《数据结构》习题集

第一章 序论

思考题：

1.1 简述下列术 语：数据、数据元素、数据对象、数据结构、存储结构、数据类型、抽象数据类型

作业题：

1.2 设有数据结构（D，R），其中

D={d1, d2, d3, d4 }

R={r1, r2}

r1={ <d1, d2>, <d2, d3>, <d3, d4>, <d1, d4>, <d4, d2>, <d4, d1> }

r2={ (d1, d2), (d1, d3), (d1, d4), (d2, d4), (d2, d3) }

试绘出其逻辑结构示意图。

1.3 设 n 是正整数。试写出下列程序段中用记号“△”标注的语句的频度：

```
(1)    i=1; k=0;
       while(i<=n-1) {
       △  k+=10*i;
          i++;
       }

(2)    i=1; k=0;
       do {
       △  k+=10*i;
          i++;
       }while(i<=n-1)

(3)    i=1; k=0;
       do {
       △  k+ = 10*i; i++;
       }while(i==n);

(4)    i=1; j=0;
       while(i+j≤n) {
       △  if(i<j) i++; else j++;
       }
```

(5)
```
x=n; y=0; //n 是不小于 1 的常数
while(x>=(y+1)*(y+1)){
△  y++;
}
```

(6)
```
x=91; y=100;
while ( y>0 )
△  if(x>100) { x-=10; y--; }
    else  x++ ;
}
```

(7)
```
for( i=0; i<n; i++)
    for( j=i; j<n; j++)
        for( k=j; k<n; k++)
        △  x+=2;
```

第二章 线性表

思考题：

2.1 描述以下三个概念的区别：头指针、头结点、首元结点。

2.2 描述以下几个概念：顺序存储结构、链式存储结构、顺序表、有序表。

作业题：

2.3 已知顺序表 La 中数据元素按非递减有序排列。试写一个算法，将元素 x 插到 La 的合适位置上，保持该表的有序性。

2.4 已知单链表 La 中数据元素按非递减有序排列。按两种不同情况，分别写出算法，将元素 x 插到 La 的合适位置上，保持该表的有序性：（1）La 带头结点；（2）La 不带头结点。

2.5 试写一个算法，实现顺序表的就地逆置，即在原表的存储空间将线性表（$a_1, a_2, \ldots, a_{n-1}, a_n$）逆置为($a_n, a_{n-1}, \ldots, a_2, a_1$)

2.6 试写一个算法，对带头结点的单链表实现就地逆置。

2.7 已知线性表 L 采用顺序存储结构存放。对两种不同情况分别写出算法，删除 L 中值相同的多余元素，使得 L 中没有重复元素：(1)L 中数据元素无序排列；(2)L

中数据元素非递减有序排列。

2.8 将 2.7 题中 L 的存储结构改为单链表，写出相应的实现算法。

2.9 设有两个非递减有序的单链表 A 和 B。请写出算法，将 A 和 B“就地”归并成一个按元素值非递增有序的单链表 C。

2.10 设有一个长度大于 1 的单向循环链表，表中既无头结点，也无头指针，s 为指向表中某个结点的指针，如图 2-1 所示。试编写一个算法，删除链表中指针 s 所指结点的直接前驱。

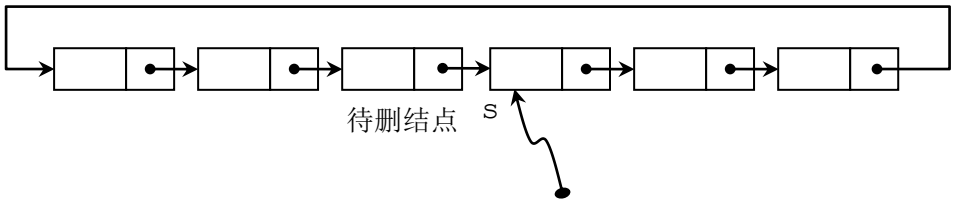

图 2-1

2.11 已知线性表用带头结点的单链表表示，表中结点由三类字符组成：字母、数字和其他字符。试编写算法，将该线性链表分割成三个循环单链表，每个循环单链表中均只含有一类字符。

