中 山 大 学

二00八年攻读硕士学位研究生入学考试试题

科目代码： 853

科目名称： 数据结构

考试时间： 2008 年 1 月 20 日 下 午

考 生 须 知

全部答案一律写在答题纸上，答在试题纸上的不得分！请用蓝、黑色墨水笔或圆珠笔作答。答题要写清题号，不必抄题。

一、选择题（每小题 2 分，总分 30）选择正确答案的代号写在答题纸上，注明题号。

1. 从逻辑结构上讲，可以把数据结构分为(　　)。
   A.动态结构和静态结构　　B.内部结构和外部结构
   C.顺序结构和链式结构　　D.线性结构和非线性结构

2. 抽象数据类型可以用(　　)、数据关系和基本操作来定义。
   A.数据元素　　B.数据对象　　C.原子类型　　D.存储结构

3. 算法的有穷性指（　　）
   A. 输入是有限的　B. 输出是有限的　C. 描述是有限的　D. 有穷步后终止

4. 将长度为 n 的单链表链接在长度为 m 的单链表之后，其算法的时间复杂度为(　　)
   A.O(1)　　B.O(n)　　C.O(m)　　D.O(m+n)

5. 设 A 为一个 n×n 对称矩阵，为了节省存储，将其下三角部分按行存放在一维数组 B[1..n(n+1)/2]，对下三角部分中任一元素 $a_{ij}(i \geq j)$在一维数组 B 的下标位置 k 值是(　　)
   A. i(i-1)/2+j-1　　B. i(i-1)/2+j　　C. i(i+1)/2+j-1　　D. i(i+1)/2+j

6. 一个非空广义表的表头(　　)
   A.不可能是子表 B.只能是子表　　C.只能是原子　　D.可以是子表或原子

7. 已知一棵二叉树的后序序列和中序序列分别是 dabec 和 debac，其先序序列是(　　)
   A. acbed　　B. cedba　　C. decab　　D. deabc

8. 一棵具有 4 层的 AVL 树至少有(　　)个结点（根结点是第一层 ）。
   A. 5　　B. 7　　C. 8　　D. 10

9. (　　)方法不可以用来判断一个有向图中是否存在回路。
   A.深度优先遍历　　B.拓扑排序　　C.广度优先遍历　　D.求关键路径

10. 含有 n 个顶点 e 条弧的有向图用邻接表表示,删除与某个顶点相关的所有弧的时间复杂度是(　　)
    A. O(n)　　B. O(e)　　C. O(n+e)　　D. O(n*e)

11. 适于对动态查找表进行高效率查找的组织结构是(　　)
    A.有序表　　B.分块有序表　　C.二叉排序树　　D.线性链表

12. 假定有 k 个关键字互为同义词，若用线性探测法把这 k 个关键字存入哈希(hash)表中，至少要进行多少次探测？(　　)
    A.k-1 次　　B.k 次　　C.k+1 次　　D.k(k+1)/2 次

四、算法设计（总分 50）

1. （10 分）完成下列算法

```
Status traverse(BinarTreeNode * root, Status (*visit)(ElemType e)){
/* BinaryTreeNode 为二叉链表结点类型， ElemType 为结点数据域类型
   traverse 按层次序遍历 root 所指二叉树，在遍历过程中将函数 visit 应用于每个结点
   如果 visit 应用失败， 则返回 ERROR, 否则，遍历结束后返回 OK.
*/

   initQueue(Q);
   if (root == NULL) return OK;
   enQueue(Q, root);//root 入队列
   while (_____){
        deQueue(Q, p); //队头元素出队列，且 p 为出队的元素
        if (visit(p->data)!=OK) return  _____;
        if (p->lchild)
             _____;
        if (p->rchild)
             _____;
   }
   return _____;
}
```

2. （16 分）阅读下列算法

```
Status   run_recursive_binary(const Ordered_list &the_list,
                               const Key &target, int &position)
{
   return recursive_binary(the_list, target, 0, the_list.size() - 1, position);
}

Status recursive_binary(const Ordered_list &the_list, const Key &target,
                        int bottom, int top, int &position)
/*
Pre:  The indices bottom to top define the range in the list to search for the target.
Post:  If a Record in the range of locations from bottom to top in the_list has key equal
       to target, then position locates one such entry and a code of success is returned.
       Otherwise, the Error_code of not_present is returned and position becomes undefined.
*/
```

```
{
    Record data;
    if (bottom < top) {                    // List has more than one entry.
        int mid = (bottom + top) / 2;
        the_list.retrieve(mid, data);
        if (data < target)    // Reduce to top half of list.
            return recursive_binary(the_list, target, mid + 1, top, position);
        else                                        // Reduce to bottom half of list.
            return recursive_binary(the_list, target, bottom, mid, position);
    }
    else if (top < bottom)
        return not_present;                // List is empty.
    else {                                      // List has exactly one entry.
        position = bottom;
        the_list.retrieve(bottom, data);
        if (data == target) return success;
        else return not_present;
    }
}
```

请完成下列任务：

（1） 将算法中的尾递归去掉，实现 run_recursive_binary 的非递归版本；
（2） 说明递归版本和非递归版本的时间复杂度和空间复杂度各是什么。

3. (24 分) 使用 C/C++编写一个判断无向图是否连通的算法。

要求：

（1）写出图的存储结构；
（2）用自然语言说明算法思想；
（3）使用 C 或者 C++写出算法；你的算法应该清晰、易懂；
（4）分析你的算法的时间复杂度。