北方工业大学

《数据结构》课程期末复习材料

（2016-2017 学年度）

一、 选择（填空）题（第一、二、三章）.....1
二、 选择（填空）题（第四、五、六章）.....3
三、 选择（填空）题（第七、九、十章）.....4
四、 判断题（第一、二、三章）.....5
五、 判断题（第四、五、六章）.....6
六、 判断题（第七、九、十章）.....6
七、 计算简答题（第二章）.....7
八、 计算简答题（第三章）.....9
九、 计算简答题（第四章）.....10
十、 计算简答题（第六章）.....11
十一、 计算简答题（第七章）.....12
十二、 计算简答题（第九章）.....15
十三、 计算简答题（第十章）.....16
答案解析.....18

北方工业大学计算机学院

二〇一六年十二月

一、 选择（填空）题（第一、二、三章）

1. 一个算法应该是（ ）。

A. 程序 B. 问题求解步骤的描述 C. 要满足五个基本特性 D. A 和 C_

2. 以下数据结构中，（ ）是非线性数据结构

A. 树 B. 字符串 C. 队 D. 栈

3. 下面关于线性表的叙述中，错误的是（ ）？

A. 线性表采用顺序存储，必须占用一片连续的存储单元。

B. 线性表采用顺序存储，便于进行插入和删除操作。

C. 线性表采用链接存储，不必占用一片连续的存储单元。

D. 线性表采用链接存储，便于插入和删除操作。

4. 在一个长度为 n 的顺序表中，在第 i（0<i<=n+1）个元素之前插入一个元素时，需向后移动（ ）个元素。

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

5. 若某线性表最常用的操作是存取任一指定序号的元素和在最后进行插入和删除运算，则利用（ ）存储方式最节省时间。

A. 顺序表 B. 双链表 C. 带头结点的双循环链表 D. 单循环链表_

6. 设一个链表最常用的操作是在末尾插入结点和删除尾结点，则选用( )最节省时间。

A. 单链表 B. 单循环链表 C. 带尾指针的单循环链表 D. 带头结点的双循环链表

7. 链表不具有的特点是（ ）

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

8. 线性表（ a1,a2,…,an）以链接方式存储时，访问第 i 位置元素的时间复杂性为（ ）

A. O（i） B. O（1） C. O（n） D. O（i-1）

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

A. O(0) B. O(1) C. O(n) D. O(n2)

10. 在一个单循环链表(长度为 n)中，已知 p 指针指向链表中一个非空结点，现要删除链表中 p 指针所指结点，其时间复杂度为（ ）。

A. O(n) B. O(1) C. O(n2) D. 不确定

11. 对于顺序存储的线性表，访问结点和增加、删除结点的时间复杂度为（ ）。

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

12. 非空的循环链表 head 的尾结点 p↑满足（ ）

A. p->next=head B. P->next=NULL C. p=NULL D. p= head

13. 带头结点的循环链表 L 为空的条件是（ ）。

A. L == NULL B. L->next ==NULL

C. L->next == L D. L->next == L->next

14. 在单链表指针为 p 的结点之后插入指针为 s 的结点，正确的操作是：（ ）

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

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

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

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

15. 在双向链表指针 p 的结点前插入一个指针 q 的结点操作是( )。

A. p->Llink=q;q->Rlink=p;p->Llink->Rlink=q;q->Llink=q;

B. p->Llink=q;p->Llink->Rlink=q;q->Rlink=p;q->Llink=p->Llink;

C. q->Rlink=p;q->Llink=p->Llink;p->Llink->Rlink=q;p->Llink=q;

D. q->Llink=p->Llink;q->Rlink=q;p->Llink=q;p->Llink=q;

16.若已知一个栈的入栈序列是 1,2,3,…,n, 其输出序列为 p1,p2,p3, …, pN,若 pN 是 n, 则 pi 是( )。

A. i　　B. n-i　　C. n-i+1　　D. 不确定

17. 一个栈的输入序列为 123…n, 若输出序列的第一个元素是 n, 输出第 i（1<=i<=n）个元素是（)。

A. 不确定　　B. n-i+1　　C. i　　D. n-i

18. 有六个元素 6，5，4，3，2，1 的顺序进栈，问下列哪一个不是合法的出栈序列？（ ）

A. 5 4 3 6 1 2　B. 4 5 3 1 2 6　　C. 3 4 6 5 2 1　　D. 2 3 4 1 5 6

19. 设栈的输入序列是 1，2，3，4,则（ ）不可能是其出栈序列。

A. 1，2，4，3，　B. 2，1，3，4，　C. 1，4，3，2，

D. 4，3，1，2，　E. 3，2，1，4

20.栈和队列的共同点是（)。

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

21. 假设以数组 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. (rear-front+m)%m　　B. rear-front+1

C. (front-rear+m)%m　　D. (rear-front)%m

22. 设循环队列存储空间的下标范围是 0..n-1，当队列尾指针为 rear（初始时 rear=0），队列长度为 len 时，循环队列中队头元素所在位置为_____。

A.　rear - len　　B.　rear　-　len +1

C.　(rear　-　len + 1) % n　　D. (rear　-　len + n) % n

23. 循环队列存储在数组 A[0..m]中，则入队时的操作为（)。

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

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

24.　最大容量为 n 的循环队列，队尾指针是 rear，队头是 front，则队空的条件是 （)。

A. (rear+1) MOD n=front　　B. rear=front

C. rear+1=front　　D. (rear-1) MOD n=front

25. 执行完下列语句段后，i 值为:（)

```
int f(int x)
{ int y;
 y=((x>0) ? x* f(x-1):2);
 printf("%d ",y);
 return y;
 }
int i ;
i =f(f(1));
```

A. 2　　B. 4　　C. 8　　D. 无限递归

26. 程序段： c = 0 ；

```
for( i=0 ; i<m ; i++ )
   for( j=0 ; j<n ; j++ )
    c = c + i*j ;
```

的时间复杂度为_____

A) O(m2) B) O(n2) C) O(m×n) D) O(m+n)