2.12 已知线性表用顺序存储结构表示，表中数据元素为 n 个正整数。试写一算法，分离该表中的奇数和偶数，使得所有奇数集中放在左侧，偶数集中放在右侧。要求：(1)不借助辅助数组;(2)时间复杂度为 O(n)。

2.13 设以带头结点的双向循环链表表示的线性表 $L=(a_1, a_2, a_3, \ldots, a_n)$。试写一时间复杂度为 O(n)的算法，将 L 改造为 $L=(a_1, a_3, \ldots, a_n, \ldots, a_4, a_2)$。

第三章 栈和队列

思考题：

3.1 简述栈和线性表的差别。

3.2 如果进栈序列为 A、B、C、D，写出所有可能的出栈序列。

3.3 简述栈和队列的相同点和差异。

3.4 已知栈 S 中存放了 8 个数据元素，自栈底至栈顶依次为(1,2,3,4,5,6,7,8)。

（1）写出在执行了函数调用 algo1(S)后，S 中的元素序列。

（2）在（1）的基础上，又执行了函数调用 algo2(S,5)，写出此时 S 中的元素序列。

```
void algo1(Stack &S){
    int a[10], i, n=0;
    while(!StackEmpty(S)){ n++; Pop(S, a[n]); }
    for(i=1; i<=n; i++) Push(S, a[i]);
}


void algo2(Stack &S, int e){
    Stack T;
    int d;
    InitStack(T);
    while(!EmptyStack(S)){
       Pop(S,d);
       if(d!=e) Push(T,d);
    }
    while(!StackEmpty(T)){
       Pop(T,d);
       Push(S,d);
    }
}
```

3.5 已知队列 Q 中自队头至队尾依次存放着(1,2,3,4,5,6,7,8)。写出在执行了函数调用 algo3(Q)后，Q 中的元素序列。

```
void algo3(Queue &Q){
    Stack S;
    int d;
    InitStack(S);
    while(!QueueEmpty(Q)){
```

```
        DeQueue(Q,d); Push(S,d);
    }
    while(!StackEmpty(S)){
        Pop(S,d); EnQueue(Q,d);
    }
}
```

作业题：

3.6 试写一个算法，识别依次读入的一个以@为结束符的字符序列是否为形如“序列$_1$&序列$_2$”模式的字符序列。其中，序列$_1$和序列$_2$中都不包含字符‘&’，且序列$_2$是序列$_1$的逆序。例如，“a+b&b+a”是属于该模式的字符序列，而“1＋3&3－1”则不是。

3.7 假设一个算术表达式中可以包含三种符号：圆括号“(”和“)”、方括号“[”和“]”、花括号“{”和“}”，且这三种括号可按任意次序嵌套使用。编写判别给定表达式中所含的括号是否正确配对的算法(已知表达式已存入数据元素为字符的顺序表中)。

3.8 设表达式由单字母变量、双目运算符和圆括号组成(如:“(a*(b+c)-d)/e)”。试写一个算法，将一个书写正确的表达式转换为逆波兰式。

3.9 试用类 C 写一个算法，对逆波兰式求值。

3.10 假设以带头结点的单循环链表表示队列，只设一个尾指针指向队尾元素，不设头指针。试编写相应的队列初始化、入队和出队的算法。

3.11 假设将循环队列定义为：以 rear 和 length 分别指示队尾元素和队列长度。试给出此循环队列的队满条件，并写出相应的入队和出队算法(在出队算法中要传递回队头元素的值)。

3.12 试写一个算法：判别读入的一个以‘@’为结束符的字符序列是否是“回文”(所谓“回文”是指正读和反读都相同的字符序列，如“xxyzyxx”是回文，而“abcab”则不是回文)。

第五章 多维数组

5.1 已知多维数组 A[2][2][3][3]按行优先方式存储。试按存储位置的先后次序，列出所有数组元素 A[i][j][k][l]序列（为了简化表达，可以只列出形如“i,j,k,l”的序列，如元素 A[0][0][2][1]可表示为“0,0,2,1” ）。

