

# 湖南大学课程考试试卷

课程名称: 数据结构与算法; 试卷编号: 7; 考试时间: 120 分钟

题号	一	二	三	四	五	六	七	八	九	十	总分
应得分											100
实得分											评分:
评卷人											

(请将所有题目的答案写在答题纸上)

一、选择题。(每题 2 分, 共 20 分)

1. 设栈 S 和队列 Q 的初始状态为空, 元素 e1, e2, e3, e4, e5, e6 依次通过栈 S, 一个元素出栈后即进入队列 Q, 若 6 个元素出队的顺序是 e2, e4, e3, e6, e5, e1, 则栈 S 的容量至少应是 (B)。

(A) 2 (B) 3 (C) 4 (D) 6

2. 采用顺序查找方法查找长度为 n 的线性表, 平均查找长度为 (C)。

(A) n (B) n/2 (C) (n+1)/2 (D) (n-1)/2

3. 队列操作的原则是 (D)。

(A) 只能进行删除 (B) 后进先出 (C) 只能进行插入 (D) 先进先出

4. 以下哪一个不是队列的基本运算? (B)

(A) 从队尾插入一个新元素 (B) 从队列中删除第 i 个元素  
(C) 判断一个队列是否为空 (D) 读取队头元素的值

5. 任何一棵二叉树的叶结点在先序、中序和后序遍历序列中的相对次序 (A)。

(A) 不发生改变 (B) 发生改变 (C) 不能确定 (D) 以上都不对

6. 若用数组 S[n] 作为两个栈 S1 和 S2 的共用存储结构, 对任何一个栈, 只有当 S[n] 全满时才能作入栈操作, 为这两个栈分配空间的最佳方案是 (C)。

(A) S1 的栈底位置为 0, S2 的栈底位置为 n  
(B) S1 的栈底位置为 -1, S2 的栈底位置为 n/2  
(C) S1 的栈底位置为 0, S2 的栈底位置为 n-1  
(D) S1 的栈底位置为 0, S2 的栈底位置为 n/2

7. 对待排序数据的初始状态不作任何要求的排序方法有 (A)。

(A) 插入和快速排序 (B) 插入和归并排序  
(C) 归并和快速排序 (D) 归并和选择排序

8. 在下列排序算法中, (D) 算法可能会出现下面情况: 在最后一趟开始之前, 所有元素都不在其最终的位置上。

(A) 堆排序 (B) 冒泡排序 (C) 快速排序 (D) 插入排序

9. 采用邻接表存储的图的广度优先算法类似于二叉树的 (D)。

(A) 先序遍历 (B) 中序遍历 (C) 后序遍历 (D) 层次遍历

10. 具有 6 个顶点的无向图至少应有 (B) 条边才能保证图的连通性。

(A) 4 (B) 5 (C) 6 (D) 7

二、填空题。(每题 2 分, 共 20 分)

1. 一个算法的时间复杂度为  $(n^3 + n^2 \log_2 n + 14n) n^2$ , 其数量级表示为  $O(n^6)$ 。

2. 将两个各有 n 个元素的有序表归并成一个有序表, 其最少的比较次数是 n。

3. 设二叉树中结点的两个指针域分别为 lchild 和 rchild, 则判断指针变量 p 所指向的结点为叶子结点的条件是 p->lchild == NULL p->rchild == NULL。

4. 设一棵二叉树的前序序列为 ABC, 则有 5 种不同的二叉树可以得到这种序列。

5. 为了给 n 个字母编码而建立起来的 Huffman 树一共有 2n-1 个结点。

6. n 个顶点的连通图用相邻矩阵表示时, 该矩阵至少有 n-1 个非零元素。

7. 对于一个具有 n 个顶点和 e 条边的无向图, 在其对应的邻接表中, 所含边结点有 2e 个。

8. 冒泡排序在最好情况下的元素交换次数为 0。

9. 对于线性表 (78, 4, 56, 30, 65) 进行散列存储时, 若选用 H(K) = K mod 5 作为散列函数, 散列地址为 4 的有 1 个。

10. 向一棵 B 树插入元素的过程中, 若最终引起树根结点的分裂, 则新树比旧树的高度 增加 1。

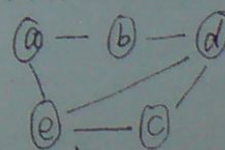
三、应用题。(每题 6 分, 共 30 分)