27. 若输入序列为 1，2，3，4，借助于一个输入受限的双向队列，不可能得到的输出序列为____。

A) 2，4，3，1 B) 3，1，2，4

C) 4，1，3，2 D) 4，2，3，1

二、 选择（填空）题（第四、五、六章）

1. 下面关于串的的叙述中，哪一个是不正确的？（ ）

A. 串是字符的有限序列 B. 空串是由空格构成的串 C. 模式匹配是串的一种重要运算 D. 串既可以采用顺序存储，也可以采用链式存储

2. 设有两个串 p 和 q，其中 q 是 p 的子串，求 q 在 p 中首次出现的位置的算法称为（ ）

A.求子串 B.联接 C.匹配 D.求串长

3. 已知串 S=‘aaab’,其 Next 数组值为（ ）

A. 0123 B. 1123 C. 1231 D. 1211 。

4. 串‘ababaaababaa’的 next 数组为（ ） 。

A. 012345678999 B 012121111212 C. 011234223456 D. 0123012322345

5.串的长度是指（ ） 。

A. 串中所含不同字母的个数 B. 串中所含字符的个数

C. 串中所含不同字符的个数 D. 串中所含非空格字符的个数

6.字符串‘ababaabab’ 的 nextval 为（ ） 。

A.(0,1,0,1,0,4,1,0,1) B. (0,1,0,1,0,2,1,0,1)

C. (0,1,0,1,0,0,0,1,1) D. (0,1,0,1,0,1,0,1,1)

7. 设有一个 10 阶的对称矩阵 A，采用压缩存储方式，以行序为主存储，a11 为第一元素，其存储地址为 1，每个元素占一个地址空间，则 a85 的地址为（ ）

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

8. 假设以行序为主序存储二维数组 A=array[1..100，1..100]，设每个数据元素占 2 个存储单元，基地址为 10，则 LOC[5，5]=（ ）

A.808 B.818 C.1010 D.1020

9.数组 A[0..5,0..6]的每个元素占五个字节，将其按列优先次序存储在起始地址为 1000 的内存单元中，则元素 A[5，5]的地址是( )。

A.1175 B.1180 C.1205 D.1210

10.二维数组 A 的每个元素是由 6 个字符组成的串，其行下标 i=0,1,…,8,列下标 j=1,2,…,10。若 A 按行先存储，元素 A[8,5]的起始地址与当 A 按列先存储时的元素（ ）的起始地址相同。设每个字符占一个字节。

A.A[8,5] B.A[3,10] C.A[5,8] D.A[0,9]

11.对稀疏矩阵进行压缩存储目的是（ ） 。

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

12.在具有 n 个结点的二叉树中，其空指针域的个数为_____。

A）2n+1　　B）n-1　　C）n+1　　D） 不确定

13.如果二叉树 T 是由多叉树 F 转换而成,那么树 F 的叶子在二叉树 T 中应是满足_____条件结点。

A）左子树为空且右子树非空　　B）左、右子树均为空

C）左子树为空　　D）右子树为空。

14.已知二叉树的后序序列为:FDEBHGCA,中序序列为:BFEDAGHC,则其先序序列为:_____。

A）ABEDFGCH　　B）ABCEGFDH

C）ABCEFDGH　　D）ABEFDCGH

15.一棵共有 23 个结点的二叉树中，树中除度为 2 的结点外其它均为叶子结点，那么该树中叶子结点的个数是_____个。

A）9　　B）10　　C）11　　D） 12

16.如果二叉树T是由多叉树F转换而成,那么树F的后根序列应该是二叉树T的_____。

A）层次遍历序列　　B）先序序列

C）中序序列　　D）后序序列

17.具有 64 个结点的完全二叉树的深度为_____。

A）5　　B）6　　C）7　　D）8

18.具有 3 个结点的二叉树共有_____种形态。

A）3　　B）4　　C）5　　D）6

19.若一棵二叉树具有 10 个度为 2 的结点,5 个度为 1 的结点,则度为 0 的结点个数是（　）

A） 9　　B） 11　　C） 15　　D）不确定

20.一棵共有 13 个结点的 4 叉树中，其中除叶子结点外其它结点的度均为 4，那么该树中叶子结点的个数是_____个。

A）8　　B）9　　C）10　　D） 11

21.一棵深度为 h 的二叉树，其结点个数最多为_____。

A）2^{h+1}　　B）2^{h-1}　　C）$2^{h}+1$　　D）$2^{h}-1$

三、 选择（填空）题（第七、九、十章）

1.下列排序方法中，不稳定的排序算法是_____。

A）冒泡排序　B）归并排序　C）快速排序　D）直接插入排序

2.具有 n 个顶点的无向图用邻接距阵表示,若该图为连通图,则其邻接距阵中至少有_____个非零元素。

A）2（n-1）　　B）n-1　　C）n*n　　D） n(n-1)

3.要连通具有 n 个顶点的有向图，至少需要_____条边。

A．n-1　B．n　　C．n+1　　D．2n

4.快速排序在最坏情况下的时间复杂度为_____。

A）O(1)　　B）O(n)　　C）O(n log2n)　　D） O(n2)

