**数据结构练习题**

说明：除题中另有说明外，本书均采用下面的存储结构及操作

**二叉链表的C语言存储描述如下 单向链表的存储**

typedef struct BiTNode typedef struct LNode

{char data； //元素信息 {ElemType data；

struct BiTNode \*lchild，rchild； struct LNode \*next；

}BiTNode，\*BiTree； }LNode，\*LinkList；

**邻接表的C语言存储描述如下 基本操作如下**

typedef struct Arc InitStack(S)

{int adj； //顶点序号 StackEmpty(S)

struct Arc \*next； Push(S，e)

}ArcNode，\*ARC； //表结点 Pop(S，e)

typedef struct Vnode{ //头结点 InitQueue（Q）

char data； //元素信息 QueueEmpty（Q）

ARC \*first； Enqueue（Q，x）

}Graph[N]； //N为顶点的实际个数 DelQueue（Q，x）

一、判断正误

1、线性表采用链式存储，则链表中结点总数一定等于元素总数。

2、栈是一种后进先出的线性表，因此，元素的进栈序列和出栈序列不可能相同。

3、无论采用何种存储结构，队列的入队、出队次序一定相同。

4、由三个不同关键字构造的二叉排序树的基本形态共有五种。

5、若非空二叉树的先序、中序遍历序列相同，则必为仅有一个根的二叉树。

6、二叉链表存储的二叉树中结点总数为n，则空指针域总数必为n+1。

7、n(>0)个权值作为叶子构造的哈夫曼树中结点总数为2n-1。

8、森林的后序遍历与它对应的二叉树的中序遍历相同。

9、折半查找的平均查找效率高于顺序查找，因此，查找某一元素时，折半查找所需与关键字的比较次数一定少于顺序查找。

10、二叉排序树的中序遍历序列按关键字递增有序。

11、有向图的邻接表的第i个链表中表结点总数为该顶点的度。

12、顺序表，链表，二叉链表，邻接表，散列表均为存储结构。

13、选取散列函数时，冲突越少越好。

14、冒泡，直接插入，归并，基数排序均为稳定排序。

15、快速排序最坏的时间复杂度为O（N\*logN）。

16、不稳定排序后，相同关键字元素的相对位置一定发生改变。

17、直接插入排序适用于元素个数较少或初始序列基本有序情况。

18、对称矩阵压缩存储后不具备随机存取的特点。

19、在考虑存储结构的前提下，有向图的拓扑排序序列一定唯一。

20、折半查找要求元素必须按关键字有序，且只能采用顺序存储结构。

二、填空题

数据的基本单位是\_\_\_\_\_，数据结构常见的逻辑结构有\_\_\_\_\_，存储结构有\_\_\_\_。

数据结构的逻辑结构可视为一个二元组Data\_Structure=（D，S），其中D是\_\_\_\_\_，S是\_\_\_\_\_。

数据结构的抽象数据类型ADT是指一个数学模型及定义在该模型上的一组操作。它可用三元组（D，S，P）表示，其中D是数据对象，S是\_\_\_\_\_，P是\_\_\_\_\_

设计一个好的算法应达到的目标是正确性、\_\_\_\_\_、\_\_\_\_\_和高效率低存储量。算法时间效率、空间效率的度量分别称为\_\_\_\_\_、\_\_\_\_\_。

5、顺序表、链表分别通过\_\_\_\_\_、\_\_\_\_\_体现元素的前驱、后继的逻辑关系。

6、在表长为n的顺序表中插入、删除元素时，在等概率条件下平均移动元素的次数分别为\_\_\_\_\_、\_\_\_\_\_，具体移动的元素个数与\_\_\_\_\_有关。

7、已知L为头指针的带头结点的单循环链表（指针域为next），a．若R为尾指针，则满足的条件为\_\_\_\_\_\_，b．若该表为空表，则满足的条件是\_\_\_\_\_\_。

8、已知五个元素ABCDE的进栈次序为ABCDE，a．若C为第一个出栈元素，则下一个出栈的元素可能是\_\_\_\_\_；b．能否得到出栈序列CDBEA\_\_\_\_\_\_、CEDAB\_\_\_\_\_\_？

9、若三个元素ABC的进栈次序为ABC，则借助栈一共可以得到\_\_\_\_\_\_中出栈序列？

10、已知顺序存储的循环队列中，f，r分别为队头、队尾指针，MAX为队列中存储单元的最大个数。若当队列中仅有一个空闲单元时视为队满，则队空、队满条件分别为\_\_\_\_\_、\_\_\_\_\_；一般情况下，队列中元素个数可表示为\_\_\_\_\_ 。