1. 设某棵二叉树的中序遍历序列为 DBEAC, 前序遍历序列为 ABDEC, 则该二叉树的后序遍历序列为 DEBCEA。

2. 假设有字符 A, B, C, D, E, F 的使用频率分别是 0.07, 0.09, 0.12, 0.15, 0.18, 0.21, 写出 A, B, C, D, E, F 的 Huffman (哈夫曼) 编码。

3. 已知一个无向图的顶点集为 {a, b, c, d, e}, 其邻接矩阵如下所示。

a	0	1	0	0	1
b	1	0	0	1	0
c	0	0	0	1	1
d	0	1	1	0	1
e	1	0	1	1	0



(1) 画出该图的图形;

(2) 根据邻接矩阵从顶点 a 出发进行深度优先遍历和广度优先遍历, 写出遍历序列。

深度优先遍历序列: a b d c e

广度优先遍历序列: a b c d e



已知一个图的原点集  $V$  和边集  $E$  分别为:

$V = \{1, 2, 3, 4, 5, 6, 7\}$ ;

$E = \{(1, 2)3, (1, 3)5, (1, 4)8, (2, 5)10, (2, 3)6, (3, 4)15,$

$(3, 5)12, (3, 6)9, (4, 6)4, (4, 7)20, (5, 6)18, (6, 7)25\}$ ;

用克鲁斯卡尔算法得到最小生成树, 试写出在最小生成树中依次得到的各条边。

设散列函数  $H(K) = K \bmod 7$ , 闭散列表的地址空间为  $0, \dots, 6$ , 开始时散列表为空, 用线性探查法解决冲突, 调出依次插入键值  $\{23, 14, 9, 6, 30, 12, 18\}$  后的散列表, 并作适当说明。

$h(K)$	2	0	2	6	2	5	4
	0	1	2	3	4	5	6
	14	18	23	9	30	12	6
(1)	(5)	(1)	(2)	(3)	(1)	(1)	

四. 算法设计。(每题 10 分, 共 30 分)

1. 阅读程序, 说明下列递归程序的功能

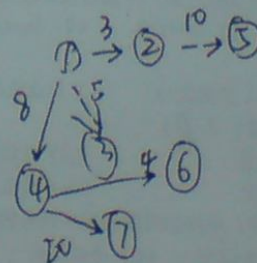
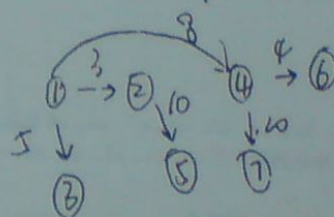
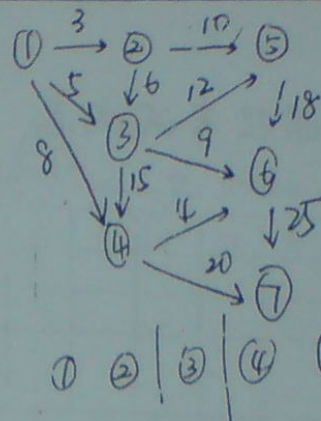
```
void unknown(BinTreeNode *T, int i) {
    // 指针 T 是完全二叉树的根指针。
    if (T == NULL) {
        cout << "i -> data << ", " << i << endl;
        unknown(T->leftChild, i+1);
        unknown(T->rightChild, i+1);
    }
}
```

主程序调用方式 `unknown(Root, 0);`

2. 二叉搜索树的查找——非递归算法如下, 请填写完整。

```
bool Find(BinTreeNode *BST, ElemType & item)
{
    while (BST != NULL) {
        if (item == _____) {
            item = BST->data; // 查找成功
            return true;
        }
        else if (item < BST->data)
            BST = BST->_____;
        else BST = BST->_____;
    }
    // end while
    return _____; // 查找失败
}
```

