

数据结构本-1

单选题

问题 1:栈的基本运算包括()答案:取栈顶元素

问题 2:串是()。答案:有限个字符的序列

问题 3:广义表(a, d, e, (i, j), k)的表尾是()。答案:(d, e, (i, j), k)

问题 4:链表所具备的特点是()。答案:插入删除元素的操作不需要移动元素结点

问题 5:无向图的邻接矩阵是一个()。答案:对称矩阵

问题 6:数据的存储结构包括数据元素的表示和()。答案:数据元素间的关系的表示

问题 7: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()。答案:29

问题 8:某串的长度小于一个常数,则采用()存储方式最节省空间。答案:顺序

问题 9:有关线性表的正确说法是()。

答案:除了一个和最后一个元素外,其余元素都有一个且仅有一个直接前驱和一个直接后

问题 10:从未排序序列中依次取出元素与已经排好序的序列中的元素作比较。将其放入已排序序列的正确的位置上,此方法称为()。答案:插入排序

问题 11:假定一棵二叉树中,叶子结点数为10,单分支结点数为30,则双分支结点数为()。答案:9

问题 12:判断一个顺序队列sq(最多元素为m)为空的条件是()。答案:sq->front==sq->rear

问题 13:已知一个有序表为{11,22,33,44,55,66,77,88,99},则顺序查找元素55需要比较()次。答案:5

问题 14:链栈和顺序栈相比,有一个比较明显的优点,即()。答案:通常不会出现栈满的情况

问题 15:若Head为一个带头结点的单链表的表头指针,则该表为空表的条件是()。答案:Head->next==NULL

判断题

问题 1:采用顺序查找方法查找长度为n的线性表时,每个元素的平均查找长度为n/2 答案:x

问题 2:哈夫曼树一定是完全二叉树或满二叉树。答案:x

问题 3:线性表用顺序方式存储可以随机访问。答案:v

问题 4:用数组实现顺序栈,栈底可以是数组空间的任何一端答案:v

问题 5:对于一个具有n个结点的单链表,在*p结点后插入一个新结点的时间复杂度是O(n)。答案:x

问题 6:父亲李贵有两个儿子李万胜和李万利,李万胜又有三个儿子李建新、李建中和李建国,这个家庭可以用树结构来描述。答案:v

问题 7:用字符数组存储长度为n的字符串,数组长度至少为n+1。答案:v

问题 8:计算机所处理的数据一般具有某种关系,这是指数据元素与数据元素之间存在的某种关系。答案:v

问题 9:AOV网是一个带权的有向图。答案:x

问题 10:待排序的序列为8,3,4,1,2,5,9,采用直接选择排序算法,当进行了两趟选择后,结果序列为1,2,8,3,4,5,9。

答案:x

问题 11:循环队列队头指针在队尾指针后一个位置,队列是“满”状态。答案:v

问题 12:对稀疏矩阵进行压缩存储,矩阵中每个非零元素对应的三元组包括该元素的行号、列号和元素值三项信息。答案:v

问题 13:二叉排序树中某一结点的左儿子一定小于树中任一个结点的右儿子。答案:x

问题 14:一棵二叉树每一层的结点数都达到最大值,则这个二叉树是完全二叉树。答案:x

问题 15:对于一个无向图,每个顶点的入度等于出度。答案:v

综合题

问题 1:一组记录的关键字序列为(46, 79, 56, 38, 40, 45, 62),利用堆排序(堆顶元素是最小元素)的方法建立的初始堆为()。

选项:40, 38, 45, 46, 56, 79, 62

选项:38, 40, 45, 79, 46, 56,62

选项:38, 79, 45, 46, 40, 62,56

选项:38, 46, 45, 62,79, 40, 56

答案: 38, 40, 45, 79, 46, 56,62

问题 2:以下为求二叉树深度的算法, 完成程序中空格部分。int

```

BTreeDepth(BTreeNode* BT){   if
(BT==NULL)   return
0;   else   {int
dep1=BTreeDepth(BT->left);/* 计
算左子树的深度 */   int
dep2=BTreeDepth(BT->right);/*
计算右子树的深度 */   if
(_____)   return
dep1+1;   else   return
dep2+!;   }}

```

答案: dep1>dep2

问题 3:在下面空格处填写一条语句, 以使下面的串比较算法完整。

```

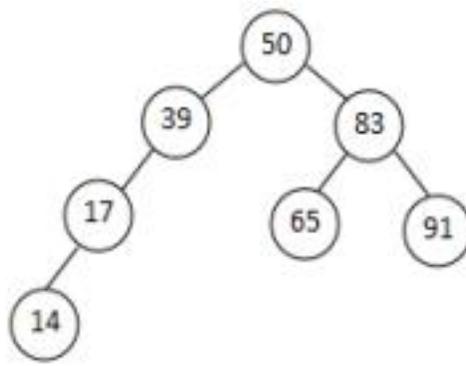
int strcmp(char *s1,char *s2){int i;
for(i=0;s1[i]!='\0'&&
s2[i]!='\0';i++)if(s1[i]>s2[i])
return
1;else   if(s1[i]<s2[i])return
-1;if(s1[i]=='\0'&& s2[i]=='\0')
_____else if(s1[i]!='\0') return
1; else return -1; }

```

答案: return 0; 问题 4:

设有数据集合{50, 39, 17, 83, 91, 14, 65}, 依次取集合中各数据构造一棵二叉排序树, 是如下的()。

答案:



问题 5:在下面空格处填写一条语句, 以使下面的出栈算法完整。

```

ElemType Pop(struct
SeqStack*s,ElemType x){if
(StackEmpty(s))   {printf("栈下溢
错误! \n");
exit(1);}x=s->data[s->top];
_____   return x;}

