

**Heilongjiang University of Science and Technology**

**专业基础训练1（数据结构）**

**课程设计指导书**

**计算机与信息工程学院**

**一、课程性质及教学目标**

1、本课程是软件工程专业本科生培养方案所规定的一门专业实践必修课。而本实践课程做为一门综合实践课程，要求学生利用现阶段所学的程序设计语言、数据结构等有关理论知识，结合实际问题设计相关算法及程序，以达到理论与实践相结合的目的。

2、本课程的教学目标是通过本综合实践课程的实践训练，使学生能够掌握综合运用现阶段所学的程序设计、数据结构等有关理知识来解决实际问题的方法；巩固和掌握程序设计、数据结构设计、算法设计的基本方法与技能；进一步加强学生的程序设计、编码、运行与调试能力和算法设计与分析的能力；培养学生分析问题、解决问题的能力。

**二、设计题目**

1.带命令的顺序表

顺序表的类型定义如下。

typedef struct SqList{

int \*base;

int length;

int listsize;

}SqList;

给定一个长度为n的顺序表L，顺序表L中存放了n个整数。

下面对顺序表L实施两种命令，命令的代号为A和B。

命令A的格式为：A X Y

它的含义是将顺序表L中值为X的元素移到值为Y的元素的左边

命令B的格式为：B X Y

它的含义是将顺序表L中值为X的元素移到值为Y的元素的右边

例如，假设当前顺序表L的长度为8，所含的8个元素如下所示。

L

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 10 | 95 | 23 | 41 | 6 | 7 | 9 |

|  |
| --- |
| length=8 |
| base |
| listsize=8 |

则执行命令A 10 41后，顺序表L如下图所示。

L1023

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 95 | 23 | 10 | 41 | 6 | 7 | 9 |

|  |
| --- |
| length=8 |
| base |
| listsize=8 |

再执行命令B 95 6后，顺序表L如下图所示。

L1023

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 23 | 10 | 41 | 6 | 95 | 7 | 9 |

|  |
| --- |
| length=8 |
| base |
| listsize=8 |

要求：

输入顺序表中的元素个数n和命令的条数m，然后向顺序表中输入n个整数，再输入m条命令，最后输出经过m条命令操作后得到的顺序表。

例如，若输入：

8 2

8 10 95 23 41 6 7 9

A 10 41

B 95 6

则输出：

8 23 10 41 6 95 7 9

2. Josephus环问题和反Josephus环问题

（1）带人名的Josephus环问题

Josephus环问题为：设编号为1，2，… n的n个人围坐一圈，约定从编号为k（1<=k<=n）的人开始从1报数，数到m 的那个人出列，它的下一位又重新从1开始报数，数到m的那个人又出列，依次类推，直到所有人出列为止。

要求：

输入人的个数n（n<=64），接下来每行输入一个人的名字（人名不超过15个字符），接下来输入k和m，k和m之间用逗号隔开。

最后按照人名输出出列的顺序，每行输出一个人名。

如下表给出的样例。

|  |  |
| --- | --- |
| 样例输入 | 样例输出 |
| 5 | Caobainan |
| Mazhongyi | Shenyongqiang |
| Shenyongqiang | Mazhongyi |
| Taozhengyi | Taozhengyi |
| Caobainan | Jiangdebing |
| Jiangdebing |  |
| 2,3 |  |

（2）反Josephus环问题

反Josephus环问题为：设编号为1，2，… n的n个人围坐一圈，约定从编号为k（1<=k<=n）的人开始从1报数，数到m 的那个人出列，它的下一位又重新从1开始报数，数到m的那个人又出列，依次类推，直到所有人出列为止，若想使最后出列的人为第x（1<=x<=n）个人，那么应从第几个人开始报数才行呢？输入n、m和x的值，输出满足条件的k值。

