北京大学信息科学技术学院 2007-2008 学年

第一学期本科生期末考试试卷

考试科目： 数据结构与算法 A　　考试时间：2007 年 11 月 10 日

姓名_____**学号**_____**考场**_____ **教员**：张铭、赵海燕、王腾蛟、宋国杰

题号	一	二	三	四	总分
得分					

注意事项：

1. 第一题答案请写在试卷的留白上，其他题目都写在空白答题纸上。
2. 本试卷对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。

一、 填空(共 15 空，每题 2 分，共 30 分)

1. 以下哪几种结构是逻辑结构，而与存储和运算无关（_____）。

   A. 顺序表　　B.散列表　　C.线性表　　D.单链表

2. 假设计算机的速度为每秒处理 10^{10} 个操作。现有 5 个算法，其操作数与 n 的关系如下。请填写下表，列举一个小时之内每个算法可处理的数据规模（即 n 的大小）。

算法复杂度	$n\log_{10} n$	n^2	$100n^2$	n^3	$2n$
数据规模					

3. 依次读入数据元素序列{a，b，c，d，e，f，g}进栈,每进一个元素，机器可要求下一个元素进栈或弹栈，如此进行，则栈空时弹出的元素构成的序列是以下哪些序列(____)

   A. {d ，e，c，f，b，g，a}　　B. {f，e，g，d，a，c，b}

   C. {e，f，d，g，b，c，a}　　D. {c，d，b，e，f，a，g}

4. 对二叉树从 0 开始进行连续编号，要求每个结点的编号大于其左右孩子的编号，同一结点的左右孩子中，其左孩子的编号小于其右孩子的编号，则可采用_____次序的遍历实现编号。

5. 对于键值序列{12,13,11,18,60,15,7,18,25,100}，用筛选法建堆，必须从键值为_____的结点开始。

6. 一个有 2001 个结点的完全二叉树的高度为_____（独根树高度为 0）。

7. 如果按关键码值递增的顺序依次将关键码值插入到二叉排序树中，则对这样的二叉排序树检索时，平均比较次数为_____。

8. 假设先根次序遍历某颗树的结点次序为 SACEFBDGHIJK，后根次序遍历该树的结点次序为 CFEABHGIKJDS，一颗可能的树为_____(在下面空白处给出树的结构)

9. 2-3 树是一种特殊的树，它满足两个条件：
（1）每个内部结点有两个或三个子结点；（2）所有的叶结点到根的路径长度相同；
如果一棵 2-3 树有 9 个叶结点，那么它可能有_____个非叶结点。

10. 一个高度为 h 的满 k 叉树，最多有_____个叶结点，整棵树最多有_____个结点。（独根树高度为 0）。

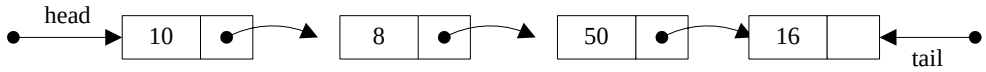
二、 辨析与简答(共 3 题，每题 5 分，共 15 分)

1. 试证明，在具有n(n>=1)个结点的m叉树中，有n(m-1)+1个指针是空的。

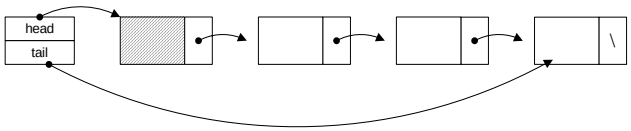
2. 求字符串"BAAABBBAA"的特征向量，并与目标串"BAAABBBCDDDCCHHHHBBBAAABBBAADD"进行匹配，请画出匹配的过程示意图。其中特征向量的定义如下：

$$next[i]=\begin{cases}-1, & \text{对于 } i=0\\ \max\{k: 0<k<i \ \&\& \ P(0...k-1)=P(i-k...i-1)\}, & \text{如果k存在}\\ 0, & \text{否则}\end{cases}$$

3. 下图(a)是不带头结点的单链表。在实际组织单链表时，通常会引入一个称为头结点（header node）的特殊结点作为表头，如图(b)所示。请讨论采用头结点的好处。



(a) 不带头结点的单链表



(b) 带头结点的单链表

三、 算法填空(25 分)