5.在长度为 12 的有序表中，按折半查找法对表进行查找，在表内各元素等概率情况下查找成功所需的平均比较次数为_____。

A）43/12　　B）39/12　　C）37/12　　D）35/12

6.下面哪一方法可以判断出一个有向图是否有环（回路） ___ 。

A) 深度优先遍历 B) 拓扑排序　C) 求最短路径　D)求关键路径

7.若对给定的关键字序列利用折半查找方法进行查找,则关键字序列需满足的条件是_____。

A）顺序存储且有序　B）顺序存储且升序　C）链式存储且有序　D）有序

8.下列排序方法中，某一趟结束后选出一个元素不一定能放在其最终位置上的是_____。

A) 堆排序　　B) 冒泡排序　　C) 快速排序　　D) 直接插入排序

9.下列四个序列中，满足堆的条件的是_____。

A)　75,65,30,15,25,45,20,10　　B) 75,65,45,10,30,25,20,15

C)　75,45,65,30,15,25,20,10　　D) 75,45,65,10,25,30,20,15

10.下列排序方法中，时间效率最高的排序算法是_____。

A）直接插入排序　B）归并排序　C）简单选择排序　D）冒泡排序

11.　在长度为 12 的有序表中，按折半查找法对表进行查找，在表内各元素等概率情况下查找成功的平均查找长度为_____。

A）43/12　　B）39/12　　C）37/12　　D）35/12

12.　对于给定的 8 个元素：34，76，45，18，26，54，92，65 按给定顺序生成一棵二叉排序树，该树的深度为_____。

A）4　　B）5　　C）6　　D） 7

13.　一棵 5 阶 B-树，除根结点以外的其它非终端结点中最少有____个关键字。

A）1　　B）2　　C）3　　D）4

四、　判断题（第一、二、三章）

1.　（　　）数据元素是数据的最小单位。

2.　（　　）健壮的算法不会因非法的输入数据而出现莫名其妙的状态。

3.　（　　）数据的逻辑结构是指数据的各数据项之间的逻辑关系。

4.　（　　）数据的逻辑结构说明数据元素之间的顺序关系，它依赖于计算机的储存结构。

5.　（　　）数据的物理结构是指数据在计算机内的实际存储形式。

6.　（　　）数据结构的抽象操作的定义与具体实现有关。

7.　（　　）顺序存储方式的优点是存储密度大，且插入、删除运算效率高。

8.　（　　）顺序存储方式插入和删除时效率太低，在这方面它不如链式存储方式好。

9.　（　　）顺序存储结构的主要缺点是不利于插入或删除操作。

10.　（　　）对任何数据结构链式存储结构一定优于顺序存储结构。

11.　（　　）取线性表的第 i 个元素的时间同 i 的大小有关。

12.　（　　）线性表、栈和队列都是线性结构。

13.　（　　）链表是采用链式存储结构的线性表,进行插入、删除操作时，在链表中比在顺序存储结构中效率高。

14.　（　　）线性表中每一个元素均存在唯一一个前驱和唯一一个后继。

15.　（　　）循环链表不是线性表。

16.　（　　）线性表的长度是线性表所占用的存储空间的大小。

17.　（　　）在单链表表示的线性表中，取线性表第 i 个元素操作的时间复杂度为 O(1)。

18.　（　　）删除带头结点单链表的第一个元素结点的时间复杂度是 O(1)。

19.　（　　）栈是实现过程和函数等子程序所必需的结构。

20.　（　　）栈是一种插入与删除操作都限定在表的一端进行的线性表。

21.　（　　）若输入序列为 1,2,3,4,5,6,则通过一个栈可以输出序列 3,2,5,6,4,1。

22.　（　　）在顺序存储结构表示的栈中删除一个元素时可能会引起栈内数据元素的移动。

23.　（　　）栈既可以采用顺序存储结构表示也可以采用链式存储结构表示。

24.　（　　）队列是一种插入与删除操作分别在表的两端进行的线性表,是一种先进后出型结构。

25. （　　）无论队列采用顺序存储结构还是采用链式存储结构,入队列和出队列操作的时间复杂度均为 O(1)。
26. （　　）循环队列就是用循环链表表示的队列。
27. （　　）队列和栈都是运算受限的线性表，只允许在表的两端进行运算。

五、 判断题（第四、五、六章）

1. （　　）串是一种数据对象和操作都特殊的线性表。
2. （　　）KMP 算法的特点是在模式匹配时指示主串的指针不会变小。
3. （　　）设模式串的长度为 m,目标串的长度为 n,当 n≈m 且处理只匹配一次的模式时，朴素的匹配（即子串定位函数）算法所花的时间代价可能会更为节省。
4. （　　）数组可看成线性结构的一种推广，因此与线性表一样，可以对它进行插入，删除等操作。
5. （　　）数组不适合作为任何二叉树的存储结构。
6. （　　）从逻辑结构上看，n 维数组的每个元素均属于 n 个向量。
7. （　　）稀疏矩阵压缩存储后，必会失去随机存取功能。
8. （　　）一个稀疏矩阵 Am*n 采用三元组形式表示， 若把三元组中有关行下标与列下标的值互换，并把 m 和 n 的值互换，则就完成了 Am*n 的转置运算。
9. （　　）一棵多叉树转换成二叉树，该二叉树的根结点一定没有右子树。
10. （　　）二叉树只能用链式结构存储，无法用顺序存储结构存储。
11. （　　）完全二叉树中最多只能有一个度为 1 的结点。
12. （　　）完全二叉树中一定不存在度为 1 的结点。
13. （　　）具有 n 个结点的二叉树其深度一定小于 n。

六、 判断题（第七、九、十章）

1. （ ）选择排序是一种稳定的排序方法。
2. （ ）一个有向无环图拓扑序列可能不唯一。
3. （ ）一棵二叉排序树的先序序列一定是有序序列。
4. （ ）在 n 个结点的无向图中，若边数大于 n-1,则该图必是连通图。
5. （ ）完全二叉树中一定不存在度为 1 的结点。
6. （)有向图中第 i 个顶点的出度等于其邻接矩阵中第 i 行非 0 元素的个数
7. （ ）简单选择排序的比较次数与待排序列的初始排列顺序无关。
8. （ ）二叉排序树的中序序列一定是一个有序序列。
9. （ ）快速排序是稳定排序。
10. （ ）当初始待排关键字排列为正序时，简单选择排序的比较次数达到最少。
11. （ ）当初始待排关键字排列为正序时，直接插入排序的比较次数达到最少。
12. （ ）二叉树只能用链式结构存储，无法用顺序存储结构存储。
13. （ ）B-树中的所有叶子结点均在同一层上。
14. （ ）一棵 9 阶 B-树中的所有非终端结点的分支数一定大于 4。
15. （ ）所谓动态查找是指查找表在查找后可能会发生变化。
16. （)平衡二叉树是指二叉树根结点的左子树深度和右子树深度之差的绝对值不大于 1。
17. （ ）在二叉排序树中删除结点时，只能删除树中的叶子结点。
18. （ ）完全二叉树一定是一棵平衡二叉树。
19. （ ）具有 n 个顶点和至少 n-1 条边无向图一定是一个连通图。
20. （ ）用邻接矩阵法存储一个图所需的存储单元数目与图的边数有关。