设计算法, 判断单链表元素是否递增。



$(4, 6) 4$   
 $(1, 2) 3$   $(1, 3) 5$   
 $(1, 4) 8$   $(2, 5) 10$   $(4, 7) 20$



# 湖南大学课程考试试卷

课程名称: 数据结构与算法 ; 试卷编号: C ; 考试时间: 120 分钟

题号	一	二	三	四	五	六	七	八	九	十	总分
应得分											100
实得分											评分:
评卷人											

(请将所有题目的答案写在答题纸上)

## 一、选择题 (每小题 2 分, 共 20 分)

1. 若某线性表最常用的操作是存取任一指定序号的元素和在最后进行插入和删除运算, 则采用 (C) 存储方式最节省时间。

(A) 顺序表 (B) 双链表 (C) 带头结点的双循环链表 (D) 单循环链表

2. 队列操作的原则是 (D)。

(A) 只能进行删除 (B) 后进先出 (C) 只能进行插入 (D) 先进先出

3. 某二叉树的先序序列和后序序列正好相反, 则该二叉树一定是 (B) 的二叉树。

(A) 空或只有一个结点 (B) 高度等于其结点数  
(C) 任一结点无左孩子 (D) 任一结点无右孩子

4. 在有  $n$  个结点的二叉链表中, 值为非空的链域个数为 (A)。

(A)  $n-1$  (B)  $2n-1$  (C)  $n+1$  (D)  $2n+1$

5. 对二叉树从 1 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, 同一个结点的左、右孩子中, 其左孩子编号小于右孩子编号, 则可采用 (D) 次序的遍历实现编号。

(A) 先序 (B) 中序 (C) 后序 (D) 从根开始的层次遍历

6. 若线性表中采用二分查找法查找元素, 该线性表应该 (C)。

(A) 元素按值有序 (B) 采用顺序存储结构

(C) 元素按值有序, 且采用顺序存储结构 (D) 元素按值有序, 且采用链式存储结构

7. 对待排序数据的初始状态不作任何要求的排序方法有 (A)。

(A) 插入和快速排序 (B) 插入和归并排序

(C) 归并和快速排序 (D) 归并和选择排序

8. 已知数据表  $A$  中每个元素距其最终位置不远, 则采用 (B) 排序算法最节省时间。

(A) 堆排序 (B) 插入排序 (C) 快速排序 (D) 选择排序

9. 以下哪一个不是队列的基本运算? (B)

(A) 从队尾插入一个新元素 (B) 从队列中删除第  $i$  个元素  
(C) 判断一个队列是否为空 (D) 读取队头元素的值

10. 具有  $n$  个顶点的有向图最多有 (B) 条边,  $A_n^2$   
(A)  $n(n-1)/2$  (B)  $n(n-1)$  (C)  $n(n+1)$  (D)  $n^2$

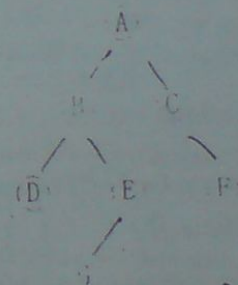
## 二、填空题 (每题 2 分, 共 20 分)

1. 下面给出了一段程序, 该程序的时间复杂度为  $O(n^2)$ 。

```
int sum (int n) /*n 为一个正整数*/
{
    int sum=0, i, j;
    for (i=1; i<=n; i++)
    {
        p=i;
        for (j=1; j<=i; j++) p*=j;
        sum+=p;
    }
    return (sum);
}
```

2. 向一个长度为  $n$  的顺序表的第  $i$  个元素 ( $1 \leq i \leq n+1$ ) 之前插入一个元素时, 需移动  $n-i+1$  个元素。

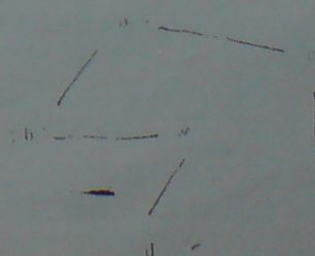
3. 对于下图给出的二叉树, 相应的顺序表示结点表为 ( " / " 代表空指针 NULL )  $ABD/EG/HC/F/$