11、广义表A=((a，(b)，c)，(d)，(e，f，g))的表长为\_\_\_\_\_，深度为\_\_\_\_\_，利用取表头、表尾运算表示原子e的表达式为\_\_\_\_\_。

12、广义表中元素既可以是单原子，也可以是\_\_\_\_\_，广义表的两个基本操作是\_\_\_\_\_和\_\_\_\_\_。

13、10个结点二叉树至多有\_\_\_\_\_个叶子，最低高度为\_\_\_\_\_。

14、83个结点的完全二叉树高度为\_\_\_\_\_，其中叶子个数为\_\_\_\_\_，一度结点个数为\_\_\_\_\_。

15、n(n>0)个结点的完全二叉树叶子个数为\_\_\_\_\_，一度结点个数为\_\_\_\_\_。

16、高度为5的BST树结点总数至多，至少为\_\_\_\_\_，\_\_\_\_\_。

17、n个叶子的哈夫曼树最高高度为\_\_\_\_\_，结点总数为\_\_\_\_\_。

18、从本质上而言，森林的孩子兄弟链表与它转换所对应的二叉树的\_\_\_\_\_存储结构一致。

19、n个顶点的无向、有向完全图的边数分别是\_\_\_\_\_、\_\_\_\_\_，n个顶点的强连通图至少有\_\_\_\_\_条边。

20、无向网的最小生成树的构造算法是\_\_\_\_\_和\_\_\_\_\_；n个顶点的无向网构造的最小生成树中边的条数为\_\_\_\_\_。

21、构造散列表要解决的两个问题是\_\_\_\_\_；若将关键字序列的散列地址定位在100至110之间，则散列函数可表示为H（key）=\_\_\_\_\_。

22、从理论上讲，散列查找的算法效率为\_\_\_\_\_，但实际应用时，常常因为\_\_\_\_\_使查找速度有所降低。

23、若关键字序列为正序，则直接插入、冒泡、简单选择、归并、快速、堆排序中排序效率最高的是\_\_\_\_\_和\_\_\_\_\_，此时时间复杂度为\_\_\_\_\_\_\_\_\_\_。

24、希尔排序每趟的排序方法是\_\_\_\_\_，平均时间复杂度为\_\_\_\_\_\_\_\_\_；快速排序的平均时间复杂度为\_\_\_\_\_\_\_，最坏时间复杂度为\_\_\_\_\_\_\_\_。

25、排序时所需与关键字的比较次数仅与元素个数有关而与关键字的初始序列无关的排序方法是\_\_\_\_\_，若关键字个数为n个，则第一趟的比较次数为\_\_\_\_\_。

26、写出三种不稳定的排序方法的名称\_\_\_\_\_、\_\_\_\_\_和\_\_\_\_\_。

27、n个元素进行起泡排序时，最好情况下只需进行\_\_\_\_\_次比较和\_\_\_\_次交换，最坏情况需进行\_\_\_\_\_次比较和\_\_\_\_\_次交换。

三、综合题

1、已知线性链表中结点的数据域、指针域依次为data、next，写出在表中删除\*p结点的语句（设该结点存在后继）及\*p结点前插入\*s结点语句。

2、简述顺序表、链表的各自特点。

3、简述链式存储中头结点、头指针；结点、元素的区别。

4、n阶矩阵Aij（1≤i，j≤n）或Aij（0≤i，j≤n-1）的上三角(不含对角线)为常数C，下三角无规律，设计压缩存储方案。

5、画出广义表A=((a，(b)，c)，(d)，(e，f))的链式存储结构。

6、画出由三个结点构造的二叉树的基本形态。

7、已知一棵度为5的树中1度，2度，．．．，5度结点的个数依次为1，2，．．．，5，则叶子个数为多少？

8、证明任意一棵二叉树中叶子数等于两度结点数加1。

9、已知字符Ａ，Ｂ，．．．，G的权值依次为(5，7，8，10，9，20，30)，设计各个字符的Huffman编码，画出Huffman树的存储结构，并计算WPL值。

10、已知二叉树的先序，中序遍历序列分别为EFHBDGAC，FHDBEAGC，画出二叉树的逻辑结构，顺序存储结构和中序线索二叉树的逻辑结构图示。

11、已知某二叉树由八个字符A--H组成，先序，中序，后序遍历序列分别为\_\_\_AC\_\_\_E\_\_\_GH；C\_\_\_DB\_\_\_GHF；\_\_\_DA\_\_\_FEB，画出二叉树的逻辑结构和后序线索二叉树的存储结构。提示：每个空白处至少缺少一个字符。