七、 计算简答题（第二章）

2.11 设顺序表 va 中的数据元素递增有序。试写一算法，将 x 插入到顺序表的适当位置上，以保持该表的有序性。

```
Status Insert_SqList(SqList &va,int x)//把 x 插入递增有序表 va 中
{
 if(va.length+1>va.listsize) return ERROR;
 for(i=va.length-1;va.elem[i]>x&&i>=0;i--)
  va.elem[i+1]=va.elem[i];
 va.elem[i+1]=x;
 va.length++;
return OK;
}//Insert_SqList_
```

2.15 已知指针 ha 和 hb 分别指向两个单链表的头结点，并且已知两个链表的长度分别为 m 和 n。试写一算法将这两个链表连接在一起(即令其中一个表的首元结点连在另一个表的最后一个结点之后)，假设指针 hc 指向连接后的链表的头结点，并要求算法以尽可能短的时间完成连接运算。请分析你的算法的时间复杂度。

```
void ListConcat(LinkList &ha, int    m, LinkList &hb, int    n, LinkList &hc )
{
    if ( m>n) { p = hb ; q = ha ; }
       else { p = ha ; q = hb ; }
    ha = hb = hc = p ;
  while ( p->next )   p=p->next;
    p->next = q->next ;
    free ( q ) ;
}//ListConcat
```

时间复杂度：O(min(m,n))

2.19 已知线性表中的元素以值递增有序排列，并以单链表作存储结构。试写一高效的算法，删除表中所有值大于 mink 且小于 maxk 的元素（若表中存在这样的元素），同时释放被删结点空间，并分析你的算法的时间复杂度（注意，mink 和 maxk 是给定的两个参变量，它们的值可以和表中的元素相同，也可以不同）。

```
Status Delete_L(Linklist &L,ElemType mink, ElemType maxk)
{ if ( mink >= maxk )   return   ERROR ;
  p=L;
 while(p->next->data<=mink && p->next) p=p->next;
                    //p 是最后一个不大于 mink 的元素
 if(p->next)   //如果还有比 mink 更大的元素
 {
  q=p->next;
  while(q->data<maxk && q->next)
  { r=q;
    q=q->next;
     //q 是第一个不小于 maxk 的元素
        free(r);
      }
  p->next=q;
 }
   return OK ;
}//Delete_L
```

2.21 试写一算法，实现顺序表的就地逆置，即利用原表的存储空间将线性表$(a_1,a_2,\cdots,a_n)$逆置为$(a_n,a_{n-1},\cdots,a_1)$。

```
void reverse(SqList &A)//顺序表的就地逆置
{
   int i,j,temp;
   for(i=0,j=A.length-1;i<j;i++,j--)
   {
      temp=A.elem[i];
      A.elem[i]=A.elem[j];
      A.elem[j]=temp;
   }
}_
```

2.33 已知由一个线性链表表示的线性表中含有三类字符的数据元素（如：字母字符、数字字符和其他字符），试编写算法将该线性表分割为三个循环链表，其中每个循环链表表示的线性表中均只含一类字符。

```
Status L_Divide(LinkList &L, LinkList &A, LinkList &B, LinkList &C)
{ if (!(A=(LinkList)malloc(sizeof(LNode)))) return ERROR;
   if (!(B=(LinkList)malloc(sizeof(LNode)))) return ERROR;
   if (!(C=(LinkList)malloc(sizeof(LNode))))    return ERROR;
   A->next = A ; B->next = B ; C->next = C ;
     p=A ; q=B ; r=C ;
     s = L->next ;
   while(s)
   {
      if((s->data>='a'    &&    s->data<='z')||(s->data>='A'    &&
s->data<='Z'))
      { p->next=s;p=s; }
      else if((s->data>='0' && s->data<='9'))
{ q->next=s;q=s; }
                  else
{r->next=s;r=s; }
       s=s->next;
    }//while

p->next=A;q->next=B;r->next
=C; //完成循环链表
   return OK;
}//LinkList_Divide
```

2.38 设有一个双向循环链表，每个结点中除有 prior，data 和 next 三个域外，还增设了一个访问频度域 freq。在链表被起用之前，频度域 freq 的值均初始化为零，而每当对链表进行一次 LOCATE(L,x)的操作后，被访问的结点（即元素值等于 x 的结点）中的频度域 freq 的值便增 1，同时调整链表中结点之间的次序，使其按访问频度非递增的次序顺序排列，以便始终保持被频繁访问的结点总是靠近表头结点。试编写符合上述要求的 LOCATE 操作的算法。

```
Status Locate(DuLinkList &L,ElemType x)
{ p=L->next;
   while(p!=L && p->data!=x )    p=p->next ;
   if(p==L) return ERROR;
   p->freq++;q=p->prior;
   while(q!=L && q->freq<p->freq) q=q->prior;
   if (q!=p->prior)
    { p->prior->next=p->next;p->next->prior=p->prior;
          q->next->prior=p;p->n
ext=q->next;
       q->next=p;p->prior=q;
     }
   return OK ;
}//Locate_DuList
```

八、 计算简答题（第三章）

3.1 若按教科书 3.1.1 节中图 3.1(b)所示铁道进行车厢调度（注意：两侧铁道均为单向行驶道），则请回答： (1) 如果进站的车厢序列为 123，则可能得到的出站车厢序列是什么？

(2) 如果进站的车厢序列为 123456，则能否得到 435612 和 135426 的出站序列，并请说明为什么不能得到或者如何得到（即写出以 ‘S’表示进栈和以 ‘X’表示出栈的栈操作序列）。