5.2 假设有一个二维数组 A[0..5][0..7]，每个元素占 6 个字节，首元素 A[0][0]的地址为 1000，求：

(1)A 的体积；

(2)最后一个元素 A[5][7]的地址；

(3)按行主序方式存储时，A[2][4]的地址；

(4)按列主序方式存储时，A[2][4]的地址；

5.3 设有上三角矩阵 An×n，

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ & a_{22} & a_{23} & \cdots & a_{2n} \\ & & a_{33} & \cdots & a_{3n} \\ & C & & \cdots & \cdots \\ & & & & a_{nn} \end{bmatrix}$$

将其上三角的元素逐行存于数组 B[0..m-1]中（m 充分大），使得 B[k]=a_{ij}且 k＝$f_1(i)+f_2(j)+c$。试推导出函数 f_1、f_2和常数 c（要求 f_1和 f_2中不含常数项）。

5.4 设有一个准对角矩阵

$$\begin{bmatrix} a_{11} & a_{12} & & & & & & \\ a_{21} & a_{22} & & & & & & \\ & & a_{33} & a_{34} & & & & \\ & & a_{43} & a_{44} & & & & \\ & & & & \cdots & \cdots & & \\ & & & & \cdots & \cdots & & \\ & & & & & & a_{2m-1,2m-1} & a_{2m-1,2m} \\ & & & & & & a_{2m,2m-1} & a_{2m,2m} \end{bmatrix}$$

按以下方式存于一维数组 B[4m]中：

0	1	2	3	4	5	6		k		4m-2	4m-1
a_{11}	a_{12}	a_{21}	a_{22}	a_{33}	a_{34}	a_{43}	...	a_{ij}	...	$a_{2m-1,2m}$	$a_{2m,2m}$

写出由一对下标(i,j)求 k 的转换公式。

5.5 已知稀疏矩阵 $A_{4\times5}$ 如下：

$$A=\begin{bmatrix}0&1&0&0&5\\2&3&0&6&0\\0&0&0&0&0\\0&4&0&0&7\end{bmatrix}$$

(1)用三元组表作为存储结构，绘出相应的三元组表示意图；
(2)用十字链表作为存储结构，绘出相应的十字链表示意图。

5.6 设稀疏矩阵 A 和 B 均以三元组顺序表作为存储结构。试写出计算矩阵相加 C＝A＋B 的算法，其中，C 是另设的、存放结果的三元组表（提示：可用类似于两个有序顺序表归并的处理方法）。

5.7 试编写一个算法，实现以三元组的形式打印用十字链表表示的稀疏矩阵中所有非零元素及其下标。

5.8 试编写一个算法，实现以矩形阵列的形式打印用十字链表表示的稀疏矩阵。

第六章 树和二叉树

6.1 试分别绘出具有 3 个结点的树和 3 个结点的二叉树的所有不同形态。

6.2 设结点 X 是二叉树上一个度为 1 的结点，X 有几个子树？

6.3 描述满足下列条件的二叉树形态：

(1) 先序遍历序列与中序遍历序列相同；
(2) 后序遍历序列与中序遍历序列相同；
(3) 先序遍历序列与后序遍历序列相同；

6.4 一个深度为 H 的满 k 叉树有如下性质：第 H 层上所有结点都是叶子结点，其余各层上每个结点都有 k 棵非空子树。如果从 1 开始按自上而下、自左向右的次序对全部结点编号，问：

(1) 各层的结点数目是多少？
(2) 编号为 i 的结点的父结点(若存在)的编号是多少？
(3) 编号为 i 的结点的第 j 个孩子(若存在)的编号是多少？
(4) 编号为 i 的结点有右兄弟的条件是什么？其右兄弟的编号是多少？