```

答案: s->top--;

数据结构本-2

单选题

问题 1:向顺序栈中压入新元素时, 应当()。答案: 先移动栈顶指针, 再存入元素

问题 2:两个字符串相等的条件是()。答案: 两个串的长度相等且对应位置的字符相同

问题 3:广义表(f, h, (a, b, d, c), d, e, ((i, j), k))的长度是()。答案: 6

问题 4:在长度为 n (n>1) 的()上, 删除第一个元素, 其算法的时间复杂度为 O(n)。

答案: 只有首结点指针 h 的不带头结点的单向循环链表

问题 5:在一个图 G 中, 所有顶点的度数之和等于所有边数之和的()倍。答案: 2

问题 6:下列的叙述中, 不属于算法特性的是()。答案: 可读性

问题 7:利用 2、4、5、10 这四个值作为叶子结点的权, 生成一棵哈夫曼树, 该树的带权路径长度为()。

答案: 38

问题 8:两个字符串相等的条件是()。答案: 两串的长度相等, 并且对应位置上的字符相同

问题 9:在一个长度为 n 的顺序表中为了删除第 5 个元素, 由第 6 个元素开始从后到前依次移动了 15 个元素。则原顺序表的长度为()。答案: 20

问题 10:已知 10 个数据元素为 (54, 28, 16, 34, 73, 62, 95, 60, 26, 43), 对该数列从小到大排序, 经过一趟冒泡排序后的序列为()。答案: 28, 16, 34, 54, 62, 73, 60, 26, 43, 95

问题 11:一棵二叉树采用链式存储, n 个结点的二叉树共有()个指针域为空。答案: n+1。

问题 12:()的一个重要应用是解决主机和打印机之间速度不匹配的问题。答案: 队列

问题 13:在有序表{1, 3, 8, 13, 33, 42, 46, 63, 76, 78, 86, 97, 100}中, 用折半查找值 86 时, 经()次比较后查找成功。答案: 4

问题 14:下面的应用中, 不符合栈的后进先出特点的是()。答案: 算数运算、逻辑运算和关系运算

问题 15:在双向循环双链表中,删除*p 结点需要 ()。答案:
 $p \rightarrow \text{prior} \rightarrow \text{next} = p \rightarrow \text{next}; p \rightarrow \text{next} \rightarrow \text{prior} = p \rightarrow \text{prior};$

判断题

问题 1:采用顺序查找法对长度为 n (n 为偶数) 的线性表进行查找,采用从前向后的方向查找。在等概率条件下成功查找到前 $n/2$ 个元素的平均查找长度为 $(n+2)/4$ 。答案:
√

问题 2:哈夫曼树叶结点数比非叶结点数多 1。答案:√

问题 3:线性表是一个有限序列,不可以为空。答案:×

问题 4:若让元素 1,2,3 依次进栈,则出栈次序 1,3,2 是不可能出现的情况; 答案:×

问题 5:设有一个单向链表,结点的指针域为 next,头指针为 head, p 指向尾结点,为了使该单向链表改为单向循环链表,可用语句 $p \rightarrow \text{next} = \text{head}$ 。答案:√

问题 6:对于一棵深度为 4 的满三叉树,其结点数为 40。答案:√

问题 7:用字符数组存储长度为 n 的字符串,数组长度至少为 $n+1$ 。答案:√

问题 8:链接存储表示中数据元素之间的逻辑关系是由指针表示的。答案:√

问题 9:使用邻接矩阵存储图的时候,占用空间大小与图的结点个数没有关系。答案:×

问题 10:序列 3,1,7,18,6,9,13,12 经一趟归并排序的结果为 1,3,7,18,6,9,13,12。答案:×

问题 11:队列的特性是先进后出。答案:×

问题 12:对稀疏矩阵进行压缩存储,矩阵中每个非零元素对应的三元组包括该元素的行号、列号和元素值三项

答案:√

问题 13:在有序表 $A[1...18]$ 中,采用二分查找算法查找元素值等于 $A[17]$ 的元素,所比较过的元素的下标依次是 9、14、16、17。答案:√

问题 14:森林是 m ($m \geq 0$) 棵互不相交的树的集合。答案:√

问题 15:图的连通分量是无向图的极大连通子图。答案:√

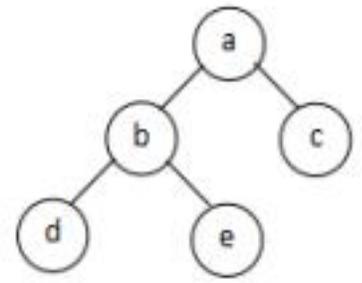
综合题

问题 1:以下是冒泡排序算法对存放在 $a[1], a[2], \dots, a[n]$ 中序列按关键字 key 由小到大排序,完成程序中空格部分。

```
void bsort (NODE a [], int n){ int i,j,flag; NODE temp; for (j=1;j<=n-1;j++) { flag=0; for (i=1;i<=n-j;i++) if (_____) { flag=1; temp=a[i];a[i]=a[i+1]; a[i+1]=temp; }if (flag==0) break; }}
```

答案: $a[i].\text{key} > a[i+1].\text{key}$

问题 2:设某二叉树先序遍历为 abdec, 中序遍历为 dbaec。该二叉树的图形是 ()。



答案:

问题 3:以下程序段的结果是: c 的值为 () `char a[]="abcdefghj"; int *p=a,c=0;While (*p++) c++;`答案: 9