测试数据：当n=41，m=3，x=3时，**k=14**。

（3）Josephus环问题的变形

设有编号为1，2，…，k-1，k，k+1，…，2k-1，2k（1=<k<=7）的2k个人围坐一圈，其中，前k个人为好人，后k个人为坏人。约定从编号为1的人开始从1报数，数到m 的那个人就被处死，它的下一位又重新从1开始报数，数到m的那个人又被处死，依此类推，直至剩下k个人为止。若想使最后剩下的这k个人都是好人，求满足条件的最小的m值。输入k的值，输出满足条件的最小的m值。

测试数据1：若k=3，则**m=5。**

测试数据2：若k=4，则**m=30。**

（4）直线型报数问题

克林格是美国某野外集团军的上校军官，他负责处理军队的一些事务。克林格现在要从军队中选择X个幸运的人回国进行招兵宣传，而且他希望通过报数的方式来做选择。报数的方式是将本单位的所有成员排成一排，然后从一叠卡片的最上方取卡片，假设最上方的卡片上的数值为m，然后从队列中的第1个人开始从1报数，数到m的人离开队列，然后下一个人再从1开始报数，数到m的人再离开队列，当报数报到队列结束的时候，再从这叠卡片的最上方取下一张卡片，再从剩余队列中的第1个人开始根据新的卡片上的数值进行报数。队列中最后剩下的X个人可以回家。

克林格在选拔过程开始前叠好了一叠卡片，然而到了最后一分钟他才知道有多少人参与。请你编写程序，基于克林格的这叠卡片上各张卡片的数值和队列中人员的数量，告诉他队列中哪些位置的人可以回家。可以确定最多用20张卡片。

例如，队列中有10个人，编号从1到10，共有2个幸运位置，卡片的数值分别为3 5 4 3 2，则队列中的第1个人和第8个人可以回家。过程如下：

队列1 2 3 4 5 6 7 8 9 10，X=2，卡片的数值分别为3、5、4、3、2……则：

第1张卡片数值3：划掉3，6，9；剩下1，2，4，5，7，8，10

第2张卡片数值5：划掉7；剩下1，2，4，5，8，10

第3张卡片数值4：划掉5；剩下1，2，4，8，10

第4张卡片数值3：划掉4；剩下1，2，8，10

第5张卡片数值2：划掉2，10；剩下1，8

要求：

输入22个整数，第1个整数n（1=<n<=50）表示队列中的人的数量，第2个整数X（1=<X<=n）给出有多少个幸运的回家位置。后面的20个整数给出前20张卡片的数值，卡片的数值是从1到11的整数。

输出可以回家的这X个人的编号，并输出最后所用的卡片数量。

例如：

若输入：

10 2 3 5 4 3 2 9 6 10 10 6 2 6 7 3 4 7 4 5 3 2

则输出：

1 8

5

若输入：

47 6 11 2 7 3 4 8 5 10 7 8 3 7 4 2 3 9 10 2 5 3

则输出：

1 3 16 23 31 47

11

3．链表算法设计

（1）设循环单链表中结点的类型定义若下：

typedef struct LNode{

int data;

struct LNode \*next;

}LNode;

假设有一个不带头结点的循环单链表，其长度大于1，R为它的尾指针，指向循环单链表的最后一个结点。编写算法在链表中删除R结点的直接前驱结点。

如下图所示的循环单链表：

R

76

23

57

12

87

40

则删除R的前驱结点之后链表变为：

76

23

57

12

40

R

再如下图所示的循环单链表：

76

12

R

则删除R的前驱结点之后链表变为：

76

R

要求采用尾插法先创建一个循环单链表，然后输出该链表的各结点的数据域值，接下来编写算法删除R结点的前驱结点，最后输出删除结点后的链表的各结点的数据域值。

（2）设单链表中结点的类型定义若下：

typedef struct LNode{

char data;

struct LNode \*next;

}LNode;

已知一个带有头结点的单链表中的数据元素包含3类字符，即字母字符、数字字符（0到9）和其他字符，编写算法构造3个带头结点的单链表，使每个单链表中只包含同一类的字符，且利用原表中的结点空间作为这3个链表的结点空间，头结点可另辟空间。

