江苏大学
硕士研究生入学考试样题

科目代码： 851 **A卷**

科目名称 数据结构 满分： 150分

注意：①认真阅读答题纸上的注意事项；②所有答案必须写在答题纸上，写在本试题纸或草稿纸上均无效；③本试题纸须随答题纸一起装入试题袋中交回！

一、选择题(每小题 1 分，共 10 分)

1. 在存储数据时，通常不仅要存储各数据元素的值，而且要存储(　　)。

   A. 数据的操作方法　　B. 数据元素的类型

   C. 数据元素之间的关系　　D. 数据的存取方法

2. 长度分别为 m 和 n 的升序顺序表，若将它们合并为一个长度为 m+n 的升序顺序表，则最坏情况下算法执行的比较次数为(　　)。

   A．m+n-1　　B. m×n　　C. min(m,n)　　D. max(m,n)

3. 四个元素 1,2,3,4 依次进栈，允许进栈、退栈操作交替进行，则(　　)不可能是其出栈序列。

   A. 1,2,3,4　　B. 4,1,3,2　　C. 1,4,3,2　　D. 4,3,2,1

4. 若将 n 阶下三角矩阵 A 按行优先压缩存放在一维数组 B[1…n(n+1)/2+1]中，则存放到 B[k]中的非零元素 $a_{i,j}(1\leqslant i,j\leqslant n)$的下标 i、j 与 k 的对应关系是(　　)。

   A. i(i+1)/2 + j　　B. i(i-1)/2 + j-1

   C. j(j -1)/2 + i　　D. j(j-1)/2 + i-1

5. 已知广义表 L=((a,b),(c,d))，Tail 是取表尾操作。广义表运算式 Tail(L)的操作结果是(　　)。

   A. (c,d)　　B. c,d　　C. ((c,d))　　D. d

6. 具有 10 个叶结点的二叉树中有(　　)个度为 2 的结点。

   A.8　　B.9　　C. 10　　D. 11

7. 设某无向图有 n 个顶点和 e 条边，则该图的邻接表中有(　　)个边结点。

   A. n　　B. e　　C. n+e　　D. 2e

8. 若一个有向图的顶点不能排成一个拓扑序列，则判定该有向图(　　)。

   A.含有多个出度为 0 的结点　　B.是个强连通图

   C.含有多个入度为 0 的顶点　　D.必然有环

9. 当采用分块查找时，数据的组织方式的特点是(　　)。

   A.数据分成若干块，每块内数据必须有序，块间也必须有序

   B.数据分成若干块，每块内数据必须有序，但块间不必有序

C.数据分成若干块，每块内数据不必有序，但块间必须有序

D.数据分成若干块，每块内数据不必有序，块间也不必有序

10. 简单选择排序算法中，关键字的总比较次数为(　　)。

A.O(n)　　B. $O(\log_2 n)$

C. $O(n^2)$　　D. $O(n\log_2 n)$

二、填空题(每题 2 分，共 10 分)

1. 双循环链表中，每个元素结点包含数据域 data、前驱指针 prior 和后继指针 next。当带头结点的双循环链表 L（假设 L 即为头结点指针）只有一个元素结点时，其判定条件是_____。

2. 串是一种特殊的线性表，两个串相等的充分必要条件是_____。

3. n 个顶点的连通图用邻接矩阵表示时，该矩阵至少有_____个非零元素。

4. 假设二叉排序树结点的关键字是左小右大，那么当删除二叉排序树中一个左右子树均非空的结点 p 时，可以用结点 p 的_____子树中关键字最小的结点 s 的值来代替 p 结点的值，并删除结点 s。

5. AVL 树得名于它的发明者 G. M. Adelson-Velsky 和 E. M. Landis，他们在 1962 年的论文《An algorithm for the organization of information》中发表了它。AVL 树的中文说法是_____ 。

三、应用题(共 80 分)

1. (13 分)假设在树中，结点 x 是结点 y 的双亲时，用(x,y)表示树边。已知由 T1、T2、T3 共 3 棵树组成的森林，其中 T1 的树边集合为{(A,B), (A,C)}，T2 的树边集合为{(D,E), (E,F)}，T3 的树边集合为{(G,H), (G,I), (G,J), (H,K), (H,L), (J,M), (J,N), (J,O), (M,P) }。