问题 4:二叉排序树结点类型定义如下: `typedef struct Bnode{ int key;struct Bnode *left;struct Bnode *right;} Bnode;` 以下为二叉排序树的查找算法,完成程序中空格部分。

```
Bnode *BSearch(Bnode *bt,int k){ Bnode *p; if (bt==NULL) return (bt); p=bt;while (_____) {if (k<p->key) p=p->left; else p=p->right; if (p==NULL)break; }return (p);}
```

答案: $p \rightarrow \text{key} != k$

问题 5:在下面空格处填写一条语句,以使下面的进栈算法完整。

```
void Push(struct SeqStack*s,ElemType x){if (s->top==MaxSize-1) {printf("栈满溢出错误! \n"); exit(1);}_____s->data[s->top]=x;}
```

答案: $s \rightarrow \text{top}++;$

数据结构本-3

单选题

问题 1:当利用大小为 N 的数组顺序存储一个栈时,假定用 $\text{top} == -1$

表示栈空，则入栈应该执行（）语句修改 top 指针。答案: top++

问题 2:空串与空格串（）。答案:不相同

问题 3:设有一个广义表 A(a)，其表尾为（）。答案:()

问题 4:如果以链表作为栈的存储结构，则退栈操作时（）。答案:必须判断栈是否空

问题 5:对于一个具有 n 个顶点和 e 条边的无向图，若采用邻接表表示，则所有顶点邻接表中的结点总数为（）。

问题 6:线性结构中数据元素之间的关系是（）答案:一对一

问题 7:已知某二叉树的后续遍历序列是 dabec，中序遍历是 debac，则它的先序遍历序列是（）答案:cedba

问题 8:如果进行串的比较，下列哪个串最大？（）答案:“BEIJING”

问题 9:有关线性表的正确说法是（）。

答案:除了一个和最后一个元素外，其余元素都有一个且仅有一个直接前驱和一个直接后继

问题 10:就排序算法所用的辅助空间而言，堆排序、快速排序、归并排序的关系是（）。

答案:堆排序<快速排序<归并排序

问题 11:树中所有结点的度等于所有结点数加（）。答案:-1

问题 12:关于栈和队列的说法中，错误的是（）。答案:栈是先进先出，队列是后进先出

问题 13:采用折半查找方法查找长度为 n 的线性表时，其算法的时间复杂度为（）。答案:O(log2n)

问题 14:判断向上增长型的顺序栈空的条件是（）。答案:top=-1

问题 15:设头指针为 head 的非空的单向链表，指针 p 指向尾结点，则通过以下操作（）可使其成为单向循环链表。

答案:p->next=head;

判断题

问题 1:采用分块查找时，数据的组织方式为把数据分成若干块，每块内数据有序，每块内最大(或最小)的数据组成索引表。答案:x

问题 2:二叉树的遍历就是按照一定次序访问树中所有结点，并且每个结点的值仅被访问一次的过程。答案:v

问题 3:线性表用顺序方式存储可以随机访问。答案:v

问题 4:递归的算法简单、易懂、容易编写，而且执行效率也高。答案:x

问题 5:设有一个单向循环链表，结点的指针域为 next，头指针为 head，指针 p 指向表中某结点，若逻辑表达式 p->next==head;的结果为真，则 p 所指结点为尾结点。答案:v

问题 6:对于一棵深度为 h，度为 3 的树最多有 (3h-1)/2 个结点。答案:x

问题 7:字符串属于线性的数据结构答案:v

问题 8:数据项是数据的最小单位。答案:v

问题 9:AOV 网是一个带权的有向图。答案:x

问题 10:在归并排序中，在第 3 趟归并中，是把长度为 4 的有序表归并为长度为 8 的有序表。答案:v

问题 11:在一个顺序存储的循环队列中，队头指针指向队头元素的后一个位置。答案:x

问题 12:使用三元组表示稀疏矩阵中的非零元素能节省存储空间。答案:v

问题 13:在一个查找表中，能够唯一地确定一个记录的关键字称为主关键字。答案:v

问题 14:深度为 k 的完全二叉树至少有 2k-1 个结点。答案:x

问题 15:一个有向图的邻接表和逆邻接表中的节点个数一定相等。答案:v

综合题

问题 1:以下程序是快速排序的算法，完成程序中空格部分。设待排序的记录序列存放在 a[start],...a[end]中，按记录的关键字进行快速排序，先进行一次划分，再分别进行递归调用。 void quicksort(NODE a[],int start,int

```
end){ inti,j;NODE mid;if
(start>=end )
return;i=start;j=end;mid=a[i];while
(i<j){ while(i<j && a[j].key>mid.key)
j--;if(i<j) {a[i]=a[j];i++;}while(i<j &&
a[i].key<=mid.key) i++;if(i<j)
{a[j]=a[i]; j--;} } a[i]=mid;
```

quicksort(a,i+1,end);}答案:
quicksort(a,start,i-1);

问题 2: 以下为求二叉树深度的算法, 完成程序中空格部分。int
BTreeDepth(BTreeNode* BT){ if
(BT==NULL) return
0; else {int
dep1=BTreeDepth(BT->left);/* 计
算左子树的深度 */ int
dep2=BTreeDepth(BT->right);/*
计算右子树的深度 */ if
(_____) return
dep1+1; else return
dep2+1; } }答案: dep1>dep2

问题 3: 在下面空格处填写一条语句, 以使下面的串比较算法完整。
int strcmp(char *s1,char *s2){int i;
for(i=0;s1[i]!='\0'&&
s2[i]!='\0';i++)if(s1[i]>s2[i])
return
1;else if(s1[i]<s2[i])return
-1;if(s1[i]=='\0'&& s2[i]=='\0')
_____else if(s1[i]!='\0') return
1; else return -1; }答案: return 0;