例如，带头结点的单链表L如下：

8

A

$

a

#

9

5

c

&

^

L

则最后单链表L变成了以下三个链表：

8

9

5

^

L

A

a

c

^

L1

$

#

&

^

L2

要求先创建一个带头结点的单链表，然后输出该链表的各结点的数据域值，接下来编写算法将单链表L拆分成三个单链表，最后输出这三个单链表中各结点的数据域值。

4．一元多项式的表示及其运算（要求：包括相加、相减、相乘等运算）

符号多项式的操作，已经成为表处理的典型用例。在数学上，一个一元多项式Pn(x)可按升幂写成： Pn(x) = p0+ p1x+ p2x2+….+ pnxn 它由n+1个系数唯一确定，因此，在计算机里，它可用一个线性表P来表示：

P = (p0 ,p1 ,p2 ,… pn)每一项的指数i隐含在其系数pi的序号里。

假设Qm(x)是一元m次多项式，同样可用线性表Q来表示:Q = (q0 ,q1 ,q2 ,… qm)。  
 不失一般性，设m<n，则两个多项式相加的结果 Rn(x) = Pn(x)+Qm(x)可用线性表R表示：R = (p0+q0 , p1+q1 , p2 +q2 , … , pm +qm , pm+1 ,… pn)。显然，我们可以对P、Q和R采用顺序存储结构，使得多项式相加的算法定义十分简洁。至此，一元多项式的表示及相加问题似乎已经解决了。  
    然而在通常的应用中，多项式的次数可能很高且变化很大，使得顺序存储结构的最大长度很难决定。特别是在处理形如:S(x) = 1+3x10000+2x20000的多项式时，就要用一长度为20001的线性表来表示，表中仅有三个非零元素，这种对内存空间的浪费是应当避免的，但是如果只存储非零系数项则显然必须同时存储相应的指数。   
    一般情况下的一元n次多项式可写成:

Pn(x) = p1xe1 + p2xe2 + … + pmxem

    其中pi，是指数为ei的项的非零系数，且满足0 ≤ e1 < e2 < …< em = n，若用一个长度为m且每个元素有两个数据项(系数项和指数项)的线性表便可唯一确定多项式Pn(x)。

**((p1 ,e1) , (p2 ,e2) , … ,(pm,em))**   
    在最坏情况下，n+1(=m)个系数都不为零，则比只存储每项系数的方案要多存储一倍的数据。但是，对于S(x)类的多项式，这种表示将大大节省空间。

本题要求选用线性表的一种合适的存储结构来表示一个一元多项式，并在此结构上实现一元多项式的加法，减法和乘法操作。可以参考教材中P54页的2.4.2小节。

5.亲兄弟问题

给定一个长度为n的整数序列：a0，a1，…，an-1，序列中的任何一个元素ai（0=<i<=n-1）的亲兄弟元素ak定义为：ak是在ai的右边最靠近ai且不小于它的元素。既满足：

k=min{ j | aj>=ai},i<j<n

对于给定的元素序列，要求求出每个元素的亲兄弟元素所在的位置，如果一个元素没有亲兄弟元素，约定其亲兄弟位置为-1。

要求首先用穷举法编程解决该问题，然后再借助于“栈”来编程解决该问题，对于栈，可采用简化的栈结构。

首先输入一个整数n，然后输入n个整数，输出每个整数的亲兄弟元素的位置。

例如：

若输入：

6

8 5 4 7 1 2

则输出：

-1 3 3 -1 5 -1

若输入：

6

1 2 3 4 5 6

则输出：

1 2 3 4 5 -1

若输入：

6

6 5 4 3 2 1

则输出：

-1 -1 -1 -1 -1 -1

6.是栈还是队列

你知道栈和队列吗？它们都是重要的数据结构，栈是后进先出的顺序表，而队列则是先进先出的顺序表。

现在给出这样的问题：给出进入一种结构和离开该结构的一些整数的序列（假设在栈和队列中的数据元素都是整数），请你确定这种数据结构到底是栈还是队列。请注意：本题要求只有在该序列中的所有整数都进入结构后才能执行相应的弹出操作。

