# 北京大学信息科学技术学院考试试卷

**考试科目：**数据结构与算法 A **姓名：**_____ **学号：**_____

**考试时间：** 2008 年 1 月 9 日 **教师：**张铭、赵海燕、王腾蛟、宋国杰

| 题号 | 一 | 二 | 三 | 四 | 五 | 六 | 七 | 八 | **总分** |
|---|---|---|---|---|---|---|---|---|---|
| 分数 | | | | | | | | | |
| 阅卷人 | | | | | | | | | |

装订线内

不要答题

**考 场 纪 律**

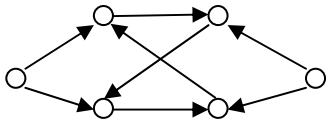
1. 请持学生证入场考试，并按指定座位就座；除必要的文具和教师指定的用具用书外，其他所有物品包括手机、呼机、MP3、电子词典、书籍、笔记、纸张等严禁带入座位，必须放在指定位置。凡有试题印制问题请向监考教师提出，不得向其他考生询问。
2. 认真、诚实、独立并在规定时间内完成答卷，严禁任何形式的违纪作弊行为；否则，本答卷成绩以 0 分记，并根据《北京大学本科考试工作与学术规范条例》给予纪律处分。
3. 提前交卷的考生不要在考场逗留，不要在门口、窗外大声喧哗。考试结束时间到，请停止答卷，在座位等候监考教师收卷并清点完毕，方可离开考场；考题和试卷不得带出考场。

以下为试题和答题纸，共 4 页。

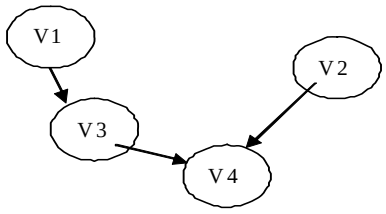
## 一、（共 30 分，每空 3 分）填空

1. 1．无向图 G=(V, E)，其中：V={a, b, c, d, e, f}， E={(a, b), (a, e), (a, c), (b, e), (c, f), (f, d), (e, d)}，对该图进行深度优先遍历，得到的顶点序列正确的是____。
   A．a,b,e,c,d,f　　B．a,c,f,e,b,d　　C．a,e,b,c,f,d　　D．a,e,d,f,c,b

2. 下图中的强连通分量的个数为_____个。



3. 设有向图G如下:




   写出所有拓扑序列:_____
   添加一条弧_____之后, 则仅有唯一的拓扑序列.

4. 请问下面哪些操作在已排序数据上实施比在无序的数据上快_____?

   A．找最小值 B. 计算算术平均值 C. 找中位数 D. 找出现次数最多的值

5. 序列{15，142，51，68，121，46，57，575，60，89，185 }按最低位优先法进行基数排序，进行一次分配和收集后得到的序列_____ 。

6. 设输入的关键码满足$k_1>k_2>\ldots>k_n$，缓冲区大小为$m$，用最小值堆进行置换-选择排序方法可产生____个初始归并段。

7. 在包含n个关键码的线性表中进行顺序检索，若检索第i个关键码的概率为$p_i$, 且分布如下：

$$p_1=\frac{1}{2}, p_2=\frac{1}{4},\ldots., p_{n-1}=\frac{1}{2^{n-1}}, p_n=\frac{1}{2^n}$$

   成功检索的平均检索长度是_____。

8. 假设计算机系统有2048个字节的磁盘块，要存储的每一条记录中4个字节是关键码，磁盘指针4个字节，64个字节是数据字段。记录已经排序，顺序地存储

在磁盘文件中。如果使用大小为4M的线性索引，那么最多有_____个索引项，最多可以有____条记录.

## 二、（25 分）辨析题

1. （6 分）对于连通的无向图，采用 Dijkstra 最短路径算法，在 Dist 数组中能否给出足够形成一棵支撑树的信息？是否能给出一棵最小支撑树(MST)？请证明你的结论。

2. （6 分）有 8 个顺串，每个顺串的第一个记录的关键码分别为 14，22，24，15，16，11，100，18，而第二个记录的关键码分别为 26，38，30，26，50，28，110，40。请画出对顺串开始 8 路合并时的败者树。从败者树输出一个全局优胜者(并有相应的一个记录进入败者树)后需对败者树进行重构，请画出输出第一个全局优胜者并进行重构后的败者树。

3. （7 分）设散列表为 HT[13]，散列函数为 h (key) = key%13。用闭散列法解决冲突，对下列关键码序列 12，23，45，57，20，03，78，32，15，33 造表。