问题 4: 设查找表为
(16,15,20,53,64,7), 用冒泡法对该表进行排序, 在排序后的有序表的基础上进行折半查找, 在等概率条件下, 成功查找的平均查找长度为 ()。答案: 14/6

问题 5: 设有一个头指针为 head 的不带头结点单向链表中(结点类型为 NODE), p 为指向该链表中某

个结点的指针。以下程序段为插入一个指针为 s 的结点, 使它成为 p 结点的直接前驱, 请把合适选项填写到空行处。NODE *q;q=head;
while(q->next!=p)
q=q->next;s->next=p;_____ ;答
案: q->next=s

数据结构本-4

单选题

问题 1: 以下数据结构中 () 是线性结构。答案: 栈

问题 2: 下面关于串的叙述中, 正确的是 ()。答案: 模式匹配是串的一种重要运算

问题 3: 稀疏矩阵采用压缩存储的目的主要是 ()。答案: 减少不必要的存储空间的开销

问题 4: 链表不具有的特点是 ()。答案: 可随机访问任一元素

问题 5: 一个具有 n 个顶点的有向完全图包含 () 条边。答案: $n(n-1)$

问题 6: 下面程序段的时间复杂度是 ()。for(i=1;i<=n;i++)
for(j=1;j<=n;j++){ c[i][j]=0;
for(k=1;k<=n;k++)c[i][j]=c[i][j]+a[i][k]
*b[k][j]; }答案: $O(n^3)$

问题 7: 已知某二叉树的后续遍历序列是 dabec, 中序遍历是 debac, 则它的先序遍历序列是 ()。答案: cedba

问题 8: 以下四个串中最小的是 ()。答案: "ABADF"

问题 9: () 不属于线性表的基本操作。答案: 求子表

问题 10: 对具有 n 个元素的任意序列采用插入排序法进行排序, 排序趟数为 ()。答案: n-1

问题 11: 假定一棵二叉树中, 叶子结点数为 10, 单分支结点数为 30, 则双分支结点数为 ()。答案: 9

问题 12: () 的一个重要应用是解决主机和打印机之间速度不匹配的问题。答案: 队列

问题 13: 有一个长度为 10 的有序表, 按折半查找对该表进行查找, 在等概率情况下查找成功的平均比较次数为 ()。答案: 29/10

问题 14: 下面关于栈的基本运算算法中, 复杂度最高的是 ()。答案: 链栈清空运算

问题 15: 非空的单向循环链表的尾结点满足 () (设头指针为 head, 指针 p 指向尾结点) 答案: p->next==head

判断题

问题 1: 采用顺序查找法对长度为 n (n 为偶数) 的线性表进行查找, 采用从前向后的方向查找。在等概率条件下成功查找到前 n/2 个元素的平均查找长度为 $(n+2)/4$ 。答案: v

问题 2: 如果一个叶子结点是某二叉树中序遍历序列的最后一个结点, 那么它也是该二叉树的先序遍历序列的最后一个结点。答案: v

问题 3: 设有一个长度为 40 的顺序表, 要删除第 8 个元素需移动元素的个数为 33。答案: x

问题 4:若让元素 1,2,3 依次进栈, 则出栈次序 1,3,2 是不可能出现的情况。答案: x

问题 5:在双向循环链表上, 删除最后一个结点, 其算法的时间复杂度为 $O(1)$ 。答案: \checkmark

问题 6:完全二叉树中没有度为 1 的结点。答案: x

问题 7:字符串 a1= "heijing", a2 = "hen", a3= "heifang", a4="heni", 其中最小的是 a2。答案: x

问题 8:数据的物理结构是指数据在计算机内的实际存储形式。答案: \checkmark

问题 9:若一个强连通图有 n 个顶点, 则该强连通图中至少含有 n 条有向边。答案: \checkmark

问题 10:在归并排序中, 在第 3 趟归并中, 是把长度为 4 的有序表归并为长度为 8 的有序表。答案: \checkmark

问题 11:在队列的顺序存储结构中, 当插入一个新的队列元素时, 尾指针后移, 当删除一个元素队列时, 头指针后移。答案: \checkmark

问题 12:一个广义表的表头总是一个广义表。答案: x

问题 13:顺序查找是一种最简单的查找方法。答案: \checkmark

问题 14:具有 n 个结点的二叉树, 采用二叉链表存储, 共有 n-1 个空链域。答案: x

问题 15:存储图的邻接矩阵中, 邻接矩阵的大小不但与图的顶点个

数有关, 而且与图的边数也有关。

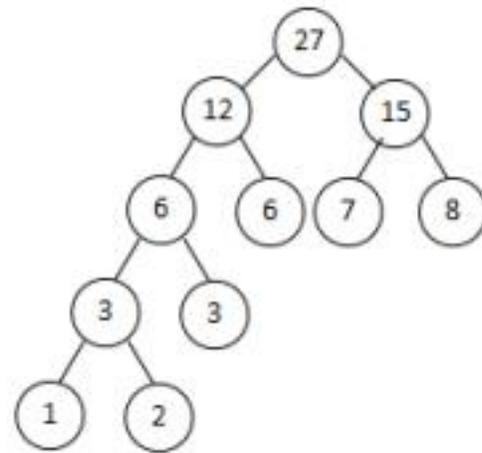
答案: x

综合题

问题 1:以下是直接插入排序算法对存放在 a[0],a[1],.....,a[n-1]中, 长度为 n 的记录序列按关键字 key 由小到大排序, 完成程序中空格部分。void disort (NODE a[], int n){ inti,j;NODE temp;for (i=1;i<n;i++){ temp=a[i]; j=i-1; while (j>=0&&temp.key<a[j].key) { a[j+1]=a[j]; _____; }a[j+1]=temp; }答案: j--