答：(1) 123 231 321 213 132

(2) 可以得到 135426 的出站序列，但不能得到 435612 的出站序列。因为 4356 出站说明 12 已经在栈中，1 不可能先于 2 出栈。

3.3 写出下列程序段的输出结果（栈的元素类型 SElemType 为 char）。

```
void main()
{
Stack S;
char x,y;
InitStack(S);
x=‘c’; y=‘k’;
Push(S,x);Push(S,‘a’);Push(S,y);
Pop(S,x);Push(S,‘t’);Push(S,x);
Pop(S,x);Push(S,‘s’);
while(!StackEmpty(S)) { Pop(S,y); printf(y); }
printf(x);
}
```

答： stack

3.4 简述以下算法的功能（栈的元素类型 SElemType 为 int）。

```
(1) status algo1(Stack S){
      int i,n,A[255];
      n=0;
      while(!StackEmpty(S))
     {n++; Pop(S,A[n]); }
  for(i=1;i<=n;i++) Push(S,A[i]);
     }
```

```
(2) status algo2(Stack S,int e){
    Stack T; int d;
    InitStack(T);
    while(!StackEmpty(S)){
        Pop(S,d);
        if(d!=e) Push(T,d);}
    while(!StackEmpty(T)){
        Pop(T,d);
        Push(S,d);}
    }
```

答： (1) 栈中的数据元素逆置 (2) 如果栈中存在元素 e，将其从栈中清除

3.16

假设如题 3.1 所属火车调度站的入口处有 n 节硬席或软席车厢（分别以 H 和 S 表示）等待调度，试编写算法，输出对这 n 节车厢进行调度的操作（即入栈或出栈操作）序列，以使所有的软席车厢都被调整到硬席车厢之前。

```
void Train_arrange(char *train)//这里用字符串 train 表示火车,'H'表示硬席,'S'表示软席
{ p=train;q=train;
 InitStack(s);
 while(*p)
 { if(*p=='H') push(s,*p); //把'H'存入栈中
   else *q++=*p; //把'S'调到前部
   p++;}
 while(!StackEmpty(s))
 {pop(s,c);
  *q++=c; //把'H'接在后部
 }
}//Train_arrange_
```

3.28 假设以带头结点的循环链表表示队列，并且只设一个指针指向队尾元素结点（注意不设头指针），试编写相应的队列初始化、入队列和出队列的算法。

（一）

```
void InitCiQueue(LinkList &Q)//初始化循环链表表示的队列 Q
{
 Q=(LinkList)malloc(sizeof(LNode));
 Q->next=Q;
}//InitCiQueue
void EnCiQueue(LinkList &Q,int x)//把元素 x 插入循环链表表示的队列 Q,Q 指向队尾元素,Q->next 指向头结点,Q->next->next 指向队头元素
{
 p=(LinkList)malloc(sizeof(LNode));
 p->data=x;
 p->next=Q->next; //直接把 p 加在 Q 的后面
 Q->next=p;
 Q=p; //修改尾指针
}
```

（二）

```
Status DeCiQueue(LinkList &Q,int x)//从循环链表表示的队列 Q 头部删除元素 x
{
 if(Q==Q->next) return ERROR; //队列已空
 p=Q->next->next;
 x=p->data;
 Q->next->next=p->next;
 free(p);
 return OK;
}//DeCiQueue_
```

3.30 假设将循环队列定义为:以域变量 rear 和 length 分别指示循环队列中队尾元素的位置和内含元素的个数。试给出此循环队列的队满条件，并写出相应的入队列和出队列的算法（在出队列的算法中要返回队头元素）。

（一）

```
#define     MAXQSIZE    100
typedef    struct {
      QElemType      *base ;
      int      rear ;
      int      length ;   //替代书中循环队列标准定义的 int front;
} SqQueue ;
Status InitQueue( SqQueue    &Q )
{           Q.base          =          (QElemType * )malloc(MAXQSIZE*sizeof(QElemType));
   if(!Q.base) return OVERFLOW ;
   Q.rear = Qlength = 0 ;
   return OK ;
}
```

（二）

```
Status EnQueue(SqQueue &Q , QElemType    x)
{ if(Q.length==MAXSIZE) return OVERFLOW;
  Q.base[Q.rear]=x;
    Q.rear=(Q.rear+1)%MAXSIZE;
  Q.length++;
 return OK;
}//EnQueue
Status DeQueue(SqQueue &Q , QElemType &x)
{ if(Q.length==0) return ERROR;
 head=(Q.rear - Q.length + MAXSIZE)%MAXSIZE;
 x=Q.base[head];
 Q.length--;
 } //DeCyQueu
```

九、 计算简答题（第四章）

求下列串的 next[]的值 nextval[]的值

(1) abababaababaa

(2) abababab

答案:

	abababaababaa	abababab
next[]的值:	011234223456	011234234
nextval[]的值:	010104210104	010104101

十、 计算简答题（第六章）

6.5 已知一棵度为 k 的树中有 n1 个度为 1 的节点，n2 个度为 2 的节点，…，nk 个度为 k 的节点，问该树中有多少个叶子节点。

答案：n=n1+2*n2+…+k*nk+1

n=n0+n1+n2+…+nk

所以 n0=n2+2n3+…+(k-1)*nk+1

6.41 编写递归算法，在二叉树中求位于先序序列中第 k 个位置的结点的值。

```
int count=0;_
int Get_PreSeq(Bitree T, int k)   //函数返回值表示是否找到
 {
   if(!T)return 0;
   else
   {       count++;
             if(count==k)
                 {    printf("Value is %c\n",T->data);
                      return 1; }
           else {
                  if(Get_PreSeq(T->lchild,k))return 1;
                 else return(Get_PreSeq(T->rchild,k));
                }
    }
```

