上 海 交 通 大 学 试 卷（A 卷）

（ 2013 至 2014 学年 第 一 学期 ）

班级号_____ 学号_____ 姓名 _____

课程名称 _____《数据结构（A 类）》_____ 成绩 _____

一、单项选择题（每格 1.5 分，共 22.5 分）

1. 当采用分块查找时，数据的组织方式为 _____。

   A. 数据分成若干块，每块内数据有序

   B. 数据分成若干块，每块内数据不必有序，但块间必须有序，每块内最大（或最小）的数据组成索引块

   C. 数据分成若干块，每块内数据有序，但块间必须有序，每块内最大（或最小）的数据组成索引块

   D. 数据分成若干块，每块（除最后一块外）中数据个数需相同

2. 设给定权值总数有 n 个，其哈夫曼树的结点总数为 _____。

   A. 不确定　　B. 2n　　C. 2n+1　　D. 2n-1

3. 一个 n 个顶点的连通无向图，其边的个数至少为 _____。

   A. n-1　　B. n　　C. n+1　　D. nlogn

4. 一棵深度为 10 的完全二叉树，从根结点开始，对所有结点按照层次依次编号：0, 1, 2…，则编号为 18 的结点，其父结点编号为 _____。

   A. 10　　B. 9　　C. 8　　D. 5

5. 对有 14 个元素的有序表 A[1..14]作二分查找，查找元素 A[6]时的被比较元素依次为 _____。

   A. A[1], A[2], A[3], A[4]　　B. A[1], A[14], A[7], A[4]

   C. A[7], A[5], A[3], A[6]　　D. A[7], A[3], A[5], A[6]

6. 对顺序存储的线性表，设其长度为 20，容量足够大。在任何位置上插入或删除操作都是等概率的。插入一个元素时平均要移动表中的 _____ 个元素。

   A. 9.5　　B. 10　　C. 10.5　　D. 11

7. 当利用大小为 n 的数组顺序存储一个栈时，假定用 top == n 表示栈空，则向这个栈插入一个元素时，首先应执行 _____ 语句修改 top 指针。

   A. top++　　B. top--　　C. top = 0　　D. top

8. 设入栈顺序为 A，B，C，D，E，则出栈序列不可能是 _____。

   A. EDCBA　　B. ABCDE　　C. ADEBC　　D. ABDEC

我承诺，我将严格遵守考试纪律。

承诺人： _____

题号	一	二	三	四	五	六	七	总分
得分								
批阅人(流水阅) 卷教师签名处)								

9. 设有关键字初始序列{ Q, H, C, Y, P, A, M, S, R, D, F, X }，新序列{ F, H, C, D, P, A, M, Q, R, S, Y, X }是采用 _____ 方法对初始序列进行第一趟扫描的结果。

A. 直接插入排序　　B. 二路归并排序

C. 以第一元素为标准元素的快速排序　　D. 冒泡排序

10. 快速排序在最坏情况下的时间复杂度为 _____。

A. $O(\log_2 n)$　　B. $O(n\log_2 n)$　　C . $O(n)$　　D. $O(n^2)$

11. 从二叉搜索树中查找一个元素时，其时间复杂度大致为 _____。

A. $O(n)$　　B. $O(1)$　　C. $O(\log_2 n)$　　D. $O(n^2)$

12. 设某散列表的长度为 100，散列函数 H(k) = k % P，则 P 通常情况下最好选择 _____。

A. 99　　B. 97　　C. 91　　D. 93

13. 后缀表达式 8 2 3 + - 6 2 / - 的计算结果是 _____。

A. -1　　B. 1　　C. 0　　D. 2

14. 已知一个线性表中最常用的操作是删除第一个元素和在最后一个元素之后插入一个元素，则采用 _____ 存储方式最节省运算时间。

A. 双链表　　B. 仅有头指针的单循环链表

C. 单链表　　D. 仅有尾指针的单循环链表

15. 分别按照下列序列构造二叉查找树，与用其它三个序列所构造的结果不同的是 _____。

A. (105, 84, 97, 63, 122, 118, 131)

B. (105, 122, 118, 131, 84, 63, 97)

C. (105, 63, 84, 97, 122, 118, 131)

D. (105, 84, 63, 97, 122, 131, 118)

二、填充题（每格 1.5 分，共 25.5 分）

1. 表达式 23+((12*3 - 2)/4+34*5/7)+108/9 的后缀表达式是 _____。

2. 假设有一棵二叉树有 20 个叶子结点，有 30 个结点仅有一个孩子，则这棵二叉树的总结点个数应该为_____。

3. 在一棵 m 阶 B+树中，非叶子的树根结点的子树数目最少为 _____，非树根结点的分支结点的子树数目最少为 _____。

4. 设有一空栈，现有输入序列 s, k, t, r, e, p，经过 push, push, pop, push, pop, push, push, pop, pop, pop, push, pop 后，输出序列是 _____。

