

北方工业大学
2003 年硕士学位研究生入学考试试题

考试科目：数据结构

适用专业：计算机科学与技术

说明：算法用类 C 或类 PASCAL 语言描述，在题目之前注明自己所用语言，算法书写必须规范。

一、简要回答下列各题（共 60 分）

- 1、在一棵具有 n 个结点的 k 叉树中，除叶子结点外其余结点的度均为 k ，该树中叶子结点的个数为多少？（4 分）
- 2、在顺序表中插入一个元素时需平均移动多少个元素？具体移动元素的个数与什么因素有关？（4 分）
- 3、若待排关键字序列已经基本有序，从关键字的比较次数和移动次数考虑，对于快速排序、直接插入排序、归并排序、简单选择排序，选择哪种排序方法对其进行排序最好？为什么？（4 分）
- 4、无论在什么情况下，采用折半查找方法一定比采用顺序查找方法要快，这种说法正确吗？为什么？（4 分）
- 5、一个带权连通图的最小生成树是否唯一，什么情况下不唯一？（3 分）
- 6、线性表的顺序存储结构和链式存储结构，若只从空间开销上来考虑，能否说一种比另一种更节省空间，为什么？在什么情况下适合顺序存储结构，在什么情况下适合链式存储结构？（4 分）
- 7、采用循环链表作为存储结构的队列就是循环队列，这种说法正确吗？说明理由。（3 分）
- 8、给出模式串“aabaababc”按 KMP 算法进行模式匹配时的 next 值和 nextval 值。（4 分）
- 9、在哈希表中是怎样实现查找的？在查找过程中会出现什么问题？（4 分）
- 10、已知一组关键字为 {25, 18, 46, 2, 53, 39, 32, 4, 74, 67, 60, 11}
 - 1) 在该表上顺序查找时，在等概率情况下查找成功的平均查找长度为多少？（3 分）
 - 2) 按表中元素的顺序构造一棵二叉排序树，在此二叉排序树上进行查找时，等概率情况下查找成功的平均查找长度为多少？（3 分）
 - 3) 按表中元素的顺序构造一棵平衡二叉树，在此平衡二叉树上进行查找时，等概率情况下查找成功的平均查找长度为多少？（3 分）
- 11、对于具有 n 个顶点 e 条边的有向图，给出在邻接矩阵和邻接表两种不同存储结构下，求图中边的个数时的时间复杂度。（4 分）
- 12、在顺序查找方法中，查找之前先将待查关键字放在顺序查找表的一端，起监视哨作用，这种处理方法对提高算法的效率有用吗？说明理由。（4 分）
- 13、在具有 n 个结点的线索树中，共有多少个线索？（3 分）
- 14、在快速排序、直接插入排序、归并排序、简单选择排序这几种排序方法中，若要求排序必须在 $O(n \log_2 n)$ 的时间内完成，且要求排序是稳定的，则可选用哪种方法。（3 分）
- 15、设有一个链栈 L_s ，用无头结点的单链表结构表示，栈空时的条件是什么？（3 分）

二、(15 分) 已知顺序存储结构的线性表 L, 编写一高效算法, 删除表中大于 $mink$ 且小于 $maxk$ 的所有元素。线性表的存储结构为:

```
#define INIT_SIZE 100          (类 C 描述)
typedef struct {
    ElemType *elem;
    int      length;
    int      listsize;
} SqList;
```

```
CONST INIT_SIZE=100;          (类 PASCAL 描述)
TYPE  SqList = RECORD
    elem : ARRAY[1.. INIT_SIZE] OF ElemType;
    length : integer;
END;
```

三、(15 分) 设以顺序存储结构表示一个双向栈, 其类型定义如下:

```
#define INIT_SIZE 100      (类 C 描述)
typedef struct {
    SElemType elem[INIT_SIZE];
    int      top[2], base[2];
} SqDuStack;
```

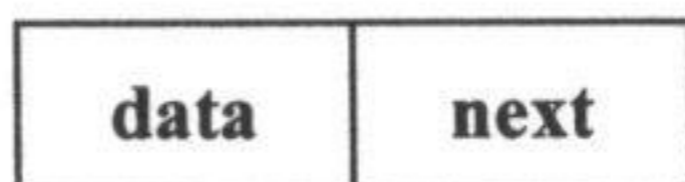
```
CONST INIT_SIZE=100;      (类 PASCAL 描述)
TYPE  SqDuStack = RECORD
    elem : ARRAY[1.. INIT_SIZE] OF SElemType;
    top , base : ARRAY[1.. 2] OF integer;
END;
```

写出栈初始化操作算法 $initstack(s)$ 和入栈 $push(s, e, i)$ 的算法, 其中用 i 为 0 或 1 表示两端的两个栈。

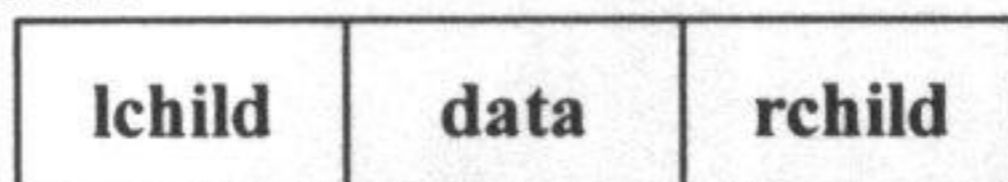
四、(15 分) 已知线性表 $L_a = (a_1, a_2, \dots, a_m)$ 和 $L_b = (b_1, b_2, \dots, b_n)$ 都采用带头单链表结构表示, 编一算法, 利用原结点合并两个链表成 L_c , 并使得:

$$L_c = \begin{cases} (a_1, b_1, a_2, b_2, \dots, a_m, b_m, b_{m+1}, \dots, b_n) & n \geq m \text{ 时} \\ (a_1, b_1, a_2, b_2, \dots, a_n, b_n, a_{n+1}, \dots, a_m) & n < m \text{ 时} \end{cases}$$

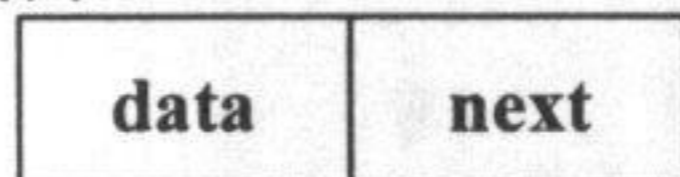
链表结点结构为:



五、(18 分) 编一算法, 打印出二叉树 T 中结点数据域值为给定值 e 的结点的所有祖先。设二叉树结点结构为:



六、(12 分) 编一算法, 统计无头结点的单循环链表 L 中的结点个数。
链表结点结构为:



七、(15 分) 设无向图 G 以邻接矩阵表示, 编写算法, 判断给定图 G 是否是连通图。
邻接矩阵结构为:

(类 C 描述)

```
#define MAXVEXNUM 20
typedef struct {
    VertexType vexs[MAXVEXNUM]; //顶点向量
    int arcs[MAXVEXNUM][MAXVEXNUM]; //邻接矩阵
    int vexnum, arcnum;
} MGraph;
```

(类 PASCAL 描述)

```
CONST MAXVEXNUM = 20;
TYPE MGraph = RECORD
    vexs:ARRAY[1.. MAXVEXNUM] OF VertexType;
    arcs:ARRAY[1.. MAXVEXNUM,1.. 1.. MAXVEXNUM] OF integer;
    vexnum, arcnum :integer;
END;
```