6.42 编写递归算法，计算二叉树中叶子结点的数目。

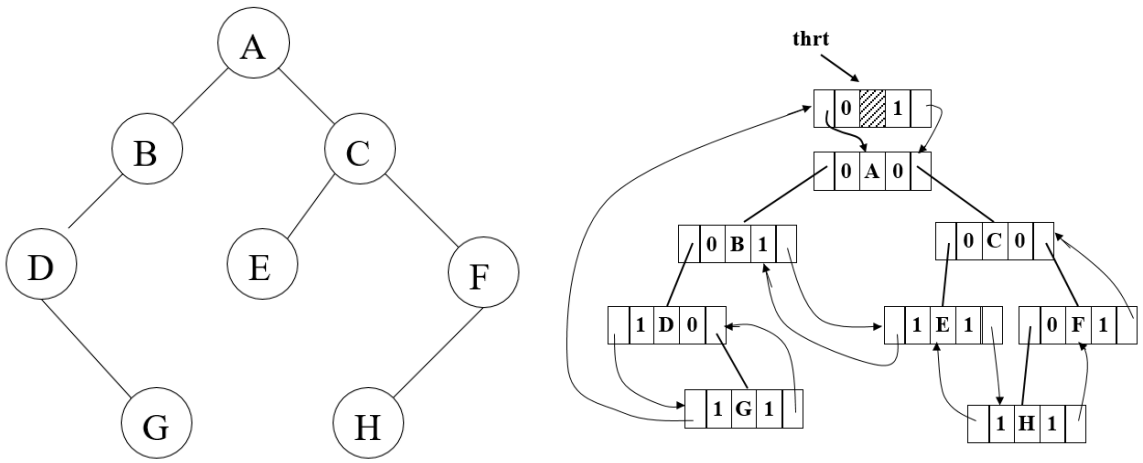
```
int LeafNum(Bitree T)
{ if(!T) return(0);
   else if(!T->lchild && !T->rchild) return(1);
              else return(LeafNum(T->lchild)+LeafNum(T->rchild));
}
```

6.43 编写递归算法，将二叉树中所有节点的左、右子树相互交换。_

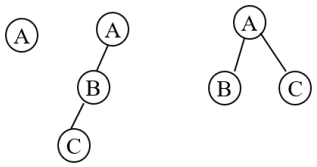
```
void extree(Bitree &T)
{ if ( T ) {
       Bitree P;
       P=T->rchild;
       T->rchild=T->lchild;
        T->lchild=P;
       extree((T->lchild);
       extree((T->rchild);
     }
}
```

（可用先序或后序，但不能用中序）

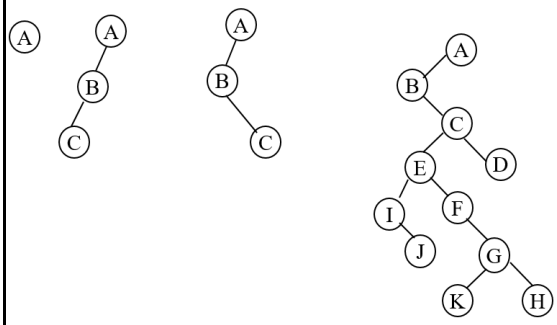
6.15 请对右图所示二叉树进行后序线索化，为每个空指针建立相应的前驱和后继线索。



6.19　分别画出和下列树对应的各个二叉树：



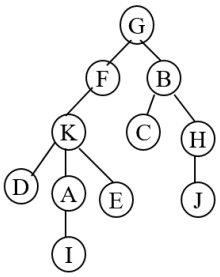
答案:



6.23　画出和下列已知序列对应的树 T：

树的先根次序访问序列为：GFKDAIEBCHJ；

树的后根次序访问序列为：DIAEKFCJHBG。

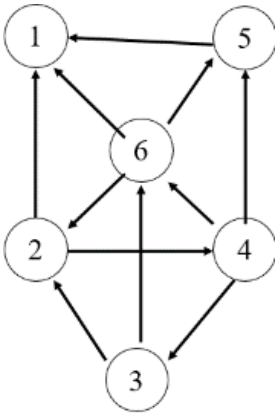


十一、　计算简答题（第七章）

7.1 已知左图所示的有向图，请给出该图的

（1）每个顶点的入/出度（2）邻接矩阵

（3）邻接表



答案：（1）每个顶点的入度：

ID(1)=3　　ID(2)=2　　ID(3)=1

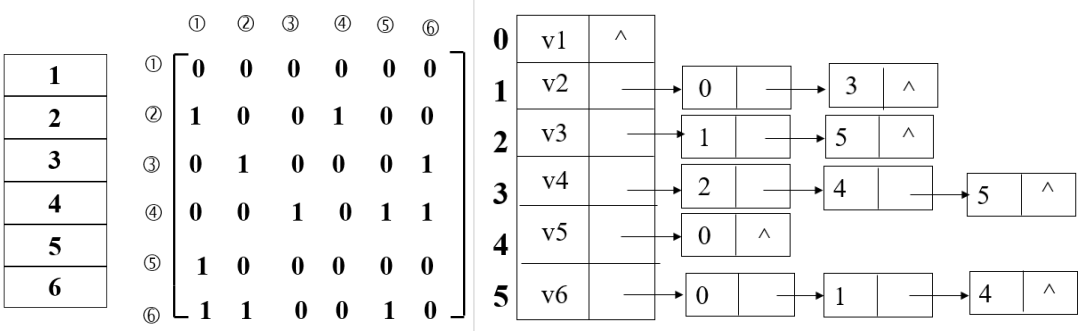
ID(4)=1　　ID(5)=2　　ID(6)=2

每个顶点的出度：

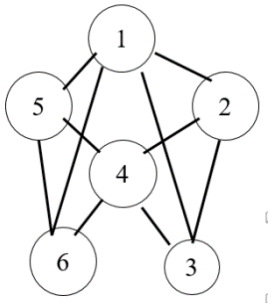
OD(1)=0　　OD(2)=2　　OD(3)=2

OD(4)=3　　OD(5)=1　　OD(6)=3

（2）邻接矩阵　　（3）邻接表



7.3 列出深度优先和广度优先搜索遍历该图所得顶点序列和边的序列。



答案：深度优先遍历：

顶点序列：1 2 3 4 5 6

边的序列：(1,2), (2,3), (3,4), (4,5), (5,6)