6.5 已知一棵度为 k 的树中有 n1 个度为 1 的结点，n2 个度为 2 的结点，...，nk 个度为 k 的结点，问该树中有多少个叶子结点？

6.6 已知在一棵含有 n 个结点的树中，只有度为 k 的分支结点和度为 0 的叶子结点。试求该树含有的叶子结点的数目。

6.7 设 n 和 m 为二叉树中两个结点，用“1”、“0”、和“∅”（分别表示肯定，否定和不一定）填写下表：

已知 \ 问	先序遍历时 n 在 m 之前？	中序遍历时 n 在 m 之前？	后序遍历时 n 在 m 之前？
n 在 m 左方			
n 在 m 右方			
n 是 m 祖先			
n 是 m 子孙			

（注：如果离 n 和 m 的最近的共同祖先 X 存在，且 n 位于 X 的左子树中，m 位于 X 的右子树中，则称“n 在 m 的左方”或“m 在 n 的右方”。）

6.8 已知一棵树如图 6-1 所示，画出与该树对应的二叉树，并写出该树的先根遍历序列和后根遍历序列。

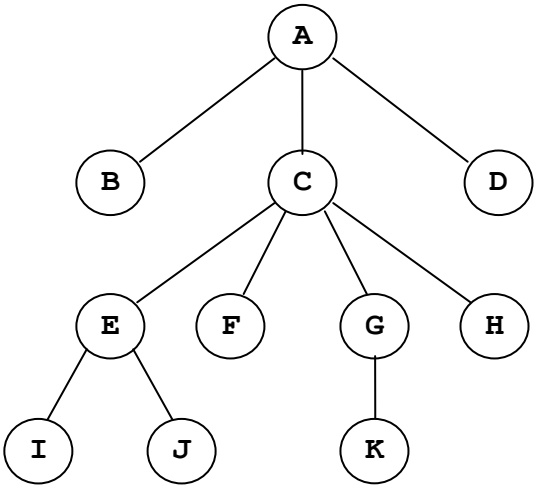

图 6-1

6.9 将如图 6-2 所示的森林转化为对应的二叉树。

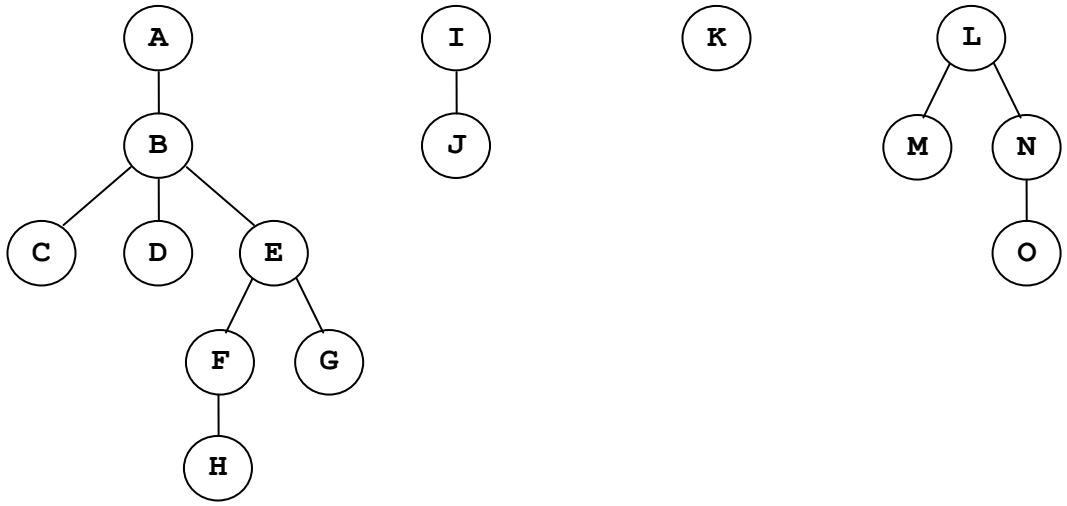

图 6-2

6.10 画出和下列二叉树（如图 6-3 所示）相应的森林。