5. 假设用于通信的电文由 8 个字母组成，其频率分别为 0.07，0.19, 0.02, 0.06, 0.32, 0.03, 0.21, 0.10，为这 8 个字母设计哈夫曼编码，其中编码长度最大的字母的编码是 _____ 位。

6. 若以一个大小为 6 的数组来实现循环队列，且当前的 rear 和 front 分别为 0 和 3，当执行两次出队操作，两次入队操作，再执行一次出队操作后，front 为 _____，rear 为 _____。

7. 一棵二叉树的前序、中序和后序遍历的序列如下，其中有些部分未标出，请将各遍历序列补齐：
   前序遍历序列（3 分）：___, ___, C, D, E, ___, G, H, I, ___, K
   中序遍历序列（3 分）：C, B, ___, ___, F, A, ___, J , K , I , G
   后序遍历序列（3 分）：___, E, F, D, B, ___, J, I, H, ___, A

8. 设一组初始记录关键字序列为(49, 38, 65, 97, 76, 13, 27, 50)，则以 d = 4 为增量的一趟希尔排序结束后的结果为 _____。

9. 设有向图 G 的顶点集合为 V = { 1, 2, 3, 4, 5 }，边的集合为{ <1, 2>, <3, 2>, <1, 3>, <2, 4>, <3, 5>, <4, 5> }，那么它的一种拓扑排序为 _____。

10. 假设有 n 个关键字，它们具有相同的 Hash 函数值，用线性探测方法解决冲突，把这 n 个关键字存入散列表中，至少需要做 _____ 次探测操作。

三、简答题（10 分）

1. 设有向图有 6 个顶点，边的输入序列为：<1, 2>, <1, 3>, <3, 2>, <3, 0>, <4, 5>, <5, 3>, <0, 1>。求该图的邻接表，强连通分量的个数。

2. 已知 T 是一棵平衡二叉树，给定关键词 K，如果在 T 中查找 K 失败，且查找路径上的任一结点的平衡因子皆为零，试回答用平衡二叉树插入算法在 T 中插入关键词为 K 的新结点后，树 T 的高度是否一定增加？为什么？

四、分析题（10 分）

1. 请叙述下列程序 function 的功能。

```
Typedef struct {
    int vertex[m];
    int edge[m][m];
 } gadjmatrix;

Typedef struct node1 {
    int info;
    int adjvertex;
    struct node1 *nextarc;
} glinklistnode;

Typedef struct node2 {
    int vertexinfo;
    glinklistnode *firstarc;
} glinkheadnode;

void function ( gadjmatrix g1[ ], glinkheadnode g2[ ] ){
    int i, j; glinklistnode *p;
    for ( i = 0; i <= n-1; i++ ) g2[i].firstarc = 0;
    for ( i = 0; i <= n-1; i++ )
        for ( j = 0; j <= n-1; j++ )
            if ( g1.edge[i][j] == 1 ) {
                p = ( glinklistnode * ) malloc ( sizeof ( glinklistnode ) );
                p->adjvertex = j;
                p->nextarc = g[i].firstarc;
                g[i].firstarc = p;
                p = ( glinklistnode * ) malloc ( sizeof ( glinklistnode ) );
                p->adjvertex = i;
                p->nextarc = g[j].firstarc;
                g[j].firstarc = p;
            }
}
```

2. 请分析下列Kruscal算法的时间复杂度。

```
struct edge {
    int beg, end;
    TypeOfEdge w;
    bool operator < ( const edge &rp ) const { return w < rp.w; }
};
template < class TypeOfVer, class TypeOfEdge >
void adjListGraph < TypeOfVer, TypeOfEdge > :: kruskal ( ) const {
    int edgesAccepted = 0, u, v;
    edgeNode *p;
    edge e;
    DisjointSet ds( Vers );
    priorityQueue < edge > pq;

    for ( int i = 0; i< Vers; ++i ) {
    for ( p = verList[i].head; p != NULL; p = p->next )
        if ( i < p->end ) {
            e.beg = i; e.end = p->end; e.w = p->weight;
            pq.enQueue ( e );
        }
    }
    while ( edgesAccepted < Vers - 1 ) {
        e = pq.deQueue ( );
        u = ds.Find ( e.beg );
        v = ds.Find ( e.end );
        if ( u != v ) {
            edgesAccepted++;
            ds.Union( u, v );
            cout << '(' << verList[e.beg].ver << ',' << verList[e.end].ver << ')\t';
        }
    }
}
```

