信息科学技术学院 2006-2007 学年第一学期

本科生期中考试试卷

考试科目：数据结构与算法 A　考试时间：2006 年 11 月 10 日

_____专业_____级_____班　考场 三教　　室

姓名_____ 学号_____**教师:** 张铭、赵海燕、王腾蛟

题号	一	二	三	四	总分
得分					

注意事项：

1. 第一题填空答案请写在试卷的留白上，其他题目都写在空白答题纸上。若超出所留空间，请使用后面的续页。
2. 本试卷对于算法改进、算法设计都有质量要求，请尽量按照试题中的要求来写算法。否则将酌情扣分。
3. 您所写的算法应该申明算法思想，并在算法段加以恰当的注释。

诚实考试宣言：

我真诚地保证严格遵循考场纪律：

1. 诚实地用自己所掌握的知识和能力，独立地回答试卷；
2. 不夹带任何有作弊嫌疑的书本、纸条或写在其他地方的提示性资料进入考场；
3. 正式考试（包括发卷和收卷）过程中，不与考场的同学交谈或借物品，有问题则举手等待监考老师来协助解决；
4. 不与他人共享答案和解题方法：不抄袭或偷看他人的答案，也不告诉他人答案或解题方法，更不传递或故意高举试卷把答案给他人看；
5. 服从监考老师的管理；
6. 不携带试卷出考场，也不会把考题以任何方式泄露出去。

保证人：_____

不在“诚实考试宣言”之后签名的试卷，计零分或根据作弊与否的情况上报学校处理！

一、填空（30 分）

1. 在有 n 个元素的顺序表中，若想在第 i 个元素（1≤i≤n＋1）之前插入一个元素时需要向表尾方向移动_____个元素。

2. 设栈 S 和队列 Q 的初始状态为空，元素 a、b、c、d、e、f 依次通过栈 S，一个元素出栈后即进入队列 Q。若这 6 个元素出队列的顺序是 b、d、c、f、e、a，请写出它们进栈出栈的顺序为：

   _____

3. 以数据集{4，5，6，7，10，12，18}为结点权值所构造的哈夫曼树，其带权外部路径长度为_____。

4. 在一棵二叉树中，度为零的结点的个数为 n_0，度为 2 的结点个数为 n_2，则有 n_0 =_____

5. 在用数组实现的完全二叉树中，当$0 < i < n$时，结点 i 的父母是结点_____，当$2i+1 \le n$时，结点 i 的左子女是结点_____，否则 i 没有左子女。

6. 4 个结点的连通有序树有_____种不同的形状。由 4 个结点 A、B、C、D 可以组成_____个不同的连通有序树。

7. 对于非空满 K 叉树，其分支结点的数目为 n，则其叶结点的数目为 _____。

8. 若按照下面给定的公式来定义模板 P 的特征向量 next 数组，则模板 P = abababa 的特征向量为_____。

$$next[j]=\begin{cases}-1, & \text{当 j = 0时}\\ \max\{k: 0<k<j \ \&\ P[0...k-1] = P[j-k...j-1]\}, & \text{当k存在时}\\ 0, & \text{其他情况}\end{cases}$$

二、辨析题（20 分）

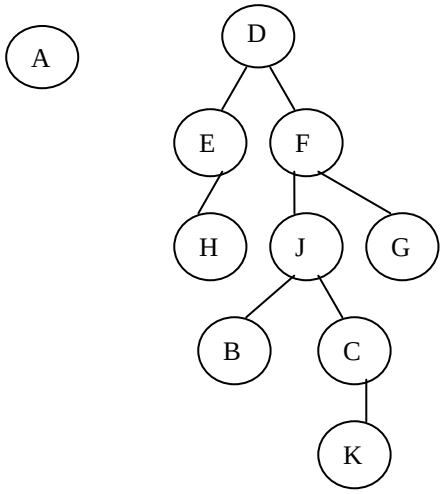
1. 若较频繁地对一个线性表进行插入和删除操作，该线性表宜采用何种存储结构？为什么？

2. 请简述下面算法的功能

```
Func1(Stack S2, Stack S1) {
    // Stack 是顺序栈类型
    ELEM x;          // ELEM 是栈中的元素类型
    InitStack(tmp);
    InitStack(S2);
    while (!StackEmpty(S1))  {
            x = top(S1);
            pop(S1);
```

```
            push(tmp, x);
        }
        while (!StackEmpty(tmp)) {
            x = top(tmp);
            pop(tmp);
            push(S1, x);
            push(S2, x);
        }
}
```