12、已知森林的逻辑结构如图1所示，画出它的孩子兄弟链表存储结构及其转换的二叉树的逻辑结构。

13、已知无向网如图2所示，画出根据PRIM算法构造的最小生成树步骤图示。

14、画出图3所示无向图的邻接表，以它为基础写出深度、广度优先遍历序列及画出深度、广度优先生成树（从结点A开始）。

15、画出图4所示有向图的邻接矩阵，邻接表，并以该邻接表为基础，写出它的深度、广度优先遍历序列、拓扑排序序列（其中拓扑排序要求标示出入度为0的顶点进栈、出栈的先后次序）。

16、分别画出关键字序列(34，23，56，32，27，30，58，89，56)构造的BST树、AVL树的逻辑结构，并分别计算等概率成功查找条件下的ASL值；设a1、a2、a3是不同的关键字且a1<a2<a3，可组成6中不同的输入序列，问哪几种输入所构造的二叉排序树的高度为3?

17、画出对13个有序关键字进行折半查找的折半查找树。

18、已知散列函数为除余法(余数为7)，关键字序列为(49，10，16，40，79，28，18，83) ，分别采用下列冲突处理方法构造散列表，并计算等概率成功查找条件下的ASL值。

1）链地址法 2）线性探测法（表长为11）；3）二次探测再散列法（表长为25）。

19、已知关键字序列为 (34，17，56，32，45，50，89，19，20，41)，分别写出以下各排序方法每趟排序的结果，并指出它们的稳定性。1）直接插入 2）二路归并 3）快速 4）堆排序 5）冒泡 6）简单选择 7）基数 8）希尔。

20、解释下列概念：

头指针，头结点，数据结构，Huffman树，BST树，AVL树，不稳定排序，散列表。

21、用C语言描述下列存储结构：双向链表，循环队列，二叉链表，线索链表，邻接表。

22、已知关键字序列{52，41，95，21，14，28，82，29}，画出构造的3阶B-树。

23、对称矩阵

1 0 0 2

0 3 0 0

A= 0 0 0 5 行列下标i，j满足：1≤i，j≤4

0 5 0

1）若将矩阵的下三角元素Aij压缩存储到数组S[k]中：求出k与i，j的关系。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 3 | 0 | 0 | 0 | 2 | 0 | 5 | 0 |

下标 1 2 3 4 5 6 7 8 9 10

2）若将视为稀疏矩阵，画出三元组表。

四、算法填空

判断带头结点的单向链表中结点数据值(整型)是否对称的算法如下，对称返回1，否则返回0。PUSH(S，x)，POP(S，x)分别代表为进栈，出栈函数

int xyx(LinkList L)

{LinkList p，q；

int n，n1，i，x；

p=L->next；q=p；n=0；

while (q)

{n++；q=q->next；}

n1=\_\_\_\_\_\_\_\_\_\_；

for(i=1；i<=n1；i++){PUSH(S，p->data)；\_\_\_\_\_\_\_\_\_\_；}

if(\_\_\_\_\_\_\_\_) p=p->next；

while(p){

POP(S，x)；

if（p->data!=x）break；

p=p->next；

}

if(\_\_\_\_\_\_\_\_)return(1)；else return(0)；

}

2、 删除L单链表中最大元素结点的算法。

void delmax(LinkList &L)

{LinkList p，q，r；int m；

r=L；p=L->next；

if (p)

{m=p->data；

\_\_\_\_\_\_\_\_；

p=p->next；

while(p)

{if (\_\_\_\_\_\_\_\_\_){\_\_\_\_\_\_\_\_\_\_\_；m=p->data；}

\_\_\_\_\_\_\_\_\_\_；

p=p->next；

}

q=r->next；

\_\_\_\_\_\_\_\_\_\_\_

free(q)；

}

}

3、将二叉树bt的各个结点左右子树互换算法，ADDQ(Q，bt)，DELQ(Q)，EMPTY(Q)分别代表入队、出队、判队空函数（10分）

void EXCHANGE(BiTree &bt)

{Tree p，q；

if (bt)

{ADDQ(Q，bt)；

while(!EMPTY(Q))

{p=DELQ(Q)；

if(p->lchild)\_\_\_\_\_\_\_\_\_\_；

if(p->rchild)\_\_\_\_\_\_\_\_\_\_；

q=\_\_\_\_\_\_\_\_；

p->rchild=\_\_\_\_\_\_\_\_\_；

\_\_\_\_\_\_\_\_=q；

}

}

}

4、将带头结点的单链表中所有的奇数放在链表的前面，偶数放在后面。

void change(LinkList &L)

