

一. 选择题

1. 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度

2. 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. $(rear-front)\%m$

3. 在一棵高度为 h 的满二叉树中, 结点总数为

- A. 2^{k-1} B. 2^k C. $2^{k-1}+1$ D. $\lfloor \log_2 k \rfloor + 1$

4. 若用冒泡排序对关键字 $\{18, 16, 14, 12, 10, 8\}$, 进行从小到大的排序, 所需进行的关键字比较次数是

- A. 10 B. 15 C. 21 D. 34

5. 对于一个头指针为 $head$ 的带头结点的单链表, 判定该表为空表的条件是

- A. $head == null$ B. $head \rightarrow next == null$
C. $head \rightarrow next == head$ D. $head != null$

6. 的长度是指

- A. 串中所含不同字母个数 B. 串中所含字符个数
C. 串中所含不同字符个数 D. 串中所含非空格字符个数

7. 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 的节点个数是

- A. 9 B. 11 C. 15 D. 不确定

8. 下列四个序列中, 那一个是堆

- 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

9. 已知二叉树的前序序列为 ABDCEFG, 中序序列为 DBCAFEF 则其后序序列为

A. DCBAFGE B. DCBFGEA C. DCBFEGA D. DCBGFEA

10. 在下面的程序段中, 对 x 的赋值语句的频度为

```
for i:=1 to n do
```

```
    for j:=1 to n do
```

```
        x:=x+1;
```

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

二. 填空题

1. 假设一个 15 阶的上三角矩阵 A 按行优先顺序压缩存储在一维数组 B 中, 则非零元素 $a_{9,9}$ 在 B 中的存储位置 $k=$. (注: 矩阵元素下标从 1 开始)

2. 由五个分别带权值为 {9, 14, 7, 5, 2} 的叶子结点构造一棵哈夫曼树, 则该树的带权路径长度为 .

3. 当增量 $d=1$ 时, 该趟希尔排序与 排序基本一致。

4. 在一个长度为 n 的顺序表中第 i 元素 ($1 \leq i \leq n$) 之前插入一个元素时, 需向后移动

个元素。

5. 设有二维数组 $A[0..9, 0..19]$, 其每个元素占两个字节, 第一个元素的存储地址为 100, 若按列优先顺序存储, 则元素 $A[6, 6]$ 存储地址为 .

三. 试利用广仪表取表头 $head(ls)$ 和表尾为 $tail(ls)$ 的基本运算, 将原子 d 从下列表中分解出来, 请写出每一步的运算结果。

$L = ((a, (b)), ((c, d)), (e, f))$

四. 已知带权有向图如图所示, 画出该图的十字链表表存储结构结构图。

五. 已知关键字序列:

{20, 30, 50, 52, 54, 60, 66, 68, 70}

试从空树开始, 画出一棵 2-3 树(B-树)插入关键字的建树过程。若删除 50, 68, 画出每一步执行 2-3 树的状态。

六. 六. 已知关键字序列:

{10,11,23,36,50,9,16,7,56,80,90}, 试从空树开始, 建立一棵二叉排序树, 并求在等概率下, 查找的平均长度。

七. 已知二叉树以二叉链表存储, 分析如下算法, 说明算法功能, 二叉链表结构:

```
typedef struct tnode
{
    elemtype data;

    struct tnode *lchild;

    struct tnode *rchild;
} *bintree;

int f1(bintree t, bintree q, bintree p)
{
    if(t) {if(t==q) return(f2(q,p));

        else{if (f1(t->lchild,q,p)

            return 1

            else return(f1(t->rchild,q,p)

        }

        return 0;
    }

    int f2(bintree pq, bintree p)

    {if(pq) {if (pq==p) return 1;

        else {if (f2(pq->lchild,p)

            return 1

            else return(f2(pq->rchild,p)

        }

    }
```

}

return 0;

}