要求：

首先输入一个整数n，它表示序列中的元素个数，接下来输入n个整数，表示进入结构的序列，然后再输入n个整数，表示离开结构的序列，要求根据进入结构的序列和离开结构的序列来判断该结构是栈还是队列。如果该结构只能是一个栈，则输出“stack”；如果该结构只能是一个队列，则输出“queue”；如果该结构既可以是栈又可以是一个队列，则输出“both”；如果该结构既不可能是栈又不可能是一个队列，则输出“neither”；

例如：

若输入：

3

1 2 3

3 2 1

则输出：stack

若输入：

3

1 2 3

1 2 3

则输出：queue

若输入：

3

1 2 1

1 2 1

则输出：both

若输入：

3

1 2 3

2 3 1

则输出：neither

7.优先级队列

计算机学院学生会只有一台打印机，它承担了非常繁重的打印工作。有时在打印机队列中有上百份的文件要打印，你可能要等上几个小时才能得到你想要的打印文件。

因为有些打印工作比较重要，所以学生会主席发明并实现了打印工作队列的一个简单的优先系统。每个打印工作被赋予了一个从1到9的优先级（9是最高优先级，1是最低优先级，两个任务的优先级是可以相同的），打印机的操作如下：

（1）将打印任务队列中的第一个打印任务j从队列中取出；

（2）如果在队列中有优先级高于j的打印任务，则不打印任务j，而是将任务j移到队列的最后端；

（3）如果在队列中没有优先级高于j的打印任务，则打印任务j，然后任务j从打印任务队列中消失。

采用这一方法后，所有重要的文件就能很快被打印。当然，令人烦恼的是其他的要被打印的文件就要等上更多的时间。

现在的任务是确定你的打印任务什么时候能完成，请你写个程序来计算它。给出当前队列中的任务个数以及你的打印任务在队列中的位置，再给出队列中各个任务的优先级，计算需要多长时间你的打印任务能打印完成。假设队列中不会加入附加的工作。为了使计算简单化，假设打印每个任务都恰好花费一分钟，向队列中添加一项打印工作和移走一项打印工作是在瞬间完成的，不花费时间。

要求：

首先输入打印任务队列中的任务个数n（1=<n<=100），这n个任务的编号是从0到n-1的，即第1个任务的编号是0，第2个任务的编号是1，依此类推。接下来输入你的打印任务的编号m（0=<m<=n-1），最后输入n个整数，表示这n个任务的优先级，范围从1到9，第1个整数表示编号为0的任务的优先级，第2个整数表示编号为1的任务的优先级，依此类推。要求输出你的打印任务完成所需的时间。

例如：

若输入：

1 0

5

则输出：1

若输入：

4 2

1 2 3 4

则输出：2

若输入：

6 0

1 1 9 1 1 1

则输出：5

**8.栈**

栈是一种重要的数据结构，具有后进先出的特性，它的两个重要操作是进栈和出栈，这里用i代表入栈，o代表出栈。对于给定的一个单词，通过一连串的栈操作颠倒字母间的顺序可以构造出另一个单词。例如，这里有两组连续的栈操作，它们都可以将TROT转换成TORT：

i i i i o o o o

i o i i o o i o

现在给你一对单词，你的程序应该可以通过一系列的栈操作把第一个单词转换成第二个单词。

键盘输入两个单词，第一行为源单词，第二行为目标单词，设计算法并编写程序输出所有可能的将源单词转换为目标单词的栈操作序列（由i和o组成的序列），如果不存在这样的序列输出“无”。