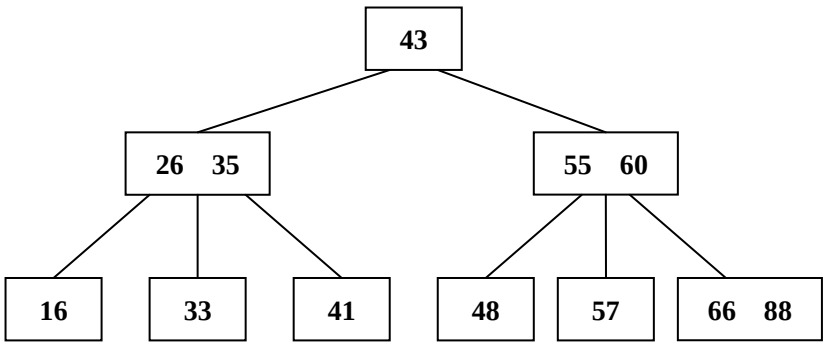
   (1) 采用线性探查法寻找下一个空位，画出相应的散列表。

   (2) 采用双散列法寻找下一个空位，再散列函数为rh (key) = (7*key) % 11 + 1，寻找下一个空位的公式为hi = ($h_{i-1}$ + rh (key)) % 13，h1 = h(key)，画出相应的散列表。

4. （6 分）设有 3 阶 B 树如下图所示：

   (1) 在该B树上插入关键码97，画出插入后的B树。

   (2) 在（1）得到的B树基础上删除66，画出删除后的B树。



## 三、（共 15 分，每空 3 分）算法填空

设 a 和 b 是两个分别包含 n 个已排序数据的数组。如果将 a 和 b 中的 2n 个数进行全体排序，此时处于整体排序后的序列中第 n 个位置上的数称为这 2n 个元素的中位数。下面的代码片段用来计算数组 a 和 b 中的中位数，请在空白的位置填写合适的语句，以完成此功能。

```
int bigger(int a, int b) {                          // 函数返回 a,b 中较大者，相等则返回 a
    if (a > b) return a; else return b;
}

int search(int a[],int b[], int s, int t, int n)   { // 查找区间[s,t]
    int i = (s + t) / 2;
    int j = (n - i - 2);
    if (_____________(1) _____________)
        return bigger(b[j],a[i]);              // 返回 b[j],a[i] 中较大者
    if (a[i + 1] < b[j])
        return _____(2)_____;
    else return _____(3)_____ _____;
}
int search_help(int a[],int b[], int n)   {
    if (_____________(4)_____________)
        cout << b[n - 1];
    else if _____________(5)_____________)
        cout << a[n - 1];
    else cout << search(a, b, 0, n - 1, n) << endl;
}
```

## 四、（共 30 分，每题 10 分）算法设计

1. 简单路径条数。

给出图的 ADT 如下：

```
Class Graph {
Public:
    int VerticesNum();
    int EdgesNum();
    Edge FirstEdge(int oneVertex);
    Edge NextEdge(Edge preEdge);
    bool IsEdge(Edge onEdge);
    int FromVertex(Edge oneEdge);
    int ToVertex(Edge oneEdge);
```

};

设计算法过出图 G 中从顶点 i 到顶点 j 之间长度为 len 的简单路径条数:

int GetPathNum_Len(Graph& G, int i, int j, int len);

2. 编写考虑到墓碑（TOMB）问题的散列表插入算法，算法中不允许有重复关键码插入。下面是散列表的定义。

```
template <class Key, class Elem, class KEComp, class EEComp> class hashdict  {
 private:
      Elem* HT;                  // 散列表
      int M;                     // 散列表大小
      int currcnt;               // 现有元素数目
      Elem EMPTY;                // 空槽
      int p(Key K, int i)        // 探查函数
      int h(Key K) const ;       // 散列函数
      Key getKey(Elem e);        // 获取元素 e 的关键码值
 public:
      hashdict(int sz, Elem e)  {  // 构造函数, e 用来定义空槽
            M = sz;                EMPTY = e;
            currcnt = 0;           HT = new Elem[sz];
            for (int i=0; i<M; i++) HT[i] = EMPTY;
      }
      ~hashdict() { delete []HT; }
      bool hashSearch(const Key&, Elem&) const;
      bool hashInsert(const Elem&);
      Elem hashDelete(const Key& K);
      int size() { return currcnt; }  // 散列表中现有元素数
};
```

墓碑就用常量 TOMB 表示。插入函数的原型如下：

```
template <class Key, class Elem, class KEComp, class EEComp>
bool hashdict<Key, Elem, KEComp, EEComp>::hashInsert(const Elem &e)
```

3. 实验证明，Shell 排序的增量序列按下述条件

$$h_1 = 1$$

$$h_{i+1} = 3h_i + 1$$

直到 $h_t$ (使得 $h_{t+2} \geq n$)选择时排序的效率较好。例如，对于 n =10000 时的增量序列为 1，4，13，40，121，364，1093，3280。

请给出按此增量序列实现的 shell 排序算法的代码。