机密★启用前

江苏大学2011年硕士研究生入学考试试题

科目代码： 851 **科目名称：** 数据结构

考生注意:答案必须写在答题纸上,写在试卷、草稿纸上无效!

一、单项选择题(每小题 1 分,共 10 分)

1. 线性表是一个具有 n 个(　　)的有限序列。
   (A) 表元素　　(B)字符　　(C)数据元素　　(D) 数据项
2. 设有数组 A[i,j],数组的每个元素长度为 4 字节,i 的值为 1 到 50,j 的值为 1 到 60,按行优先顺序存储,基地址为 200,则元素 A[18][25]的存储地址为(　　)。
   (A)3700　　(B)4376　　(C)3900　　(D)4620
3. 下列存储形式中,(　　)不是树的存储形式。
   (A)双亲表示法　(B)孩子兄弟表示法　(C) 孩子链表表示法　(D)顺序存储表示法
4. 既希望较快的查找又便于线性表动态变化的查找方法是(　　)。
   (A) 顺序查找　(B) 折半查找　(C) 基于属性查找　(D) 索引顺序查找
5. 下面关于串的叙述中,哪一个是正确的?(　　)
   (A) 空串是指长度为 0 的串　　(B) 空串是由空格构成的串
   (C) 串只可以采用链式存储结构　　(D) 串只可以采用顺序存储结构
6. 若一个具有 n 个结点、k 条边的非连通无向图是一个森林(n>k),则该森林中必有(　　)棵树。
   (A) k　　(B) n　　(C) n-k　　(D) n+k
7. 静态链表中指针表示的是(　　)。
   (A) 内存地址　　(B)下一元素地址　　(C)前一元素地址　　(D)左、右孩子地址
8. 下面说法不正确的是(　　)。
   (A)广义表的表头总是一个广义表　　(B)广义表的表尾总是一个广义表
   (C)广义表难以用顺序存储结构　　(D)广义表可以是一个多层次的结构
9. 对于双向循环链表,在两个结点之间插入一个新结点需修改的指针共(　　)个。
   (A) 2　　(B) 3　　(C) 4　　(D) 5
10. 将两个长度为 n 的递增有序表归并成一个长度为 2n 的递增有序表,最少需要进行关键字比较(　　)次。
    (A) 1　　(B) n-1　　(C) n　　(D) 2n

二、填空题(每空 1 分,共 10 分)

1. 设有一个顺序栈 S,元素 S1,S2,S3,S4,S5,S6 依次进栈,如果 6 个元素的出栈顺序为 S2,S3,S4,S6,S5,S1,则顺序栈的容量至少应为_____。
2. 已知广义表 LS=(a,(b,c,d),e),运用 head 和 tail 函数取出 LS 中原子 b 的运算是_____。
3. 已知有序表为(13,19,24,35,47,50,62,83,95,115,138),当用二分法查找 47 时,需_____次查找成功。
4. 下面程序段中带下划线的语句的重复执行次数是_____。

```
for(i=1;i<=n;i++) { j=1; while (j<=i) {x+=1;  j++;}}
```

5. 根据一组记录(66,52,60,74,58)依次插入结点生成一棵平衡二叉树时,当插入到值为_____的结点时需要进行旋转调整。
6. 若深度为 10 的完全二叉树的第 10 层有 3 个叶结点,则该完全二叉树共有_____个结点。
7. 组成串的数据元素只能是_____。
8. G 是 n 个顶点的有向网,则对 G 求最短路径的弗洛伊德(Floyd)算法的时间复杂度为_____。

9. 设循环队列用下标范围是 0 到 m 的数组 Q 存放队列元素值,队列的头指针为 front,指向队首元素的前一个位置,尾指针为 rear,指向队尾元素位置,则队列 Q 中当前所含元素个数为_____。

10. 希尔排序法、快速排序法、堆排序法和二路归并排序法四种排序法中,要求辅助空间最多的是_____。

三、应用题(共 80 分)

1.(16 分)某树的双亲-孩子存储表示如图 1 所示:

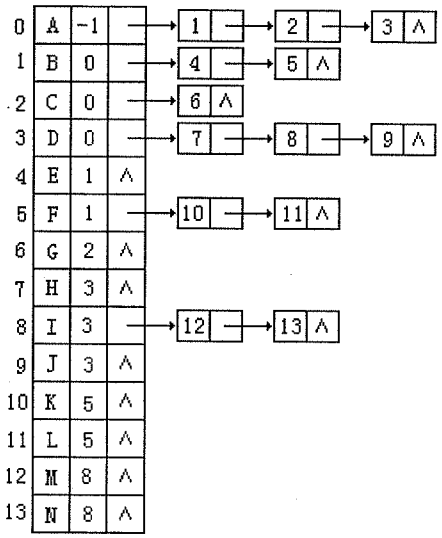

图 1

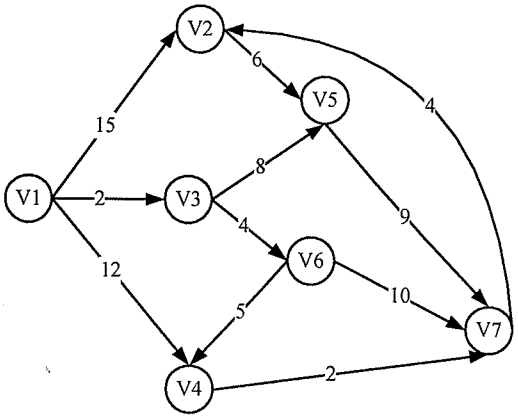

图 2

要求:

(1) 画出该树的形态。

(2) 写出该树的先根遍历和后根遍历序列。

(3) 画出该树所对应的二叉树的后序后继线索二叉树。

2.(8 分)假设 T 是一棵高度为 5 的二叉树,T 中只有度为 0 和度为 2 的结点,试问:

(1) T 树的可能的最大结点数 Nmax=?请画出这样的一棵树。

(2) 可能的最小结点数 Nmin=?请画出这样的一棵树。

3.(22 分)已知如图 2 所示有向网。要求:

(1) 画出该有向网的邻接表

(2) 画出该有向网的邻接矩阵。

(3) 求基于你的邻接表的从顶点 V1 出发的 DFS 序列以及 DFS 生成树。

(4) 用 dijkstra 算法,求从源点 V1 出发到其它各终点的最短路径以及长度,请给出执行算法过程中各步的状态。

4.(16 分)有一结点的关键字序列 F={31,26,35,13,80,32,96,39,67,23,22,91},

哈希函数为:H(K)=K%11,其中 K 为关键字,哈希地址空间为 0～13。要求:

(1) 画出以线性探测再散列处理冲突的方法所构造的哈希表。

(2) 在等概率情况下查找成功时的平均查找长度 ASL()是多少?该哈希表的装填因子 α 又是多少?

(3) 按各关键字在该关键字序列中的顺序依次插入一棵初始为空的排序二叉树,画出最后得到的二叉排序树,并计算出在等概率情况下查找成功时的平均查找长度。

5. (18 分)已知关键字序列 F={36,24,38,28,22,46,26,48,34,20,30}。要求:

（1） 将该序列调整为“大顶”堆,给出调整的过程。

（2） 写出直接选择排序的前 3 趟排序结果。

（3） 写出二路归并排序的前 3 趟排序结果。

四、简答题(共 30 分)

1.(6 分)线性表链式存储结构有哪些优缺点?单链表中,为什么会有带头结点和不带头结点单链表两种?

2.(8 分)设 T 是一棵平衡二叉树,给定关键字 K,如果在 T 中查找 K 失败,且查找路径上的任一结点的平衡因子皆为零,试回答用平衡二叉树插入算法在 T 中插入关键字为 K 的新结点后,树 T 的高度是否一定增加?并回答为什么。

3.(8 分)树形选择排序与简单选择排序相比较,优缺点是什么?请叙述堆排序算法的基本思想，并说明堆排序算法与树形选择排序算法相比有何优点?

4.(8 分)试证明:具有 n 个结点的完全二叉树的高度为 $\log_2(n+1)$ 的上取整。

五、算法设计题(共 20 分)

1.(10 分)已知两个单链表 A 和 B,其头指针分别为 ha 和 hb,编写一个算法将单链表 A 插入到单链表 B 的第 j 个元素之前。

2.(10 分)设二叉树以二叉链表为存储结构,写出实现二叉树的层次遍历算法。

注意:

(1) 可用(类)Pascal 语言或(类)C 语言或 C++语言描述你的算法;

(2) 请简要描述你的算法思想;

(3) 若你的算法是(类)Pascal 或(类)C 语言编写,则请给出相应的存储结构描述;

(4) 若你的算法是用 C++语言描述,则可参考使用以下给出的相关存储结构的类定义,算法中可以使用类中已列出的成员函数。若在你的算法中使用了未列出的成员函数,则要写出该成员函数的完整算法描述。若在你的算法中使用了未列出的其他的辅助存储结构,也请给出其类定义。

```
//单链表的类定义
template <class type> class linklist;   //单链表前视声明
template <class type> class node{//单链表结点类
   friend class linklist <type>;       //定义单链表类 linklist <type>为结点类的友元
   private:
       node <type>    *next;     //链指针域
   public:
       type data;                  //数据域
       node (node <type> *pnext = NULL) {next = pnext;}//构造函数,用于构造头结点
};
template <class type> class linklist{       //单链表类定义
 private:
    node <type>    *head;                    //指向头结点的头指针
 public:
    linklist ( ){ head = new node <type> ( ); head->next=NULL; }//构造函数
    ~linklist ( ); //析构函数
};
```

```
//二叉链表的类定义:
template <class Type> class BinaryTree;   //二叉链表类前视声明,以便使用友元
template <class Type> class BinTreeNode { //结点类
friend class BinaryTree<Type>;
private:
   BinTreeNode<Type> *leftChild,   *rightChild;      //结点的左、右孩子指针域
   Type data;              //结点的数据域
public:
   BinTreeNode ( ) : leftChild (NULL), rightChild (NULL) { } //构造函数,构造一个空结点
   BinTreeNode (Type d, BinTreeNode<Type> *lp = NULL, BinTreeNode<Type> *rp =NULL ) : data
         (d),   leftChild (lp), rightChild (rp) { }
     //构造函数,构造一个数据域的值为 d 的结点
};
template <class Type> class BinaryTree { //二叉链表类
private:
   BinTreeNode <Type> *root;     //二叉树根结点指针
public:
    BinaryTree ( ) : root (NULL) { }     //构造函数
   ~BinaryTree ( ) { destroy ( root ); }       //析构函数
};
```