测试数据1：若输入

madam

adamm

则输出：i i i i o o o i o o

i i i i o o o o i o

i i o i o i o i o o

i i o i o i o o i o

测试数据2：若输入

bahama

bahama

则输出：i o i i i o o i i o o o

i o i i i o o o i o i o

i o i o i o i i i o o o

i o i o i o i o i o i o

测试数据3：若输入

long

short

则输出：无

测试数据4：若输入

eric

rice

则输出：i i o i o i o o

**9、算术表达式求值问题**

问题描述：根据算术运算符的优先级，根据输入的算术表达式，求表达式的值。

例如，输入的算术表达式形式为：12+32\*4, 3\*(12+24/(2+4)), 3.6\*(4.3+5)

10．马踏棋盘问题（要求，设计算法为非递归算法）

设计问题：有一个8\*8的方格棋盘（如下图所示），现有一匹马从任意一个位置（方格）出发，给出一种方案使马走遍棋盘中的每一个方格，且每个方格只走过一次（马走日字）。

程序的输入：输入马的初始位置（相应的坐标）。

程序的输出：马从初始位置走遍棋盘的过程

棋盘及马的走法

11.文本编辑

功能：从键盘输入一页文字，静态存储在一个文件中

要求：（1）分别统计出其中英文字母数和空格数及整篇文章总字数；

（2）统计某一字符串在文章中出现的次数，并输出该次数；

（3）删除某一子串，并将后面的字符前移。

　　 （4）存储结构使用线性表，分别用几个子函数实现相应的功能；

输入数据的形式和范围：可以输入大写、小写的英文字母、任何数字及标点符号。

输出形式：

(1)分行输出用户输入的各行字符；

(2)分行输出"全部字母数"、"数字个数"、"空格个数"、"文章总字数"

(3)输出删除某一字符串后的文章；

12.二叉树的层次遍历

关于二叉树的遍历方法，除了我们所熟知的先序遍历、中序遍历和后序遍历方法外，还有一种遍历方法，称为层次遍历。所谓层次遍历方法是指从二叉树的根结点开始自上而下，每一层从左到右逐一访问每个结点。例如，若二叉树如下所示。

则该二叉树的层次遍历序列为：1 2 3 5 6 7 10 11 13

假设二叉树采用二叉链表存储结构，其二叉链表的类型定义如下。

typedef struct bnode{

int data; //结点的数据域用大于0的整数表示

struct bnode \*lchild;

struct bnode \*rchild;

}bnode,\*btree;

要求：

首先采用先序遍历的方法来创建一棵二叉树，创建时用-1来表示空树。最后输出所创建二叉树的层次遍历序列。

例如，如果要创建上图所示的二叉树：

应输入：

1 2 -1 5 10 -1 -1 11 -1 -1 3 6 -1 13 -1 -1 7 -1 -1

则输出：

1 2 3 5 6 7 10 11 13

13.小球下落

给定一棵深度为depth的满二叉树，并对该二叉树从根结点开始自上而下，每一层从左到右，从1开始进行编号，则结点的编号序列就是1,2,3,…,2depth-1。如下图给出的深度为4的满二叉树。

现在在结点1处放一个小球，它会往下落。二叉树中每个结点处都有一个开关，初始时全部关闭，每当有小球落到一个结点时，该结点上的开关的状态就会改变。当一个小球落到某个结点时，如果该结点上的开关的状态是打开的，则往左走，否则，往右走，直到走到叶子结点。

现在有number个小球依次从结点1处开始下落，那么最后一个小球将会落到哪里呢？

输入满二叉树的深度depth（depth<=10）和小球个数number，输出第number个小球最后所在的叶子结点的编号。

例如：

若输入4 2，则输出12

若输入3 4，则输出7

若输入10 1，则输出512

若输入2 2，则输出3

若输入8 128，则输出255

14.二叉排序树中的最小结点和最大结点

给定一棵无穷的满二叉排序树，结点的编号是1，2，3，4，…，即该二叉排序树的中序遍历序列是从1开始的递增有序序列。如下图所示。

对于树中一棵根结点的编号为X的子树来说，沿着X的左孩子结点，以及左孩子结点的左孩子结点，一路向左直至到达最后一层，可以获得以X为根的树中编号最小的结点；若沿着X的右孩子结点，以及右孩子结点的右孩子结点，一路向右直至到达最后一层，可以获得以X为根的树中编号最大的结点。