4. 为了给  $n$  个字母编码而建立起来的 Huffman 树一共有  $2n-1$  个结点。

5. 将两个各有  $n$  个元素的有序表归并成一个有序表, 其最少的比较次数是  $n$ 。

6. 对于下图, 若从顶点  $a$  出发利用广度优先搜索进行遍历, 则得到的搜索序列为  $abcefd$ 。



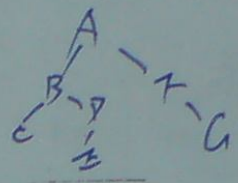


7. 冒泡排序在最好情况下的元素交换次数为 0。
8. 设一个连通图  $G$  中有  $n$  个顶点  $e$  条边，则其最小生成树上有  $n-1$  条边。
9. 对于线性表  $(78, 4, 56, 30, 65)$  进行散列存储时，若选用  $H(K) = K \% 5$  作为散列函数，散列地址为 4 的有 1 个。
10. 对一棵 B 树进行删除元素的过程中，若最终引起树根结点的合并时，会使新树的高度 比原树的高度大 1。

三、应用题。（每题 6 分，共 30 分）

1. 已知一棵二叉树的先序序列是 ABCDEFG，中序序列为 CBEDAFG，请构造出该二叉树。
2. 有一组关键字序列  $(38, 19, 65, 13, 49, 41, 1, 73)$ ，采用冒泡排序方法由小到大进行排序，请写出每趟排序的结果。
3. 设散列函数  $H(K) = K \bmod 7$ ，闭散列表的地址空间为  $0, \dots, 6$ ，开始时散列表为空，用线性探查法解决冲突，画出依次插入键值 23, 14, 9, 6, 30, 12, 18 后的散列表，并作说明。
4. 设图  $G = \langle V, E \rangle$ ， $V = \{1, 2, 3, 4, 5, 6\}$ ， $E = \{ \langle 1, 2 \rangle, \langle 1, 3 \rangle, \langle 2, 5 \rangle, \langle 3, 6 \rangle, \langle 6, 5 \rangle, \langle 5, 4 \rangle, \langle 6, 4 \rangle \}$ ，画出该图，并写出所有的拓扑序列。
5. 已知一个图的顶点集  $V$  和边集  $E$  分别为：

$V = \{1, 2, 3, 4, 5, 6, 7\}$ ;  
 $E = \{ (1, 2) 1, (1, 3) 5, (1, 4) 8, (2, 5) 10, (2, 3) 6, (3, 4) 15, (3, 5) 12, (3, 6) 9, (4, 5) 4, (4, 7) 20, (5, 6) 18, (6, 7) 25 \}$ ;  
 用 Prim 算法得到最小生成树，试写出在最小生成树中依次得到的各条边。



四、算法设计。（每题 16 分，共 30 分）

1. 统计出单链表  $L$  中结点的值等于给定值  $x$  的结点数。
2. 将下面的二分查找算法填写完整。

```

int Binsch(ElemType A[], int low, int high, KeyType K)
{
    if (low <= high) {
        int mid = (low + high) / 2;
        if (K == A[mid].key) return mid; // 查找成功，返回元素的标
        else if (K < A[mid].key)
            return Binsch(A, low, mid - 1, K); // 在左子表上继续查找
        else return Binsch(A, mid + 1, high, K); // 在右子表上继续查找
    }
    else return -1; // 查找失败，返回-1
}
  
```

```

int CountX(LNode *L, ElemType x)
{
    int i = 0; LNode *p = L;
    while (p != NULL)
    {
        if (p->data == x) i++;
        p = p->next;
    }
    return i;
}
  
```

设计在链式存储结构中交换二叉树中所有结点左右子树的算法。

```

void swapbitree(bitree *bt)
{
    bitree *p;
    if (bt == 0) return;
    swapbitree(bt->lchild);
    swapbitree(bt->rchild);
}
  
```