1. 由二叉树的前序遍历和中序遍历序列能确定唯一的一棵二叉树，下面程序的作用是实现由已知某二叉树的前序遍历和中序遍历序列，生成一棵用二叉链表表示的二叉树并打印出后序遍历序列，请写出程序所缺语句

```
#define MAX 100
typedef struct Node{
      char info;
      struct Node *llink, *rlink;
}TNODE;
char pred[MAX],inod[MAX];
main(int argc, char **argv) {
```

```
        TNODE *root;
        if(argc<3) exit 0;
        strcpy(pred,argv[1]);
        strcpy(inod,argv[2]);
        root=restore(pred,inod,strlen(pred));
        postorder(root);
    }
    TNODE *restore(char *ppos,char *ipos,int n){
        TNODE *ptr;    char *rpos; int k;
        if (n<=0) return NULL;
        ptr = new TNODE;
        ptr->info = _______(1);
        for (_______(2); rpos<ipos+n;rpos++)
            if (*rpos==*ppos)    break;
        k=_______(3);
        ptr->llink=restore(ppos+1, _______(4),k );
        ptr->rlink=restore (_______(5)+k,rpos+1,n-1-k);
        return ptr;
    }
    postorder(TNODE*ptr){
        if (ptr=NULL) return;
        postorder(ptr->llink);
        postorder(ptr->rlink);
        printf("%c",ptr->info);
    }
```

2. 下面的算法把整数的中缀表达式转换成后缀表达式。请利用题目所给出的栈 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）。假设运算符的优先级函数为 int priority(char operator); 为方便期间，定义优先级 {'*', '/': 3； '+', '-': 2; '(': 1; 其它:0 }

```
// 栈的抽象数据类型 ADT 定义
template <class T>                                  // 栈的元素类型为 T
class Stack {
public:                                             // 栈的运算集
    void clear();                                   // 变为空栈
    void push(const T item);                        //   item 入栈
    T pop( );                                       // 返回栈顶内容并弹出
    T top( );                                       // 返回栈顶内容但不弹出
    bool isEmpty();                                 // 若栈已空返回真
    bool isFull();                                  // 若栈已满返回真
 };
char *back_expression(char *mid_expression)  {      // 中缀表达式转换成后缀表达式
    char c_temp=0;
    char *str_temp;                                 // 结果后缀表达式
    int str_temp_pos=0;
    char stack_top_operator=0;                      // 栈顶的元素
    str_temp=new char[MAXSIZE];
    Stack <char> S;
    S.push (';');                                   // 栈底的标志元素
    while ((c_temp=*(mid_expression++))!='\0') {    // 读入表达式并解析
        if (c_temp=='(')                            // 如果操作符是 '('
            push(c_temp);
        else if (isdigit(c_temp))  {                // 如果是数字，存入 str_temp
            str_temp[str_temp_pos++]=c_temp;
```

```
            while (((c_temp=*(mid_expression++))!=0) && (isdigit(c_temp)))    {
                str_temp[str_temp_pos++]=c_temp;
            }
            mid_expression--;                                   // 回退一个
            str_temp[str_temp_pos++]='    ';                    // 从后缀表达式中退掉非法的符号
        }
        else if (c_temp==')')                                   // 如果输入字符是 ')',
                                                    （6）
        else if (isoperator(c_temp))                            // 如果输入字符是操作符
                                                    （7）
    }
    while (stack_top_operator=S.pop())!=';')
                                                    （8）
    str_temp[str_temp_pos]= '\0';
    return str_temp;
}
```

四、 算法设计与实现(30 分)

1. 编写算法，求得所有包含在串s中而不包含在串t中的字符（s中重复的字符只选一个）构成的新串r，以及r中每个字符在s中第一次出现的位置。

2. 请设计一个算法，计算一棵树的树叶的个数。树叶指的是没有子结点的结点。简单起见，树结点的定义如下：

```
struct TreeNode{
    char label;
    TreeNode * pChild;     //该结点的第一个子结点
    TreeNode * pSibling;   //该结点的下一个兄弟结点
}
```

(1) 设树的结点个数为 n，请说明你的算法的时间复杂度。

(2) 请实现你的算法。可以代码实现（语言不限），也可以写伪代码。若代码实现，必须要写出必要的注释，无注释的要扣分。若写伪代码，要求逻辑清晰。