现在的问题是，在一棵根为X的子树中，结点的最小编号和最大编号分别是什么？

要求输入一个整数X，表示一棵子树的根结点的编号，输出以X为根的树中结点的最小编号和最大编号。

例如：

若输入：

8

则输出：

1 15

若输入：

12

则输出：

9 15

15.二叉排序树

设二叉排序树的二叉链表存储结构的类型定义如下：

**typedef struct node{**

**int data; //用整数表示一个结点的名**

**struct node \*LChild,\*RChild; //左右指针域**

**}BSTNode,\*BSTree;**

设计算法并编写程序求解以下几个问题。

（1）键盘输入一个元素序列创建一棵二叉排序树，输出该二叉排序树的中序遍历序列；

例如，若输入45,24,55,12,37,53,60,23,40,70则创建的二叉排序树为：

输出结果为：12 23 24 37 40 45 53 55 60 70

（2）在（1）中所得的二叉排序树中插入一个值为58的结点，再输出它的中序遍历序列，输出结果为：12 23 24 37 40 45 53 55 **58** 60 70

（3）在（1）中所得的二叉排序树中删除值为45的结点，再输出它的中序遍历序列，输出结果为：12 23 24 37 40 53 55 58 60 70

（4）教材中P220给出的二叉排序树的删除操作算法中，是用左子树中最右下结点来替代要被删除的结点（即为要被删除结点的中序前驱），也可以用右子树中最左下结点来替代要被删除的结点（即为要被删除结点的中序后继），根据此思路修改P220算法8-6，并利用修改后的删除算法删除（1）中二叉排序树的值为45的结点，再输出它的中序遍历序列，输出结果为：12 23 24 37 40 53 55 58 60 70

（5）利用（1）中所得的二叉排序树的所有叶子结点构造一个带头结点的单链表L。要求不能破坏这棵二叉排序树。所得的单链表L如下。

23

40

53

70

^

L

输出该链表各结点的值，输出结果为：23 40 53 70

（6）设计算法将（1）中所得的二叉排序树的左右子树进行交换，由于二叉树是一种递归定义，所以子树的左右两棵子树也要相交换，依此类推。最后输出所得到的二叉树的中序遍历序列。

例如，经过上述操作后，（1）中所得的二叉排序树变为如下形式。

输出该二叉树的中序序列，结果为：70 60 55 53 45 40 37 24 23 12

（7）设计算法统计并输出（1）中所得的二叉排序树中只有一个孩子结点的结点个数。输出结果为：3

（8）在（1）中所得的二叉排序树中，设计算法并编写程序输出结点40的所有祖先结点。输出结果为：45 24 37

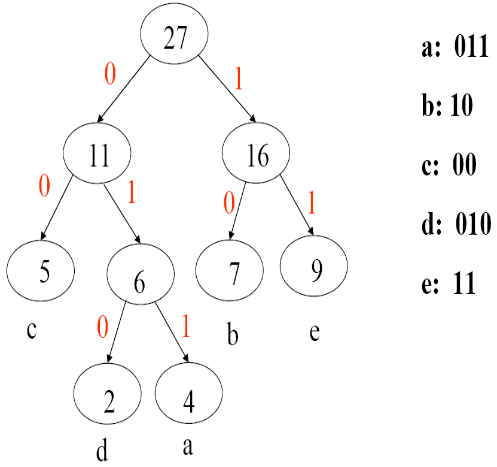
16．哈夫曼编/译码器

设计一个哈夫曼编码/译码系统，对字符串进行编码和译码

**基本要求：**

* 从键盘输入一个字符串，以回车结束；
* 统计字符串中各个字符出现的次数；
* 以各个字符出现的次数为叶子结点的权值构造一棵哈夫曼树，并为每个叶子结点构造哈夫曼编码；
* 输出每个叶子结点的哈夫曼编码；
* 根据编码结果和编码表还原字符串；