要求：

(1) 用树形表示法画出该森林。

(2) 画出该森林中 T3 的孩子兄弟表示法。

2. (7 分)已知 4 个字符 A、B、C、D 的哈夫曼编码分别是 1, 01, 000, 001，由这 4 个字符组成的一段文本所对应的哈弗曼编码序列如下：

1	0	0	1	0	0	0	0	1	1	0	1

要求：

(1) 画出以 A、B、C、D 这 4 个字符为叶子的哈夫曼树。

(2) 写出上述哈弗曼编码序列所对应的文本串。

3. (15 分)已知一带权有向图如图 1 所示。要求：

(1) 画出该带权有向图对应的邻接矩阵。

(2) 依据你所画的邻接邻接矩阵，写出从顶点 V1 出发的唯一的深度优先遍历序列。

(3) 请给出以顶点 V1 为源点、其他各个顶点为终点的前 3 条最短路径及其长度的求解过程。

4. (7 分)图 2 表示某一地区的通信网络，顶点表示不同城市，边表示城市间的通信线路，边上的权值表示城市间通信线路的造价。现要求构建造价最小的能连通 A、B、C、D、E、F 这六个城市的通信网络，请给出具体解决方案和构造过程，并给出造价最小的能连通这六个城市的通信网络。

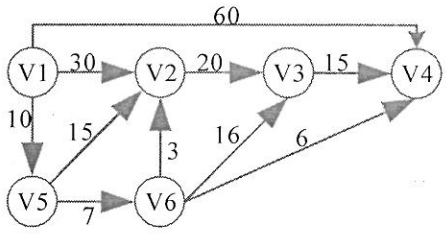

图 1 带权有向图

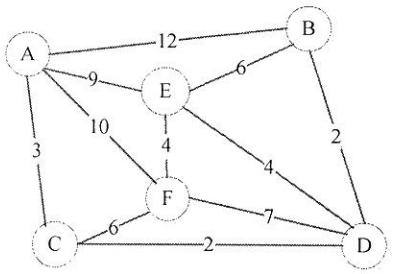

图 2 通信网络

5. (8 分)对 4 个数据元素组成的集合 S={203，305，768，789}执行查找操作，各元素对应的查找概率分别为 p1=0.35，p2=0.05，p3=0.25，p4=0.35，集合 S 中元素保存在长度为 4 的顺序表中，问：

(1) 若对该组数据采用折半方法查找，画出对应的折半查找树。

(2) 计算查找成功时的平均查找长度 ASL_{succ}。

6. (9 分)设哈希函数 H(key)=key % 7（即 key 对 7 取模），表长为 10，请利用该哈希函数把一组关键字数据 1,13,12,34,38,27,22 插入到哈希表中，要求：

(1) 使用链地址法构造哈希表。

(2) 计算等概率情况下查找成功时的平均查找长度 ASL_{succ}。

7. (12 分)使用快速排序法对一组关键字序列 F= (21，25，5，17，9，23，30)进行从小到大次序排列，试问：

(1) 若每趟都选择关键字（子）序列的第 1 个记录为枢轴点，写出每一趟的排序结果。

(2) 如果初始的关键字序列为(5，9，17，21，23，25，30)，请问快速排序法的时间性能有何变化？说出变化的原因。

8. (9 分)已知关键字序列 F={12,22,26,40,18,14,38,20,30,16,28}，要求：

(1) 若采用链式基数排序方法排序，请写出第一趟“分配”之后各队列的状态和第一趟“收集”之后的关键字序列。

(2) 简要说出基数排序和其他排序方法的区别。

四、简答题(共 30 分)

1. (10 分)队列采用顺序存储结构时，通常会使用循环队列，试问：

(1) 什么是循环队列？队列的顺序存储中引入循环队列的目的？

(2) 假设以数组存放循环队列中的元素，数组的大小为 maxsize，同时以 *rear* 和 *length* 分别指示环形队列中的队尾位置和队列中所含元素的个数。试给出该循环队列的队空条件和队首位置的计算表达式。

2. (6 分)对一个 m 行 n 列的一般矩阵 M，求其转置矩阵 N 的算法可描述为：

```
for(col=0;col<n;++col)
    for(row=0;row<m;++row)
        t[col][row]=s[row][col];
```

【注】/*s 为矩阵 M 的二维数组存储结构，t 为矩阵 N 的二维数组存储结构*/

试分析稀疏矩阵在三元组顺序存储结构下的转置操作能否采用以上一般矩阵的转置算法实现？请说明理由。

3. (6 分)设一棵 k 叉树的结点数目为 n，用多重链表表示其存储结构（同构的，即每一个结点除了数据域外，还设有 k 个指针域），试计算该 k 叉树的多重链表中的空指针域个数，写出推导过程。