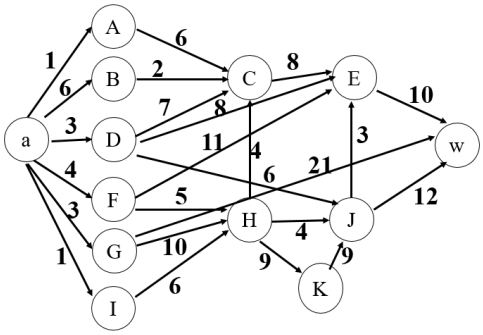
广度优先遍历：

顶点序列：1 2 3 5 6 4

边的序列：(1,2), (1,3), (1,5), (1,6), (2,4)

7.10

对于如图所示的 AOE 网络，计算各活动弧的 e(ai)和 l(ai)函数值，各事件（顶点）的 ve(vi)和 vl(vi)函数值；列出各条关键路径。



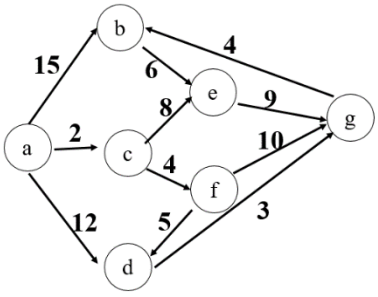
答案:

顶点	ve	vl
a	0	0
A	1	20
B	6	24
C	17	26
D	3	19
E	34	34
F	4	8
G	3	3
H	13	13
I	1	7
J	31	31
K	22	22
w	44	44

边	e	j	j-e
(a,A)	0	19	19
(a,B)	0	18	18
(a,D)	0	16	16
(a,F)	0	4	4
(a,G)	0	0	0
(a,I)	0	6	6
(A,C)	1	20	19
(B,C)	6	24	18
(D,C)	3	19	16
(D,E)	3	26	23
(D,J)	3	25	22
(F,E)	4	23	19
(F,H)	4	8	4
(G,w)	3	23	20
(G,H)	3	3	0
(I,H)	1	7	6
(C,E)	17	26	9
(H,C)	13	22	9
(H,J)	13	27	14
(H,K)	13	13	0
(K,J)	22	22	0
(J,E)	31	31	0
(J,w)	31	32	1
(E,w)	34	34	0

关键路径只有一条：(a,G,H,K,J,E,w)

7.11 试利用 Dijkstra 算法求图中从顶点 a 到其他各顶点之间的最短路径，写出执行算法过程中各步的状态。



答案:从顶点 a 到其他各点的最短路径的求解过程如下:

终点	b	c	d	e	f	g	终点集
K=1	15 (a, b)	**2** **(a, c)**	12 (a, d)				{a, c}
K=2	15 (a, b)		12 (a, d)	10 (a, c, e)	**6** **(a, c, f)**		{a, c, f}
K=3	15 (a, b)		11 (a, c, f, d)	**10** **(a, c, e)**		16 (a, c, f, g)	{a, c, f, e}
K=4	15 (a, b)		**11** **(a, c, f, d)**			16 (a, c, f, g)	{a, cf, e, d}
K=5	15 (a, b)					**14** **{a, c, f, e, d, g}**	{a, c, f, e, d, g}
K=6	**15** **(a, b)**						{a, c, f, e, d, g, b}

练习：已知图的邻接矩阵为：

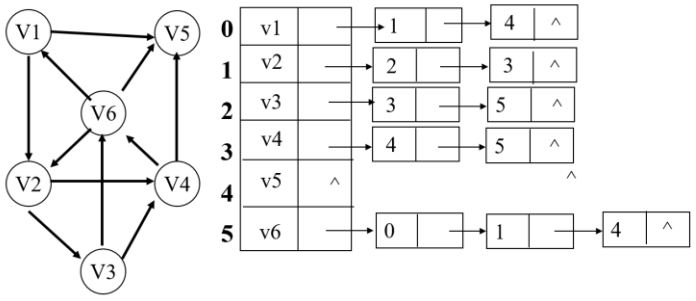
1）画出该图邻接表。

2）写出该图按以上邻接矩阵表示时，从 V1 出发的深度优先遍历序列和广度优先遍历序列。

0	V1
1	V2
2	V3
3	V4
4	V5
5	V6

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

答案：邻接表为：



深度优先遍历序列：V1 V2 V3 V4 V5 V6

广度优先遍历序列：V1 V2 V5 V3 V4 V6

十二、 计算简答题（第九章）

1. 一棵以二叉链表存储的二叉排序树，其存储结构为：

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

树中数据域为大于 0 整型数，并且可能存在值相同的结点。设计一个算法，按升序打印出树中所有结点，但相同值的结点只打印一次。

答案：

```
void  PrnNode ( BiTree  T ,  int  &predata)
{ if( T ) {
    PrnNode ( T->lchild, predata) ;
    if( T->data != predata )
        printf( "%d  ",T->data ) ;
    predata = T->data ;
    PrnNode ( T->rchild, predata ) ;
  }
}
void  PrnMain(BiTree  T)
{  predata = -1;
   PrnNode(T , predata ) ;
}
```

2. 已知一棵二叉排序树，以二叉链表形式存储，其数据域值为大于 0 的整型数，树中可能存在数据域值相同的结点。设计一个算法，统计树中不同结点的个数。

数的存储结构为：

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

答案：

```
void  incount(BiTree  T, int &c, int  &predata)
{ if(T) {
incount(T->lchild , c , predata) ;
if(T->data!=predata)  c++ ;
predata= T->data ;
incount(T->rchild , c , predata ) ;
}
int  count(BiTree  T)
{  predata = 0; count=0 ;
incount(T , count , predata ) ;
return(count);
}
```

9.32 已知一棵二叉排序树上所有关键字中的最小值为-max，最大值为 max，又-max<x<max。编写递归算法，求该二叉排序树上的小于 x 且最靠近 x 的值 a 和大于 x 且最靠近 x 的值 b。

