**数据结构试卷（九）**

**一、选择题(30分)**

1．下列程序段的时间复杂度为（ ）。

for(i=0； i<m； i++) for(j=0； j<t； j++) c[i][j]=0；

for(i=0； i<m； i++) for(j=0； j<t； j++) for(k=0； k<n； k++) c[i][j]=c[i][j]+a[i][k]\*b[k][j]；

(A) O(m\*n\*t) (B) O(m+n+t) (C) O(m+n\*t) (D) O(m\*t+n)

2．设顺序线性表中有n个数据元素，则删除表中第i个元素需要移动（ ）个元素。

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

3．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

4．利用直接插入排序法的思想建立一个有序线性表的时间复杂度为（ ）。

(A) O(n) (B) O(nlog2n) (C) O(n2) (D) O(1og2n)

5．设指针变量p指向双向链表中结点A，指针变量s指向被插入的结点X，则在结点A的后面插入结点X的操作序列为（ ）。

(A) p->right=s； s->left=p； p->right->left=s； s->right=p->right；

(B) s->left=p；s->right=p->right；p->right=s； p->right->left=s；

(C) p->right=s； p->right->left=s； s->left=p； s->right=p->right；

(D) s->left=p；s->right=p->right；p->right->left=s； p->right=s；

6．下列各种排序算法中平均时间复杂度为O(n2)是（ ）。

(A) 快速排序 (B) 堆排序 (C) 归并排序 (D) 冒泡排序

7．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+l -i (D) 不能确定

8．设散列表中有m个存储单元，散列函数H(key)= key % p，则p最好选择（ ）。

(A) 小于等于m的最大奇数 (B) 小于等于m的最大素数

(C) 小于等于m的最大偶数 (D) 小于等于m的最大合数

9．设在一棵度数为3的树中，度数为3的结点数有2个，度数为2的结点数有1个，度数为1的结点数有2个，那么度数为0的结点数有（ ）个。

(A) 4 (B) 5 (C) 6 (D) 7

10.设完全无向图中有n个顶点，则该完全无向图中有（ ）条边。

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

11.设顺序表的长度为n，则顺序查找的平均比较次数为（ ）。

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

12.设有序表中的元素为(13，18，24，35，47，50，62)，则在其中利用二分法查找值为24的元素需要经过（ ）次比较。

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

13.设顺序线性表的长度为30，分成5块，每块6个元素，如果采用分块查找，则其平均查找长度为（ ）。

(A) 6 (B) 11 (C) 5 (D) 6.5

14.设有向无环图G中的有向边集合E={<1，2>，<2，3>，<3，4>，<1，4>}，则下列属于该有向图G的一种拓扑排序序列的是（ ）。

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

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

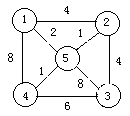
**二、填空题(30分)**

1. 设指针p指向单链表中结点A，指针s指向被插入的结点X，则在结点A的前面插入结点X时的操作序列为：

1) s->next=\_\_\_\_\_\_\_\_\_\_\_；2) p->next=s；3) t=p->data；

4) p->data=\_\_\_\_\_\_\_\_\_\_\_；5) s->data=t；

1. 设某棵完全二叉树中有100个结点，则该二叉树中有\_\_\_\_\_\_\_\_\_\_\_\_\_\_个叶子结点。
2. 设某顺序循环队列中有m个元素，且规定队头指针F指向队头元素的前一个位置，队尾指针R指向队尾元素的当前位置，则该循环队列中最多存储\_\_\_\_\_\_\_队列元素。
3. 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。
4. 在堆排序和快速排序中，如果从平均情况下排序的速度最快的角度来考虑应最好选择\_\_\_\_\_\_\_\_\_排序，如果从节省存储空间的角度来考虑则最好选择\_\_\_\_\_\_\_\_排序。
5. 设一组初始记录关键字序列为(20，12，42，31，18，14，28)，则根据这些记录关键字构造的二叉排序树的平均查找长度是\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
6. 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\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
7. 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\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。



1. 设一组记录关键字序列为(80，70，33，65，24，56，48)，则用筛选法建成的初始堆为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
2. 设无向图G（如右图所示），则其最小生成树上所有边的权值之和为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

**三、判断题(20分)**

* + 1. 有向图的邻接表和逆邻接表中表结点的个数不一定相等。( )
    2. 对链表进行插入和删除操作时不必移动链表中结点。( )
    3. 子串“ABC”在主串“AABCABCD”中的位置为2。( )
    4. 若一个叶子结点是某二叉树的中序遍历序列的最后一个结点，则它必是该二叉树的先序遍历序列中的最后一个结点。( )
    5. 希尔排序算法的时间复杂度为O(n2)。( )
    6. 用邻接矩阵作为图的存储结构时，则其所占用的存储空间与图中顶点数无关而与图中边数有关。( )
    7. 中序遍历一棵二叉排序树可以得到一个有序的序列。( )
    8. 入栈操作和入队列操作在链式存储结构上实现时不需要考虑栈溢出的情况。( )
    9. 顺序表查找指的是在顺序存储结构上进行查找。（ ）
    10. 堆是完全二叉树，完全二叉树不一定是堆。（ ）

**五、算法设计题(20分)**

* + 1. 设计计算二叉树中所有结点值之和的算法。
    2. 设计将所有奇数移到所有偶数之前的算法。
    3. 设计判断单链表中元素是否是递增的算法。

**数据结构试卷（九）参考答案**

**一、选择题**

1．A 2．A 3．A 4．C 5．D

6．D 7．C 8．B 9．C 10．A

11．C 12．C 13．D 14．A 15．A

**二、填空题**

1. p->next，s->data
2. 50
3. m-1
4. 6，8
5. 快速，堆
6. 19/7
7. CBDA
8. 6
9. (24，65，33，80，70，56，48)
10. 8

**三、判断题**

1．错 2．对 3．对 4．对 5．错

6．错 7．对 8．对 9．错 10．对

**四、算法设计题**

1. 设计计算二叉树中所有结点值之和的算法。

void sum(bitree \*bt,int &s)

{

if(bt!=0) {s=s+bt->data; sum(bt->lchild,s); sum(bt->rchild,s);}

}

1. 设计将所有奇数移到所有偶数之前的算法。

void quickpass(int r[], int s, int t)

{

int i=s,j=t,x=r[s];

while(i<j)

{

while (i<j && r[j]%2==0) j=j-1; if (i<j) {r[i]=r[j];i=i+1;}

while (i<j && r[i]%2==1) i=i+1; if (i<j) {r[j]=r[i];j=j-1;}

}

r[i]=x;

}

1. 设计判断单链表中元素是否是递增的算法。

int isriselk(lklist \*head)

{

if(head==0||head->next==0) return(1);else

for(q=head,p=head->next; p!=0; q=p,p=p->next)if(q->data>p->data) return(0);

return(1);

}