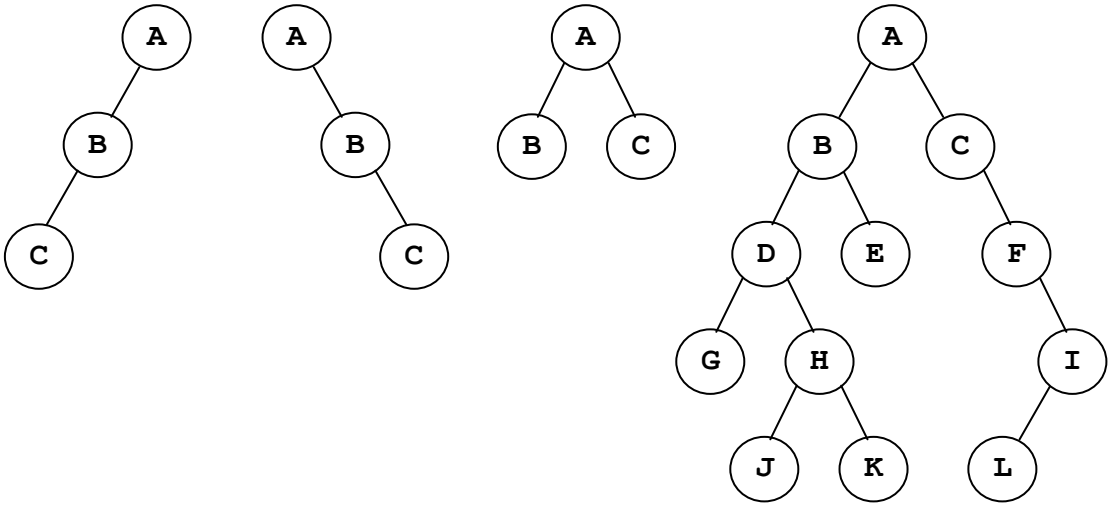

图 6-3

6.11 已知某二叉树的中序序列为 DCBGEAHFIJK，后序序列为 DCEGBFHKJIA。请画出该二叉树。

6.12 已知树 T 的先根遍历访问序列为 GFKDAIEBCHJ，后根遍历访问序列为 DIAEKFCJHBG。请画出树 T。

6.13 已知森林 F 的先根遍历访问序列为 ABCDEFGHIJKL，中根遍历访问序列为 CBEFDGAJIKLH。请画出这个森林 F。

6.14 假设某个电文由(a, b, c, d, e, f, g, h)8 个字母组成，每个字母在电文中出现的次数分别为(7, 19, 2, 6, 32, 3, 21, 10)，试解答下列问题：

(1) 画出出 huffman 树；

(2) 写出每个字母的 huffman 编码；

(3) 在对该电文进行最优二进制编码处理后，电文的二进制位数。

6.15 写出复制一棵二叉树的算法。

6.16 试编写算法，实现将二叉树所有结点的左右子树互换。

6.17 写出按层次遍历二叉树的算法。

6.18 写出判断给定二叉树是否为完全二叉树的算法。

6.19 写出判断两棵给定二叉树是否相似的算法。

(注：两棵二叉树 B1 和 B2 相似是指：B1 和 B2 皆空，或者皆不空且 B1 的左、右子树和 B2 的左、右子树分别相似。)