4. (8 分)现有一文件 F 含有 3000 个记录，其中只有少量记录次序不对，且它们距离正确位置不远，如果以比较和移动次数作为度量，那么，现在拟采用直接插入排序、堆排序、二路归并排序将其排序，请问最好采用哪种排序方法？解释一下你所采用的排序方法的理由。

五、算法设计题(共 20 分)

注：

(1)算法中可使用试卷最后给出的存储结构。

(2)若算法中使用了其他的存储结构和运算，请给出其定义和实现。

1. （10 分）试编写一算法，对头指针为 head 的单链表实现就地逆置，即利用原表的存储空间将线性表（a_1，a_2，…，a_n）逆置为（a_n，a_{n-1}，…，a_1）。单链表可以设或不设头结点，但是在你的算法中一定要说明是否设头结点。

(1) 请给出用自然语言描述的算法的基本设计思想；

(2) 根据设计思想写出用类 C 语言或 C 语言或 C++语言描述的算法，关键之处请给出简要注释。

2. (5 分)二叉树以二叉链表形式存储，试编写创建二叉树的算法。

(1) 请给出用自然语言描述的算法的基本设计思想；

(2) 根据设计思想写出用类 C 语言或 C 语言或 C++语言描述的算法，关键之处请给出简要注释。

3. (5 分)从根到叶子结点的最大距离称为树的半径。现在给定一个无向连通图，无向连通图存储结构可以是邻接矩阵或邻接表。现要求寻找一棵最小半径的生成树，并输出生成树的顶点序列和半径值。请给出用自然语言描述的算法的基本设计思想。

//单链表的存储结构描述如下：

```
// 结点类
template <class ElemType>
struct Node {
```

```
    ElemType    data;                    // 数据域
    Node<ElemType>    *next;             // 指针域
    Node(){next = NULL;}          // 无参数的构造函数，可用于构造头结点
    };
// 单链表类
template <class ElemType> class LinkList
{   protected:
    Node<ElemType> *head;      // 头结点指针
  public:
    LinkList(){head = new Node<ElemType>();} //构造函数，构造一个只有头结点的空链
表
    virtual ~LinkList(){Clear(); delete head;}       // 析构函数
    void Clear();//删除单链表中所有元素结点，使单链表成为只有头结点的空表
};
//二叉树的存储结构描述如下：
// 二叉树结点类
template <class ElemType>
struct BinTreeNode
{
// 数据成员:
    ElemType data;                                  // 数据域
    BinTreeNode<ElemType>    *leftChild;// 左孩子指针域
    BinTreeNode<ElemType>    *rightChild;    // 右孩子指针域

//    构造函数:
    BinTreeNode();            // 无参数的构造函数
    BinTreeNode(const  ElemType  &d,// 已知数据元素值,指向左右孩子的指针构造一个
结点
          BinTreeNode<ElemType> *lChild = NULL,
          BinTreeNode<ElemType> *rChild = NULL);
};

// 二叉树结点类的实现部分
template <class ElemType>
BinTreeNode<ElemType>::BinTreeNode()
// 操作结果：构造一个叶结点
{
    leftChild = rightChild = NULL;    // 叶结点左右孩子为空
}
```

```
template <class ElemType>
BinTreeNode<ElemType>::BinTreeNode(const ElemType &d,
      BinTreeNode<ElemType> *lChild, BinTreeNode<ElemType> *rChild)
// 操作结果：构造一个数据域为 d,左孩子为 lChild,右孩子为 rChild 的结点
{
      data = d;                    // 数据元素值
      leftChild = lChild;          // 左孩子
      rightChild = rChild;         // 右孩子
}
// 二叉树类
template <class ElemType>
class BinaryTree
{
protected:
//   二叉树的数据成员:
      BinTreeNode<ElemType> *root;
      void Destroy(BinTreeNode<ElemType> * &r);   // 销毁以 r 为根的二叉树
public:
      BinaryTree();        // 无参数的构造函数
      virtual ~BinaryTree();// 析构函数
};
// 二叉树类的实现部分
template <class ElemType>
BinaryTree<ElemType>::BinaryTree()
// 操作结果：构造一个空二叉树
{ root = NULL;}
template <class ElemType>
BinaryTree<ElemType>::~BinaryTree()
// 操作结果：销毁二叉树
{    Destroy(root);}
```