五、设计题（12 分）

1. 试设计一个求结点 x 在二叉树中的双亲结点算法（伪代码）。

2. 试设计一个算法（伪代码），求以二叉链表表示的二叉树中所有叶子结点数。

六、程序填充题（每格 2 分，共 20 分）

1. 下列算法实现在顺序散列表中查找值为 x 的关键字，请在下列划线处填上正确的语句：

```
template < class Type >
class closeHashTable : public hashTable < Type > {
  private :
    struct node {
        Type data;
        int state;                          // 0—empty, 1—active, 2—deleted
        node ( ) { state = 0; }
    };
    node * array;
    int size;
  procted :
    int ( *key ) ( const Type &x );
  public :
    closeHashTable ( int length = 101, int ( *f ) ( const Type &m ) = defaultKey );
    ~closeHashTable ( ) { delete [ ]array; }
    bool find ( const Type &x ) const;
};
template < class Type >
bool closeHashTable < Type > :: find ( const Type &x ) const {
    int initPos, pos;
    initPos = pos = key ( x ) % size;
    do {
        if ( array[pos].state == 0 ) _____;
        if ( array[pos].state == 1 && _____ )
        return true;
        pos = ( pos + 1 ) % size;
    } while _____;
    return false;
}
```

2. 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，请写出程序所缺的语句。[设：树结点值为字符，最多 100 个结点]

```
#include < iostream >
#include < cstring >
using namespace std;
const int MAX = 100;

typedef struct Node {
    char data;
    struct Node *left, *right;
} BinNode;

BinNode *buildBinTree ( char *ppos, char *ipos, int n ) {
```

```
    BinNode *ptr = new BinNode;
    char *pos;
    int k;
    if ( n <= 0 ) return NULL;
    ptr->data = _____;
    for ( pos = ipos; _____; pos++ ) if ( *pos == *ppos ) break;
    k = _____;
    ptr->left = buildBinTree ( ppos + 1, ipos,k );
    ptr->right = buildBinTree ( _____ );
    return ptr;
}

void postOrder ( BinNode *ptr ) {
    if ( ptr == NULL ) return;
    postOrder ( ptr->left );
    postOrder ( ptr->right );
    cout << ptr->data;
}

int main( ) {
    BinNode  *root;
    char preSeq[MAX], inSeq[MAX];
    cout << 'Please input the preorder sequence:';
    cin >> preSeq;
    cout << 'Please input the inorder sequence:';
    cin >> inSeq;

    root = buildBinTree ( preSeq,inSeq,strlen ( preSeq ) );
    postOrder ( root );

    return 0;
}
```

3. 汉诺塔的实现，也可以用非递归方式完成。请将以下程序补充完整：

```
// Operation choices: DOMOVE will move a disk
// DOTOH corresponds to a recursive call
enum TOHop { DOMOVE, DOTOH };

class TOHobj {                          // An operation object
  public :
    TOHop op;                           // This operation type
    int num;                            // How many disks
    Pole start, goal, tmp;              // Define pole order
    // DOTOH operation constructor
    TOHobj ( int n, Pole s, Pole g, Pole t ) {
        op = DOTOH; num = n;
```

```
        start = s; goal = g; tmp = t;
    }
    // DOMOVE operation constructor
    TOHobj ( Pole s, Pole g ) {
        op = DOMOVE;
        start = s; goal = g;
    }
};
void TOH ( int n, Pole start, Pole goal, Pole tmp, Stack < TOHobj* > &S) {
    S.push ( new TOHobj ( n, start, goal, tmp ) );      // Initial
    TOHobj *t;
    while ( S.length ( ) > 0 ) {            // Grab next task
        t = S.pop( );
        if ( t->op == DOMOVE )          // Do a move
        move ( t->start, t->goal );
        else if ( t->num > 0 ) {        // Store 3 recursive statements
            int num = t->num;
            Pole tmp = t->tmp; Pole goal = t->goal;
            Pole start = t->start;
            _____________________________________;
            _____________________________________;
            _____________________________________;
        }
        delete t;                       // Must delete the TOHobj we made
    }
}
```

七、附加题（10 分）

设计一个函数，在一个规模为 N 的无序数组中，找出第 K 个大的元素。

1） 函数一：要求时间复杂度为 O（ N + K * logN ）

2） 函数二：要求时间复杂度为 O（ N * logK ）