问题 2:以 1, 2, 3, 6, 7, 8 作为叶结点的权, 构造一棵哈夫曼树是如下哪个图? ()

答案:



问题 3:在下面空格处填写一条语句, 以使下面的顺序队列入队算法完整。void InQueue(struct SeqQueue *sq, int x){if (sq->rear==MaxSize) {printf("队列已满! \n");

exit(1);}_____sq->rear++; }

答案: sq->data[sq->rear]=x;

问题 4:设查找表为: 用折半查找在该查找表成功查找到元素 55 需要经过 () 次比较。

序号	1	2	3	4	5	6	7
序列	4	12	18	19	37	55	65

答案: 2

问题 5:设有一个头指针为 head 的单向链表中 (结点类型为 NODE), p 为指向该链表中某个结点的指针。以下程序段为插入一个指针为 s 的结点, 使它成为 p 结点的直接前驱, 请选择其中空格的选项。

NODE *q;q=head; while(q->next!=p) _____;s->next=p;q->next=s;答案: q=q->next

数据结构本-5

单选题

问题 1:下面的操作不是栈基本运算的是 ()。答案: 排序操作

问题 2:空串与空格串 ()。答案: 不相同

问题 3:下列广义表中的线性表是 ()。答案: E(a,b)

问题 4:设有两个长度为 n 的单向链表, 结点类型相同, 分别是循环链表和非循环链表, 则 ()。

答案: 对于两个链表来说, 删除最后一个结点的操作, 其时间复杂度都是 $O(n)$

问题 5: 在有向图的邻接表中, 每个顶点邻接表链接着该顶点所有 () 邻接点。答案: 出边

问题 6: 数据结构中, 与所使用的计算机无关的是数据的 ()。答案: 逻辑结构

问题 7: 设 a, b 为一棵二叉树的两个结点, 在后续遍历中, a 在 b 前的条件是 ()。答案: a 在 b 下方

问题 8: 串函数 `Strcat(a,b)` 的功能是进行串 ()。答案: 连接

问题 9: () 不属于线性表的基本操作。答案: 求子表

问题 10: 就排序算法所用的辅助空间而言, 堆排序、快速排序、归并排序的关系是 ()。

答案: 堆排序 < 快速排序 < 归并排序

问题 11: 假定一棵二叉树中, 叶子结点数为 10, 单分支结点数为 30, 则双分支结点数为 ()。答案: 9

问题 12: 当利用大小为 100 的数组顺序存储一参考 1: 简单的考察对基础常识的掌握。

问题 13: 有一个长度为 10 的有序表, 按折半查找对该表进行查找, 在等概率情况下查找成功的平均比较次数为 ()。答案: 29/10

问题 14: 向顺序栈中压入新元素时, 应当 ()。答案: 先移动栈顶指针, 再存入元素

问题 15: 非空的单向循环链表的尾结点满足 () (设头指针为 $head$, 指针 p 指向尾结点)。

答案: $p \rightarrow next == head$

判断题

问题 1: 在各种查找方法中, 平均查找长度与结点个数 n 无关的查找方法是哈希表查找。答案: \checkmark

问题 2: 哈夫曼树一定是完全二叉树或满二叉树。答案: \times

问题 3: 线性表的顺序存储是利用数组来实现的。答案: \checkmark

问题 4: 用数组实现顺序栈, 栈底可以是数组空间的任何一端。答案: \checkmark

问题 5: 设有一个不带头结点的单向循环链表, 结点的指针域为 $next$, 指针 p 指向尾结点, 现要使 p 指向第一个结点, 可用语句 $p = p \rightarrow next$;。答案: \checkmark

问题 6: 树是一种重要的非线性数据结构。答案: \checkmark

问题 7: 两个字符串比较时, 较长的串比较短的串大。答案: \times

问题 8: 数据的逻辑结构是与存储该结构的计算机相关的。答案: \times

问题 9: 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 $BCDA$ 。

答案: \times

问题 10: 序列 $15, 13, 16, 14, 19, 17$, 采用冒泡排序算法(升序), 经一趟冒

泡后, 结果序列是 $13, 15, 14, 16, 17, 19$ 。

答案: \checkmark

问题 11: 在队列的顺序存储结构中, 当插入一个新的队列元素时, 尾指针后移, 当删除一个元素队列时, 头指针后移。答案: \checkmark

问题 12: 设广义表 $L = ((), ())$, 则其表头是 ()。答案: \times

问题 13: 分块查找分为两个步骤: 第一步是要对索引表进行查找; 第二步是在块中查找。这两步查找都可以采用折半查找或者顺序查找方法。答案: \times

问题 14: 树是一种线性结构。答案: \times

问题 15: 由一个具有 n 个顶点的连通图生成的最小生成树中, 具有 $n-1$ 条边。答案: \checkmark

综合题

问题 1: 一组记录的关键字序列为 (36, 69, 46, 28, 30, 84), 对该序列进行直接选择排序(每次选择最小关键字), 第二趟排序后的结果序列为 ()。答案: 28, 30, 46, 36, 69, 84

问题 2: 已知某带权图的邻接矩阵如下所示: 从顶点 1 出发的广度优先搜索序列为 ()。

0	6	1	5	∞	∞
6	0	5	∞	3	∞
1	5	0	5	6	4
5	∞	5	0	∞	2
∞	3	6	∞	0	6
∞	∞	4	2	6	0

答案:

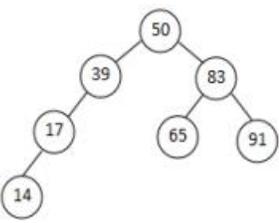
1, 2, 3, 4, 5, 6

问题 3:写出下列程序段执行后的结果 SeqQueue

```
Q; InitQueue(Q);int i,
a[4]={5,8,12,15};for(i=0;i<4;i++)
InQueue(Q,a[i]);InQueue(Q,OutQueue(Q));InQueue(Q,30);InQueue(Q,OutQueue(Q)+10);while(!QueueEmpty(Q)) printf("%d",OutQueue(Q));
```

答案: 121553018

问题 4:设有数据集{50, 39, 17, 83, 91, 14, 65}, 依次取集合中各数据构造一棵二叉排序树, 是如下的 ()。



答案:

问题 5:在下面空格处填写一条语句, 以使下面的进栈算法完整。

```
void Push(struct
SeqStack*s,ElemType x){if
(s->top==MaxSize-1) {printf("栈
满溢出错误! \n");
exit(1);}_____s->data[s->top]=x;}
```

答案: s->top++;

数据结构本-6

单选题

问题 1:下面关于栈的基本运算算法中, 复杂度最高的是 ()。答案: 链栈清空运算

问题 2:下列是"abcd321ABCD"的子串的选项是 ()。答案:"21ABC"

问题 3:设有一个广义表 A (a), 其表尾为 ()。答案: ()

问题 4:在非空双向循环链表的*p 结点之前插入*q 结点的操作是 ()。

答案:

```
q->next=p;q->prior=p->prior;p->prior->next=q;p->prior=q;
```

问题 5:邻接表是图的一种 ()。答案: 链式存储结构

问题 6:数据的存储结构包括数据元素的表示和 ()。答案: 数据元素间的关系的表示

问题 7:设 a, b 为一棵二叉树的两个结点, 在后续遍历中, a 在 b 前的条件是 ()。答案: a 在 b 下方

问题 8:某串的长度小于一个常数, 则采用 () 存储方式最节省空间。答案: 顺序

问题 9:设有一个长度为 n 的顺序表, 要删除第 i 个元素, 则需移动元素的个数为 ()。答案: n-i

问题 10:依次将每两个相邻的有序表合并成一个有序表的排序方法称为 ()。答案: 归并排序

问题 11:在一棵度具有 5 层的满二叉树中结点总数为 ()。答案: 31

问题 12:队列是一种操作受限的线性表, 其限制是 ()。

答案: 仅允许在表的一端进行插入, 而在另一端进行删除操作

问题 13:对线性表进行二分查找时, 要求线性表必须 ()。答案: 以顺序存储方式, 且数据元素有序

问题 14:栈的操作特性决定了它是一种 () 的线性表。答案: 先进后出

问题 15:在一个带头结点的单向链表中, 若要在指针 q 所指结点后插入 p 指针所指结点, 则执行 ()。

答案: p->next=q->next; q->next=p;

判断题

问题 1:二叉树中任一结点的值均大于其左孩子的值, 小于其右孩子的值, 则它是一棵二叉排序树。答案: x

问题 2:哈夫曼树只存在着双支结点, 不存在单支结点。答案: v

问题 3:在线性表的顺序存储中, 元素之间的逻辑关系是通过物理存储位置决定的; 在线性表的链式存储中, 元素之间的逻辑关系是通过链域的指针值决定的。答案: v

问题 4:若让元素 a,b,c 依次进栈, 则出栈次序 c,a,b 是不可能出现的情况。答案: v

问题 5:在单链表中, 要删除某一指定的结点, 必须找到该结点的直接前驱结点。答案: v

问题 6:对于一棵深度为 h, 度为 3 的树最多有 $(3h-1)/2$ 个结点。答案: x

问题 7:一个空格的串的长度是 0。答案: x

问题 8:数据项是数据的最小单位。答案: v

问题 9:图的深度优先搜索序列和广度优先搜索序列不是惟一的。答案: v

问题 10:冒泡排序是一种比较简单的交换排序方法。答案: v

问题 11:循环队列是将队列想象成一个首尾相接的圆环。答案: v

问题 12:数组通常具有的操作是顺序存取。答案: x

问题 13:按照一定规则, 在二叉排序树上插入、删除结点, 仍能保持二叉排序树的性质。答案: v

问题 14:若树中各结点的子树是按照一定的次序从左向右安排的, 则称之为有序树。答案: v

问题 15:图的最小生成树只有一棵。答案: x

综合题

问题 1:一组记录的关键字序列为 (46, 79, 56, 38, 40, 45, 62), 利用堆排序 (堆顶元素是最小元素) 的方法建立的初始堆为 ()。答案: 38, 40, 45, 79, 46, 56, 62

问题 2:以下为求二叉树深度的算法, 完成程序中空格部分。int BTreeDepth(BTreeNode* BT){ if (BT==NULL) return 0; else {int

```
dep1=BTreeDepth(BT->left);/* 计算左子树的深度 */ int dep2=BTreeDepth(BT->right); /* 计算右子树的深度 */ if (_____) return dep1+1; else return dep2+1; }答案: dep1>dep2
```

问题 3:在下面空格处填写一条语句, 以使下面的循环队列出队算法完整。ElemType OutQueue(struct SeqQueue *sq){if (sq->rear==sq->front) {printf("队列已空, 不能进行出队操作!\n"); exit(1);}_____sq->front=(sq->front+1)%MaxSize; return x;}答案: x=sq->data[sq->front];