答案：

```
int last=0;
void MaxLT_MinGT(BiTree T,int x)
//找到二叉排序树 T 中小于 x 的最大元素和大于 x 的最小元素
{
 if(T){
    if(last<x&&T->data>=x)
//找到了小于 x 的最大元素
          printf("a=%d\n",last);
      if(last>=x){
            last=T->data;MaxLT_MinGT(T->lchild,x);
      }
   if(last<=x&&T->data>x)
//找到了大于 x 的最小元素
        printf("b=%d\n",T->data);
    if(T->data<=x){
            last=T->data; MaxLT_MinGT(T->rchild,x);
    }
 }
}//MaxLT_MinGT_
```

9.33 编写递归算法，从大到小输出给定二叉排序树中所有关键字不小于 x 的数据元素。

答案：

```
void Print_NLT(BiTree T,int x)//从大到小输出二叉排序树 T 中所有不小于 x 的元素
{
 if(T)
        { Print_NLT(T->rchild,x);
          if(T->data>=x)
                {   printf("%d\n",T->data);
                    Print_NLT(T->lchild,x); //先右后左的中序遍历
                }
        }
}//Print_NLT_
```

十三、 计算简答题（第十章）

设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45 } 。说明分别采用直接插入排序、希尔排序、冒泡排序、快速排序、直接选择排序、堆排序、归并排序、基数排序方法进行排序的过程。

1. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用直接插入排序方法进行排序的过程。

（1） （3）87 12 61 70 97 26 45
（2） （3 87）12 61 70 97 26 45
（3） （3 12 87）61 70 97 26 45
（4） （3 12 61 87）70 97 26 45
（5） （3 12 61 70 87）97 26 45
（6） （3 12 61 70 87 97）26 45
（7） （3 12 26 61 70 87 97）45
（8） （3 12 26 45 61 70 87 97）

2. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用希尔排序方法进行排序的过程。

（1）D=5 3 26 12 61 70 97 87 45
（2）D=3 3 26 12 61 45 97 87 70
（3）D=1 3 12 26 45 61 70 87 97

3. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用冒泡排序方法进行排序的过程。

（1） 3 12 61 70 87 26 45 (97)
（2） 3 12 61 70 26 45 (87 97)
（3） 3 12 61 26 45 (70 87 97)
（4） 3 12 26 45 (61 70 87 97)
（5） (3 12 26 45 61 70 87 97)

4. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用快速排序方法进行排序的过程。

分别完成第二趟排序： 3 (45 12 61 70 26) 87 (97)
分别完成第二趟排序： 3 (45 12 61 70 26) 87 (97)
分别完成第三趟排序： 3 (26 12）45 （70 61) 87 97
分别完成第四趟排序： 3 12 26 45 61 70 87 97

5. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用直接选择排序方法进行排序的过程。

（1） （3）87 12 61 70 97 26 45
（2） （3 12）87 61 70 97 26 45
（3） （3 12 26）61 70 97 87 45
（4） （3 12 26 45）70 97 87 61
（5） （3 12 26 45 61）97 87 70
（6） （3 12 26 45 61 70）87 97
（7） （3 12 26 45 61 70 87）97
（8） （3 12 26 45 61 70 87 97）

6. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用堆排序方法进行排序的过程。

建大堆: 97 87 26 61 70 12 3 45_
（1） 87 70 26 61 45 12 3（97）
（2） 70 61 26 3 45 12 （87 97）
（3） 61 45 26 3 12（70 87 97）
（4） 45 12 26 3（61 70 87 97）
（5） 26 12 3（45 61 70 87 97）
（6） 12 3（26 45 61 70 87 97）
（7） 3（12 26 45 61 70 87 97）_
（8） （3 12 26 45 61 70 87 97）

7. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用归并排序方法进行排序的过程。

采用二路归并
（1） (3 87) (12 61) (70 97) (26 45)
（2） (3 12 61 87) (26 45 70 97)
（3） （3 12 26 45 61 70 87 97）

8. 设待排序的表有 8 个记录,其关键字分别为{3，87，12，61，70，97，26，45} 。说明采用基数排序方法进行排序的过程。

（1）按个位排序 70 61 12 3 45 26 87 97
（2）按十位排序 3 12 26 45 61 70 87 97

答案解析

一、选择（填空）题（第一、二、三章）答案：

1.B 2.A 3.B 4.B 5.A 6.D 7.B 8.C 9.C 10.A 11.C 12.A 13.C 14.B 15.C 16.D
17.B 18.C 19.D 20.C 21.C 22.D 23.D 24.B 25.B 26.C 27.D

二、选择（填空）题（第四、五、六章）答案：

1.B 2.C 3.A 4.C 5.B 6.A 7.B 8.B 9.A 10.B 11.C 12.C 13.C 14.D 15.D 16.C
17.C 18.C 19.B 20.C 21.D 9. (5*6+5)*5+1000=1175
10. 8*10+4=(y-1)*9+x => y=10, x=3 17.($n=2^K-1$)

三、选择（填空）题（第七、九、十章）答案：

1C 2A 3B 4D 4C 5B 6A 7D 8C 9B 10C 11B 12B
13. 至少有$\lceil m/2 \rceil-1=\lfloor (m-1)/2 \rfloor$个关键字

四、判断题（第一、二、三章）

1.X 2.√ 3.X 4.X 5.√ 6.X 7.X 8.√ 9.√ 10.X
11.X 12.√ 13.√ 14.X 15.X 16.X 17.X 18.√ 19.√ 20.√
21.√ 22.X 23.√ 24.X 25.√ 26.X 27.X

五、判断题（第四、五、六章）

1.√ 2.√ 3. √ 4.X 5. X 6. √ 7. √ 8. X 9.√ 10.X 11. √ 12. X 13. X

六、判断题（第七、九、十章）

1X 2√ 3X 4X 5X 6√ 7√ 8√ 9X 10X
11√ 12X 13√ 14X 15√ 16√ 17X 18√ 19X 20X