{LinkList p，q；

p=L->next；

if(p)

while(\_\_\_\_\_\_\_\_\_\_ )

if (\_\_\_\_\_\_\_\_\_\_){q=p->next；p->next=q->next；

\_\_\_\_\_\_\_\_\_\_；L->next=q；

}

else\_\_\_\_\_\_\_\_；

}

1. 设算术表达式右字符串exp表示，可以包含三种括号：“（）”，“[ ]”，“{ }”它们可以任意次序嵌套使用，实现算法填空，判定表达式中括号是否匹配

int f(char exp)

{int legal=1；

SqStack s；

InitStack(&s)；

for (int I=0；legal&&exp[I]!=’\0’；I++)

{switch(exp[ I])

{case ‘(‘： \_\_\_\_\_\_\_\_\_ ；break；

case ‘[‘： \_\_\_\_\_\_\_\_\_ ；break；

case ‘{‘： \_\_\_\_\_\_\_\_\_；break；

case ‘)’：

case ‘]’：

case ‘}’：if(StackEmpty(s)||exp[I]!= \_\_\_\_\_\_\_\_\_ )legal=0；

}//switch

}//for

return legal&&\_\_\_\_\_\_\_\_\_；

}//f

1. 说明该算法的功能，若输入数据为ABCDEFG回车，画出该算法所得到的存储结构。

void Unknown(LinkList &head){

LinkList head，s；

char ch；

head=(ListNode \*)malloc(sizeof(ListNode))；

head->next=NULL；

printf("输入数据以回车键结束！\n")；

while((ch=getchar())!='\n'){

s=(ListNode \*)malloc(sizeof(ListNode))；

s->data=ch；

s->next=head->next；

head->next=s；

}

}

1. 说明该算法的功能；若输入数据为AB#CD##E##F##回车，画出该算法所得到的存储结构。

void Unknown(BiTree &T)

{char ch；

if （（ch=getchar（））==' #'） T=NULL；

else{T=(BinTree)malloc(sizeof(BinTNode))；

T->data=ch；

Unknown(T->lchild)；

Unknown(T->rchild)；

}

}

8、完善下面实现有向图的拓扑排序算法，并说明该算法中栈的作用。

int TopologiSort(Graph G)//拓扑排序成功返回1，否则返回0

{int i，count=0；

SqStack S；

int ind [MAX]；//存储各顶点的入度

ArcNode \*p；

FindIndegree(G，ind)；//计算各个顶点的入度

InitStack(S)；

for(i=0；i<N；i++)

if(!ind[i]) \_\_\_\_\_\_\_\_\_\_\_\_\_\_；

printf("\nTopological Sort is \n")；

while(\_\_\_\_\_\_\_\_\_\_\_)

{Pop(&S，&i)；

if (count==0) printf("%c"，G[i].data)；

else printf("-->%c"，G.[i].data)；

count++；

for(p=\_\_\_\_\_\_\_\_\_\_\_\_\_；p；p=\_\_\_\_\_\_\_\_\_\_)

if (!--ind[p->adj])//若入度为零

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_；

}

if (\_\_\_\_\_\_\_\_\_\_\_\_\_)return(0)； else return(1)；

}

五、算法设计

1、已知两个带头结点的单向链表Ａ＝（a，b，．．．，c）；Ｂ＝（d，ｅ，．．．，ｆ），设计算法将它们合并为一个单向循环链表Ａ＝（a，b，．．．，c，ｆ，．．．ｅ，ｄ）。

2、实现把N个正整数中的所有奇数排列在偶数之前，分别用顺序结构、单向链表、双向循环链表实现。

3、已知单向链表L中元素为正整数，设计算法删除所有偶数。

4、已知单向循环链表L，结点有三个域：prior，data，next，其中next为后继指针，data为数据域，prior初值为NULL，写出将其改成双向循环链表的算法。

5、已知无头结点单向循环链表表长大于1，指针s指向表中某结点，写出删除\*s前驱的算法。

6、写出把单向链表L逆置算法。1）不允许申请空间；2）不允许破坏原表结构。

7、在非递减有序单向链表L中插入元素e，使序列仍有序的算法InsertList(LinkList L，ElemType e)。

类似题目：某百货公司仓库中的电视机的价格和数量信息，按其价格从低到高存储在一个带头结点的单循环链表中。结点由价格（cost）、数量（num）和链指针（next）三个域组成，现新到m台价格为c的电视机需要入库，试编写修改循环链表中存储的电视机信息的算法，并进行算法效率分析。