问题 4:以下函数在 a[0]到 a[n-1] 中, 用折半查找算法查找关键字等于 k 的记录, 查找成功返回该记录的下标, 失败时返回-1, 完成程序中的空格选项。typedef struct{intkey;.....}NODE;int Binary_Search(NODE a[], int n, int k){int low, mid, high; low=0; high=n-1; while (_____) {mid=(low+high)/2}if (a[mid].key==k) return mid; else if (a[mid].key<k)low=mid+1; else high=mid-1; } return -1;}答案: low<=high

问题 5:在下面空格处填写一条语句, 以使下面的出栈算法完整。ElemType Pop(struct SeqStack*s, ElemType x){if (StackEmpty(s)) {printf("栈下溢错误!\n"); exit(1);}x=s->data[s->top]; _____ return x;}

答案: s->top--;

数据结构本-7

单选题

问题 1:当利用大小为 N 的数组顺序存储一个栈时, 假定用 top==N 表示栈空, 则入栈应该执行 () 语句修改 top 指针。答案: top--

问题 2:串的长度是指 ()。答案: 串中所含字符的个数

问题 3:广义表 (a, a, b, d, e, (i, j), k) 的表头是 ()。答案: a

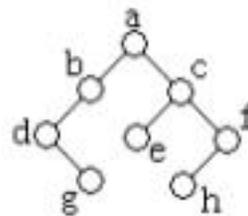
问题 4:链表不具有的特点是 ()。答案: 可随机访问任一元素

问题 5:对于一个具有 n 个顶点和 e 条边的无向图, 若采用邻接表表示, 则所有顶点邻接表中的结点总数为 ()。

答案: 2e

问题 6:数据结构中, 与所使用的计算机无关的是数据的 ()。答案: 逻辑结构

问题 7:如图所示二叉树的中序遍历序列是 ()。



答案:

dgbaechf

问题 8:以下四个串中最小的是 ()。答案:"ABADF"

问题 9:在一个长度为 n 的顺序表中为了删除第 5 个元素,由第 6 个元素开始从后到前依次移动了 15 个元素。则原顺序表的长度为 ()。答案:20

问题 10:序列状态为 () 时,快速排序达到最好的时间复杂度。答案:序列无序

问题 11:任何一棵二叉树的叶结点在先序、中序和后序遍历序列中的 ()。答案:相对次序不改变

问题 12:假设链队的队首和队尾指针是 F 和 R,那么队空的条件是()。答案: R=NULL

问题 13:对二叉排序树进行 () 遍历,可以使遍历所得到的序列是有序序列。答案:中序

问题 14:一般情况下,将递归算法转换成等价的非递归算法应该设置 ()。答案: 栈

问题 15: () 有两个指针域,分别指向直接前驱和直接后继,可以实现从前向后和从后向前查找。

答案: 双向循环链表

判断题

问题 1:二叉树的根结点值大于其左子树结点的值,小于右子树结点的值,则它是一棵二叉排序树。答案: x

问题 2:哈夫曼树只存在着双支结点,不存在单支结点。答案: v

问题 3:向一个长度为 n 的顺序表中的第 i 个元素 (1 ≤ i ≤ n) 之前插入

一个元素时,需向后移动 n-i 个元素。

答案: x。

问题 4:栈是限定在表的两端进行插入和删除操作的线性表,又称为先进先出表。答案: x

问题 5:各种链表只需定义有两个域的结点。答案: x

问题 6:树的所有结点有且只有一个前驱结点。答案: x

问题 7:两个字符串比较时,较长的串比较短的串大答案: x

问题 8:健壮算法不会因非法的输入数据而出现莫名其妙的状态。答案: v。

问题 9:图的广度优先搜索序列是惟一的。答案: x

问题 10:冒泡排序是一种比较简单的插入排序方法。答案: x

问题 11:循环队列是将队列想象成一个首尾相接的圆环。答案: v

问题 12:一个广义表的表头总是一个广义表。答案: x

问题 13:散列技术中的冲突指的是两个元素具有相同的序号。答案: x

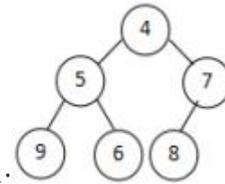
问题 14:具有 10 个结点的完全二叉树有 5 个叶子。答案: v

问题 15:设有 6 个结点的无向图,该图至少应有 6 条边才能确保是一个连通图。答案: x

综合题

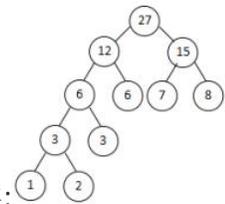
问题 1:一组记录的关键字序列为 (6, 9, 7, 4, 5, 8), 利用堆排

序(堆顶元素是最小元素)的方法建立初始堆是如下哪个图? ()



答案:

问题 2:以 1, 2, 3, 6, 7, 8 作为叶结点的权,构造一棵哈夫曼树是如下哪个图? ()



答案:

```
假设队列顺序存储结构为: struct SeqQueue { ElemType data[MaxSize]; int front,rear;};struct SeqQueue *sq;请补充下面出队算法(不考虑空间循环使用)。void OutQueue(struct SeqQueue * sq,ElemType x){ if( 1 ){ printf("队列已空,不能出队\n"); exit(1); } 2 return sq->data[sq->front-1];}
```