6.20 利用栈的基本操作，写出二叉树先序遍历的非递归算法。

6.21 写出统计树中叶子结点个数的算法，树用孩子兄弟链表表示。

6.22 写出计算树的深度的算法，树用孩子兄弟链表表示。

6.23 写出计算二叉树第 K 层结点数的算法。

6.24 写出计算二叉树宽度的算法。

第七章 图

7.1 已知有向图如图 7-1 所示，

请给出该图的

(1)邻接矩阵示意图

(2)邻接表示意图

(3)逆邻接表

(4)所有强连通分量

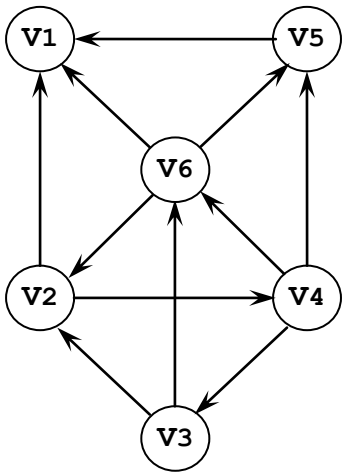

图 7-1

7.2 已知图 G 的邻接矩阵如图 7-2 所示。写出该图从顶点 1 出发的深度优先搜索序列和广度优先搜索序列，并画出相应的深度优先生成树和广度优先生成树。

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	1	0	1	0
2	0	0	1	0	0	0	1	0	0	0
3	0	0	0	1	0	0	0	1	0	0
4	0	0	0	0	1	0	0	0	1	0
5	0	0	0	0	0	1	0	0	0	1
6	1	1	0	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	1
8	1	0	0	1	0	0	0	0	1	0
9	0	0	0	0	1	0	1	0	0	1
10	1	0	0	0	0	1	0	0	0	0

图 7-2

7.3 无向带权图如图 7-3 所示，

(1)画出它的邻接矩阵，并按 Prim 算法求其最小生成树。

(2)画出它的邻接表，并按 Kruskal 算法求其最小生成树

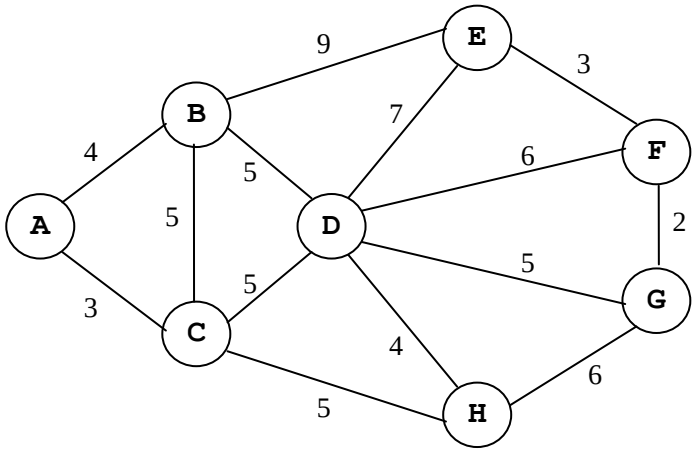

图 7-3

7.4 有向图如图 7-4 所示，试写出其所有可能的拓扑序列。

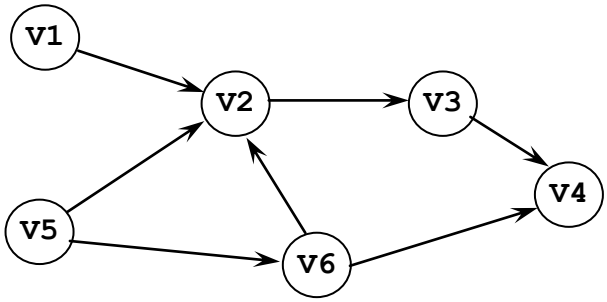

图 7-4

7.5 试利用 Dijkstra 算法求图 7-5 中顶点 A 到其他各顶点之间的最短路径。要求写出执行算法过程中，数组 D、P 和 S 各步的状态。

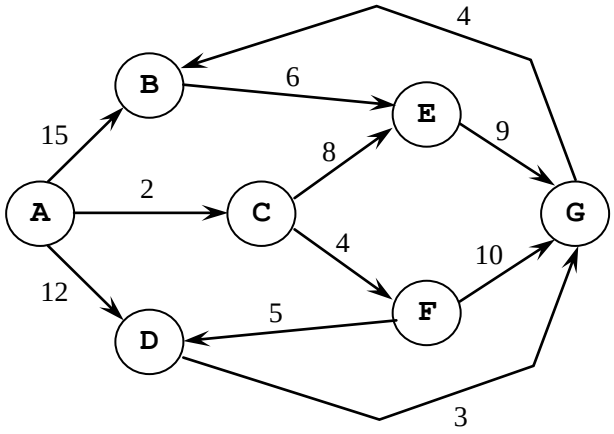

图 7-5

7.6 试在邻接矩阵存储结构上实现图的基本操作：InsertVex(G,v)，InsertArc(G,v,w)，DeleteVex(G,v)和 DeleteArc(G,v,w)。