（1）电文编码：假如有一份电文中共使用5个字符：a、b、c、d、e，它们的出现频率依次为4、7、5、2、9，其所对应的哈夫曼树(**权值最小的作为左子树，次小的作为右子树**)和每个字符对应的哈夫曼编码如下。



要求从键盘输入若干字符，输出每个字符的哈夫曼编码。

（2）电文译码：给出一段二进制代码的电文，要求根据前面构造的哈夫曼树进行译码。在前面编码的基础上，键盘输入一段电文，则能在屏幕上显示出自动翻译好的电文。比如根据上图显示的哈夫曼树，键盘输入电文如下：1011010，屏幕上能显示自动翻译的结果为bed

说明事项：

（1）在求解本问题时，所涉及到的存储结构可参考教材中**P158和P160**所给出的结点结构。

（2）界面要求：有合理的提示，每个功能可以设立菜单，根据提示，可以完成相关的功能要求。并要明确，只有在使用哈夫曼树进行编码的前提下，才有可能进行译码。

（3）考虑将此程序设计完善，比如可以将编码后的字符代码保存到文件中，将需要翻译的代码保存到另一个文件中，最后把自动翻译后的结果也保存到文本文档中等等。

**17、简单书目管理系统**

问题描述：参照教材6.6.4节，设计完成一个简单的书目管理系统。

要求，在原有基础上至少还应扩充以下功能：

（1）书目的查询功能。其查询结果应为从根目录至此目录的路径。

（2）书目的删除功能。删除某个书目时，若其还有下级书目，则还须删除其所有的下级书目。

该系统的基本的整体功能结构如下：

书目管理系统

维护图书目录

显示图书目录

查询书目信息

添加新目

删除书目

修改书目

18．图的相关操作

设计算法并编写程序求解以下几个问题。

（1）设计一个算法将无向图的邻接矩阵转换为邻接表，要求输入图的邻接矩阵，调用转换算法得到它的邻接表，然后输出图的邻接表。

（2）设计一个算法将无向图的邻接表转换为邻接矩阵，要求采用邻接表存储结构创建一个无向图，调用转换算法得到它的邻接矩阵，然后输出图的邻接矩阵。

（3）有向图G采用邻接表存储结构，设计算法删除该图的一条弧<vi,vj>，顶点vi和顶点vj从键盘输入。首先基于邻接表存储结构创建一个有向图，然后输出它的邻接表，接下来调用删除操作算法删除弧<vi,vj>，最后输出删除这条弧后所得到的图的邻接表。

（4）无向图G采用邻接矩阵存储结构，设计算法判断该图是否为连通图，若是算法返回1，否则返回0，同时统计该图的连通分量的个数。以某个无向图为实例编程实现。

（5）有向图采用邻接表存储结构，请分别借助于深度优先遍历算法和广度优先遍历算法，判断有向图中是否存在顶点vi到顶点vj的路径，vi和vj从键盘输入，若存在，则输出这条路径，否则输出“无”。

19.拓扑排序

如下图所示的一个有向网：

v0

v1

v2

v3

v4

v5

v6

基于邻接矩阵存储结构并用栈作为辅助的存储空间，改写教材中P189的算法7-16（拓扑排序算法），并以上述有向图为测试实例编程实现。

**20.自动排课系统**

问题描述：如下表格给出的是课程为计算机专业学生所需学习的专业课程，现欲将这些课程安排在5个学期学完。设计程序根据给定先行条件的课程信息表，实现课程的自动排课系统。有关要求如下：