3. 下图是父结点表示法表示的树，采用带路径压缩的 Find 算法，请画出查找 C 之后的树。



4. 树的层次周游是按照层次顺序对树中的每个结点访问而且只访问一次。请判断下面这个算法是否能正确地按照层次顺序周游树。

(1) 如果是，请写出该算法的主要思路；

(2) 如果否，请简要说明理由，并改正（请尽量保留原算法框架）。

```
template <class T>                                                  /* 第 1 行 */
void Tree<T>::WidthTraverse1(TreeNode<T>* root){                    /* 第 2 行 */
        using std::queue;           //使用 STL 队列置               /* 第 3 行 */
        queue<TreeNode<T>*> aQueue;                                 /* 第 4 行 */
        TreeNode<T>* pointer=root;                                  /* 第 5 行 */
        if (pointer){                                               /* 第 6 行 */
            aQueue.push(pointer);                                   /* 第 7 行 */
            while(!aQueue.empty()) {                                /* 第 8 行 */
                pointer=aQueue.front(); //取队列首结点指针          /* 第 9 行 */
                Visit(pointer->Value());//访问当前结点              /* 第 10 行 */
                while (pointer->RightSibling())                     /* 第 11 行 */
                {                                                   /* 第 12 行 */
                    if (pointer->LeftMostChild())                   /* 第 13 行*/
                        aQueue.push(pointer->LeftMostChild());      /* 第 14 行 */
```

```
                pointer=pointer->RightSibling();          /* 第 15 行 */
                Visit(pointer->Value());                  /* 第 16 行 */
            }                                             /* 第 17 行 */
            aQueue.pop(); //出队列                        /* 第 18 行 */
        }                                                 /* 第 19 行 */
    }                                                     /* 第 20 行 */
}                                                         /* 第 21 行 */
```

三、算法填空（15 分）

下面的算法将一个用带度数的后根次序法表示的森林转换为左子结点/右兄弟结点法表示。请利用题目所给出的树结点 ADT 和栈 ADT，填充算法的空格，使其成为完整的算法。空格中可能需要填写 0 至多条语句（或表达式）。

```
template<class value_type>
class stack {
public:
    bool empty();                              // 判栈空
    int  size();                               // 返回栈的大小
    value_type& top();                         // 读栈顶
    void push(const value_type& _X);           // 入栈
    void pop()                                 // 出栈
};

struct Node<T>                          //带度数后根次序表示法的结点结构
{
    T info;                             // 结点的数据信息
    int drgree;                         // 结点的度数信息
};

template<class T>
class TreeNode
{
    public:
       bool isLeaf();                           //如果结点是叶，返回 true
       T Value();                               //返回结点的值
       TreeNode<T>* LeftMostChild();            //返回第一个左孩子
       TreeNode<T>* RightSibling();             //返回右兄弟
       void setValue(T&);                       //设置结点的值
        void setChild(TreeNode<T>* pointer);    //设置左子结点
       void setSibling(TreeNode<T>* pointer); //设置右兄弟
};

TreeNode<T> *Convert(Node* nodes, int size)
{
      TreeNode<T> *cur, *temp1, *temp2;
      Stack<TreeNode<T> *>Cstack;
      for (int i=0; i<size; i++)
```

```
    {
        cur=new TreeNode<T>(nodes[i].info);
        if (nodes[i].degree==0)
                    // ?1
        else
        {
            assert (nodes[i].degree<= Cstack.size());
            temp2= NULL;
            for (int j=0;  // ?2  ; j++)
            {
                temp1= Cstack.top();      Cstack.pop();
                // ?3
                temp2=temp1;
            }
            // ?4
            Cstack.push(cur);
        }
    }
    cur = temp2=NULL;
    wilie (!Cstack.isempty())
    {
        cur =Cstack.top();                Cstack.pop();
        // ?5
        temp2 = cur;
    }
    return cur;
}
```

四、算法设计（35 分）（注意写算法思想和代价分析）

1. 已知 Q 是一个非空队列，S 是一个空栈，请用自然语言描述使用栈 S 将队列 Q 中的所有元素逆置的算法思想。（不需要写具体的算法实现伪码）
2. 编写一个递归函数 PrintRange，传入参数为一个 BST，一个较小值和一个较大值，按照顺序打印出介于两个值之间的所有结点。
3. 试编写一个算法，计算一棵给定二叉树的单孩子结点数。