8、写出把两个非递减有序链表A，B合并成一个非递增（或非递减）有序链表C的算法MergeList（LinkList A，LinkList B，LinkKList &C）。

9、设计算法DivideList（LinkList &A，LinkList &B，int i）实现将一个单向链表Ａ＝（a，b，．．．，c，d，．．．ｅ，f）分解为两个链表A、B，其中Ａ＝（a，b，．．．，c）；Ｂ＝（f，．．．，e，d）；c为表中第i个元素。

10、把一个正整数单向链表A分解成两个链表A，B，其中A表全为奇数，B表为偶数。

11、约瑟夫（Joseph）问题：编号为1，2，3，．．．，n的n个人按顺时针围成一圈，从第s个人开始按顺时针报数（报数依次为1，2，3，．．．），报数为m的人出队，再从下一个开始继续按上述原则报数，设计算法输出出队序列。

12、判断双向链表中元素是否对称，若对称返回1否则返回0。

13、L为头指针的双向循环链表，每个结点除了有prior，data，next域外，还增加一个访问频度域freq。初始freq值为0，每当对链表进行一次LOCATE（L，x）操作，被访问结点（元素值为x）的访问频度增1，同时调整结点次序，使其按访问频度非递增次序排列以便提高查找效率，设计该算法LOCATE（L，x）。

14、已知带头结点的双向链表表示队列，最后一个结点为队尾，**仅设头指针L**，写出元素e入队的算法。（若表示栈，第一个元素结点为栈顶元素，写出元素e入栈算法。）

15、已知无头结点的单向循环链表表示栈，**仅设尾指针R**，写出元素e入队的算法。

16、分别写出计算链式存储的广义表中原子的个数和深度算法。

17、分别实现以三元组存储结构的稀疏矩阵的转置、相加算法。

18、设计将二叉链表中各个结点的左右孩子互换算法。

19、设计先序/中序遍历二叉树的非递归算法（思考题：求二叉树的叶子个数的非递归算法）。

20、写出求二叉树的高度算法 。

21、先序线索二叉树的先序遍历算法。

22、中序线索二叉树的中序遍历算法。

23、以线索链表为存储结构，写出在后序线索链表中查找\*p的后序前驱算法。

24、以线索链表为存储结构，已知中序线索树中某结点\*p，写出将\*s结点作为\*p的右孩子或左孩子插入的算法。

25、借助队列实现二叉树的层次遍历。

26、计算孩子兄弟链表存储的森林中叶子个数。

27、已知N个结点的某二叉树的先序、中序遍历序列存储于两个数组A，B中，写出建立该二叉树的二叉链表算法。

28、写出建立一个有向图的邻接表的算法。

29、在有向图的邻接表的基础上计算各顶点的度。

30、以邻接表为存储结构实现输出有向图的拓扑排序序列。

31、采用邻接表存储实现无向图的深度优先非递归遍历。

应用或变形题目：判断以邻接表为存储结构的无向图中任意两顶点间是否存在路径，若存在，输出路径上的顶点序列；输出无向图中连通分量个数及各连通分量的顶点序列。

32、采用邻接表存储实现无向图的广度优先遍历。

33、采用邻接矩阵存储实现无向图的最小生成树的PRIM算法。

34、已知BST树root的某结点\*p的左右子树均非空，写出删除该结点的算法。

35、已知N个整型关键字序列存于向量A[1．．N]中，写出建立BST树的非递归算法。

36、已知散列表长度为m，散列函数为H（x），冲突处理方法为拉链法，设计算法输入一组关键字，建立散列表，要求同义词结点按递增排列。

37、已知散列表长度为m，散列函数为H（x），冲突处理方法为线性探测法，设计算法输入一组关键字（个数小于m）建立散列表。

38、写出以链表为存储结构的直接插入排序算法，注意：表中元素个数未知。

39、写出以链表为存储结构的简单选择排序算法。

40、写出以链表为存储结构的冒泡排序算法。

41、利用栈结构写出快速排序的非递归算法，要求打印每趟排序结果。

42、写出折半插入排序算法(即确定元素的插入位置时采用折半的方法)。

43、编写双向起泡排序算法，即扫描方向交替进行。

44、表达式a+b\*（c—d）—e/f可以表示为二叉树结构，运用后序遍历二叉树规则，写出对表达式求值算法

45、设P为某二叉排序树中一结点的指针，且P结点的左右子树均存在，写出删除指针P结点的算法DELTP（p）。