问题 3:其中, 1 和 2 处应该补充的代码是 () 答案:

```
sq->front==sq->rear , sq->front++
```

```
问题 4:二叉排序树结点类型定义如下: typedef struct Bnode{ int key;struct Bnode *left;struct Bnode *right;} Bnode;以下为二叉排序树的查找算法,完成程序中空格部分。Bnode *BSearch(Bnode *bt,int k){ Bnode *p; if (bt==NULL) return
```

```
(bt); p=bt;while
( ) {if
(k<p->key) p=p->left; else
p=p->right; if
(p==NULL)break; }return (p);}
```

答案: $p \rightarrow key \neq k$

问题 5:在下面空格处填写一条语句,以使下面的出栈算法完整。

```
ElemType Pop(struct
SeqStack*s,ElemType x){if
(StackEmpty(s)) {printf(“栈下溢
错误! \n”);
exit(1);}x=s->data[s->top];
return x;}
```

答案: $s \rightarrow top--;$

数据结构本-8

单选题

问题 1:判断向上增长型的顺序栈空的条件是 ()。答案: $top = -1$

问题 2:下列说法不正确的是 ()。答案: 串不是线性结构

问题 3:广义表 $(f, h, (a, b, d, c), d, e, ((i, j), k))$ 的长度是 ()。答案: 6

问题 4:链表所具备的特点是 ()。答案: 插入删除元素的操作不需要移动元素结点

问题 5:图的深度优先遍历算法类似于二叉树的 () 遍历。答案: 先序

问题 6:数据的存储结构包括数据元素的表示和 ()。答案: 数据元素间的关系的表示

问题 7:在一非空二叉树的中序遍历序列中,根结点的右边 ()。答案: 只有右子树上的所有结点

问题 8:某串的长度小于一个常数,则采用 () 存储方式最节省空间。

答案: 顺序

问题 9:在一个长度为 n 的顺序表中为了删除第 5 个元素,由第 6 个元素开始从后到前依次移动了 15 个元素。则原顺序表的长度为 ()。答案: 20

问题 10:就排序算法所用的辅助空间而言,堆排序、快速排序、归并排序的关系是 ()。

答案: 堆排序 < 快速排序 < 归并排序

问题 11:在二叉树的第 4 层最多含有 () 个结点答案: 8

问题 12:关于栈和队列的说法中,错误的是 ()。答案: 栈是先进先出,队列是后进先出

问题 13:有一个长度为 12 的有序表,按折半查找对该表进行查找,在等概率情况下查找成功的平均比较次数为 ()。答案: $37/12$

问题 14:表达式 $a*(b+c)-d$ 的后缀表达式是 ()。答案: $abc+*d-$

问题 15:带头结点的双向循环链表 L 为空表的条件是 ()。答案: $L \rightarrow next == L$

判断题

问题 1:采用分块查找时,数据的组织方式是把数据分成若干块,块内数据不必有序,但块间必需有序,每块内最大(或最小)的数据组成索引表。答案: \checkmark

问题 2:已知一棵树的先序序列和后序序列,一定能构造出该树。答案: \times

问题 3:长度为 0 的线性表称为空表。答案: \checkmark

问题 4:递归算法可读性差,但是效率高答案: \times

问题 5:各种链表只需定义有两个域的结点。答案: \times

问题 6:树最适合表示元素之间具有层次关系的数据。答案: \checkmark

问题 7:串中的元素只可能是字母。答案: \times

问题 8:数据结构中,元素之间存在多对多的关系称为树状结构。答案: \times

问题 9:用邻接矩阵存储图的时候,占用空间大小不但与图的结点数有关还与图的边数有关。答案: \times

问题 10:序列 15,13,16,14,19,17,采用冒泡排序算法(升序),经一趟冒泡后,结果序列是 13,15,14,16,17,19。

答案: \checkmark

问题 11:队列的特性是先进后出。答案: \times

问题 12:递归算法执行时,每次递归可将原问题的规模缩小。答案: \checkmark

问题 13:顺序查找是一种最简单的查找方法。答案: \checkmark

问题 14:树是一种重要的非线性数据结构。答案: \checkmark

问题 15: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。

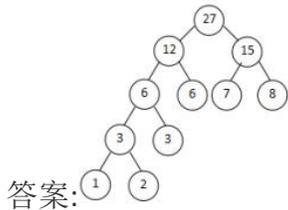
答案:√

综合题

问题 1:设关键字序列为: (36, 69, 46, 28, 30, 74), 将此序列用快速排序的方法, 以第一个记录为基准得到的一趟划分的结果为 ()。

答案: 30, 28, 36, 46, 69, 74

问题 2:以 1, 2, 3, 6, 7, 8 作为叶结点的权, 构造一棵哈夫曼树是如下哪个图? ()



答案:

```

head。以下程序的功能是输出链表中各结点中的数据域 data, 完成程序中空格部分。#define NULL 0void main( ){NODE *head,*p;p=head; /*p 为工作指针*/ do { printf("%d\n", p->data);p=p->next;} while(______);}

```

答案: p!=NULL

问题 3:在下面空格处填写一条语句, 以使下面的串连接算法完整。

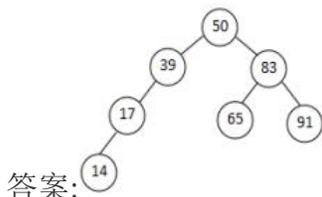
```

char *strcat(char *s1,char *s2){char *p=s1; while(*p!='\0') p++; while(*s2!='\0') { *p=*s2;p++;_____ } *p='\0'; return s1;}

```

答案: s2++;

问题 4:设有数据集合{50, 39, 17, 83, 91, 14, 65}, 依次取集合中各数据构造一棵二叉排序树, 是如下的 ()。



答案:

问题 5:设线性表以不带头结点的单向链表存储, 链表头指针为