..high]中的记录，使枢轴记录到位，并返回其所在位置，
此时，在它之前（后）的记录均不大（小）于它
   R[0] = R[low];
   pivotkey = R[low].key;
   while (low<high)
       {while(low<high&& R[high].key>=pivotkey)
        ____(1)____;
        R[low] = R[high];
        while (low<high && R[low].key<=pivotkey)
        ____(2)____;
        R[high] = R[low];
        }
   R[low] =____(3)____;
   return low;
} // Partition
```

3、设有一个表头为 head 的单链表。通过遍历一趟链表，将链表中所有结点按逆序链接。

```
typedef struct node
{int data;
 struct node *next;
}*pointer;
void invert(pointer head)
{ pointer p,u;
p=      (1)      ;
while (head!=NULL)
{u=head;
 head=head->next;
 u->next=      (2)      ;
 p=u;
}
head=p;
}
```

4、折半查找算法：

```
int binsrch(JD r[],int n,int k)
{   int low,high,mid,found;
    low=1;   high=n; found=0;
    while((low<=high)&&(found==0))
    {   mid=(low+high)/2;
        if(k>r[mid].key)      (1)      ;
        else if(k==r[mid].key)   found=1;
        else     high=mid-1;
    }
    if(found==1)
        return(      (2)      );
    else
        return(0);
}
```

五、设计题（每小题 10 分，共 30 分）

1. 设某单链表 L 的结点结构为 data,next 两个域,试用 C 语言定义其结构，并编写算法统计该链表的长度。

2.假设二叉树采用如下定义的存储结构：

```
typedef struct node {
int data;
struct node *lchild,*rchild;
}PBinTree;
```

其中，结点的 lchild 域和 rchild 域已分别填有指向其左、右孩子结点的指针。请编写一个将二叉树中所有结点的左右子树相互交换的函数。

```
void Bitree_Revolute(PBinTree  *T)
 {.....}
```

3. 假设线性表采用如下定义的存储结构：

```
typedef struct {
    KeyType key;
    infoType otherinfo;
} nodeType;
typedef nodeType SqList［MAXLEN］;
```

用C语言编写一个对 SqList 类型的变量 r 进行直接插入排序的算法。

试卷B评分标准

一、单项选择题（每小题1分，共10分）

题号	1	2	3	4	5	6	7	8	9	10
答案	A	A	B	B	C	C	C	A	B	C

二、填空题（每空2分，共20分）

1. 链式存储　　2. n*n　　3. 2*i

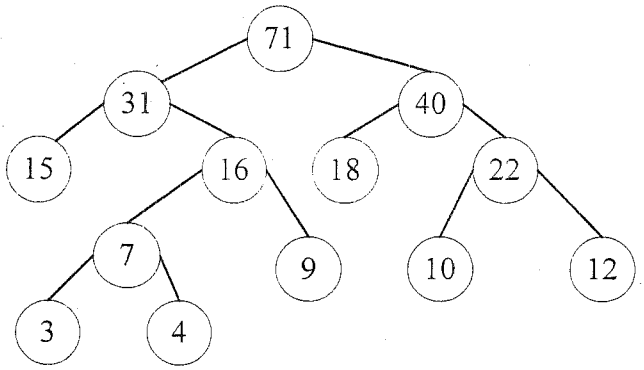
4. HL->NEXT==NULL　、　HL->NEXT==HL　　5. 双亲表示法

6 n-i+1　　7. n-1　　8. top==0　　top==m axsize

三、应用题（每小题5分，共40分）

1、EABF

2、　WPL=187



3、（1）ABFCDE

（2）

	A	B	C	D	E	F
A	0	1	0	0	1	0
B	1	0	0	0	0	1
C	0	0	0	1	0	1
D	0	0	1	0	0	1
E	1	0	0	0	0	0
F	0	1	1	1	0	0

4、

0	1	2	3	4	5	6	7	8	9	10	11	12
			29	94	18	32	46	70	48	75	63	25

四、算法填空和分析（每空2分，共20分）

1、（1）!t　　（2）t->lchild　　（3）searchinsert(x, t->rchild)

2、（1）____high--____ （2）____low++____ （3）____R[0]____

3、（1）____NU LL____ （2）____p____

4、（1）____low=mid+1____ （2）____mid____

五、设计题（每小题 10 分，共 30 分）

1、

```
}*pointer;
int fun(pointer L)
{   int count=0
    p=L;
   while(p!=N ULL)
        { i++;p=p->next;}
    Return(i);
}
```

2、

```
void Bitree_Revolute(PBinTree   *T)//交换所有结点的左右子树
{ PBinTree   *temp;
 temp= T->lchild;
 T->lchild= T->rchild;
 T->rchild= temp;
 if(T->lchild) Bitree_Revolute(T->lchild);
 if(T->rchild) Bitree_Revolute(T->rchild); //左右子树再分别交换各自的左右子树
}
```

3．

```
void zjinsert(redtype r[],int n)
{
    int i,j;
    for (i=2;i<=n;i++)                    /*从第二个记录开始插入*/
    {
        r[0]=r[i];                        /*设置一监视哨 r[0]，把待插入的记录值赋给
它*/
        j=i-1;                            /*从 j-1 起往前搜索*/
        while (r[0].key<r[j].key)         /*确定插入位置*/
        {
            r[j+1]=r[j];                  /*将第 j 个记录后移，直至 r[0].key≥r[j].key 为止
*/
            j--;
        }
        r[j+1]=r[0];                      /*插入待插的记录 r[0]*/
    }
}
```