7.7 试在邻接表存储结构上实现图的基本操作：InsertVex(G,v)，InsertArc(G,v,w)，DeleteVex(G,v)和 DeleteArc(G,v,w)。

7.8 设具有 n 个顶点的有向图用邻接表存储。试写出计算所有顶点入度的算法，可将每个顶点的入度值分别存入一维数组 int Indegree[n]中。

7.9 假设有向图以邻接表作为存储结构。试基于图的深度优先搜索策略写一算法，判断有向图中是否存在由顶点 Vi 至顶点 Vj(i!=j)的路径。

7.10 假设有向图以邻接表作为存储结构。试基于图的广度优先搜索策略写一算法，判断有向图中是否存在由顶点 Vi 至顶点 Vj(i!=j)的路径。

7.11 以邻接表作为存储结构，实现求单源最短路径的 Dijkstra 算法。

第九章 查找

9.1 若对大小均为 n 的有序顺序表和无序顺序表分别进行顺序查找，试在下列三种情况下分别讨论两者在等概率时平均查找长度是否相同？

(1)查找不成功，即表中没有关键字等于给的值 K 的记录；

(2)查找成功，且表中只有一个关键字等于给定值 K 的记录；

(3)查找成功，且表中有若干个关键字等于给定值 K 的记录，要求找出所有这些记录。

9.2 试分别写出在对有序线性表(a,b,c,d,e,f,g)中进行折半查找，查值等于 e、f 和 g 的元素时，先后与哪些元素进行了比较。

9.3 画出对长度为 17 的有序表进行折半查找的判定树，并分别求其等概率时查找成功和查找失败的 ASL。

9.4 已知如下所示长度为 12 的表：

(Jan, Feb, Mar, Apr, May, Jun, July, Aug, Sep, Oct, Nov, Dec)

表中，每个元素的查找概率分别为：

(0.1, 0.25, 0.05, 0.13, 0.01, 0.06, 0.11, 0.07, 0.02, 0.03, 0.1, 0.07)

(1)若对该表进行顺序查找，求查找成功的平均查找长度；

(2)画出从初态为空开始，依次插入结点，生成的二叉排序树；

(3)计算该二叉排序树查找成功的平均查找长度；

(4)将二叉排序树中的结点 Mar 删除，画出经过删除处理后的二叉排序树。

9.5 已知关键字序列{10,25,33,19,06,49,37,76,60}，哈希地址空间为 0～10，哈希函数为 H (Key)=Key % 11，求：

（1） 用开放定址线性探测法处理冲突，构造哈希表 HT1，分别计算在等概率情况下 HT1 查找成功和查找失败的 ASL；

（2） 用开放定址二次探测法处理冲突，构造哈希表 HT2，计算在等概率情况下 HT2 查找成功的 ASL；

（3） 用拉链法解决冲突，构造哈希表 HT3，计算 HT3 在等概率情况查找成功的 ASL。

9.6 写出折半查找的递归算法。

9.7 写出判别一棵二叉树是否为二叉排序树的算法，设二叉排序树中不存在关键字值相同的结点。

9.8 假设哈希表长为 m， 哈希函数为 H(x)，用链地址法解决冲突。编写输入一

组关键字，建造哈希表的算法。

第十章 内部排序

10.1 以关键字序列（5，1，6，0，9，2，8，3，7，4）为例，手工执行下列排序算法，写出每一趟排序结束时关键字序列状态

(1) 直接插入排序　　(2) 希尔排序（取增量为 5，3，1）

(3) 快速排序　　(4) 冒泡排序

(5) 归并排序　　(6) 堆排序

10.2 10.1 题中哪些排序方法是稳定的，哪些是不稳定的？并为每种不稳定的排序方法举一个不稳定的实例。

10.3 判别以下序列是否为堆（小顶堆或大顶堆），若不是，则吧它调整为堆

(1) (96, 86, 48, 73, 35, 39, 42, 57, 66, 21)；

(2) (12, 70, 33, 65, 24, 56, 48, 92, 86, 33)；

10.4 编写一个双向起泡的排序算法，即相邻两遍向相反方向起泡。

10.5 试以单链表为存储结构，实现简单选择排序算法。