（1）每学期至多安排3门课。

（2）某门课程在开设时，其先行课程必需已开设过。

|  |  |  |
| --- | --- | --- |
| 代码 | 课程名 | 先行课 |
| C1 | 程序设计基础 | 无 |
| C2 | 离散数学 | 程序设计基础 |
| C3 | 数据结构 | 程序设计基础、离散数学 |
| C4 | 汇编语言 | 程序设计基础 |
| C5 | 编译原理、 | 汇编语言、数据结构 |
| C6 | 操作系统 | 数据结构 |
| C7 | 计算机网络基础 | 无 |
| C8 | 数据库原理 | 数据结构 |
| C9 | 面向对象编程 | 程序设计基础 |
| C10 | 网络编程 | 面向对象编程、计算机网络基础 |
| C11 | 网站开发技术 | 数据库原理、网络编程 |
| C12 | 软件开发技术设计 | 面向对象编程、数据库原理 |

21.最小生成树

一个县城有n个村庄，现在要在这n个村庄之间修建公路，你可以在任意两个村庄之间修建公路，现在要寻找一种修路方案，使得任意两个村庄之间都有公路相通，且使修路的费用达到最少。现在假设这n个村庄在同一个平面上，给定各个村庄所在的坐标位置（从键盘输入），请设计算法求解一种最优的修路方案，并输出修路所需的最小费用。

例如，若有5个村庄，各个村庄所在的坐标位置如下：

0 0

0 1

1 1

1 0

0.5 0.5

则修建公路所需的最小费用为2.83

**22. 最短路径问题**

（1）dijkstra算法适合求解权值非负的**单源**最短路径问题，即求一个顶点到其余各个顶点的最短路径。现给定一个有向网G=(V,E)，各条边上的权值均非负，给定G中一个顶点t，要求求出其余各个顶点到顶点t的最短路径长度并构造最短路径，这个问题称为**单目标**最短路径问题。基于邻接矩阵存储结构设计算法并以某个有向网为测试实例编程实现。

提示：想办法将**单目标**最短路径问题转换为**单源**最短路径问题！

（2）假设dijkstra算法采用邻接矩阵存储结构，利用dijkstra算法求有向网中**任意两个顶点间**的最短路径长度并构造最短路径。以某个有向网为测试实例编程实现。

（3）dijkstra算法不仅可以求解**有向网**中权值非负的单源最短路径问题，对于**无向网**中权值非负的单源最短路径问题，同样适用。

设G=(V,E)是一个连通无向网，采用邻接矩阵存储结构，每条边上的权值均非负，从G中任取一个顶点i，考虑顶点i到各个顶点的最短路径长度：d(i,0)，d(i,1)，……，d(i,n-1)

其中d(i,k)（0≤k≤n-1）表示顶点i到顶点k的最短路径长度，规定d(i,i)=0。这n个值中**最大**的那个称为顶点i的最大距离，记为L(i)，在所有顶点中，使L(i)达到**最小**的顶点称为网G的**中心**。利用dijkstra算法编程求解给定无向网的中心。

例如，下面这个无向网的中心为顶点5

2

1

3

5

4

6

3

2

2

6

1.5

1.8

0

3

1.5

2.5

（4）假设将3题中的网看成是一个矿区，它有7个矿，分别在顶点0,1，…..,6处，这7个矿每天的矿产量分别是0：3000t，1：2000t，2：7000t，3：1000t，4：5000t，5：1000t，6：4000t，如下图所示。

2

1

3

5

4

6

3

2

2

6

1.5

1.8

0

3

1.5

2.5

(3000)

(5000)

(7000)

(1000)

(2000)

(1000)

(4000)

现在要在顶点0,1，…..,6中选一个来建选矿厂，如果选矿厂建在了顶点i，那么我们关心的是将各个矿生产的矿石都运到顶点i处的运输量是多少，然后再来确定在哪里建选矿厂最好。一般情况下，我们以运输的t×km数来度量运输量的大小，一吨货物运输一公里就叫1t×km。例如，如果选矿厂建在顶点0处，则总的运输量表示为g(0)，它等于

**g(0)=3000×0+2000×3+7000×5+1000×6.3+5000×9.3+1000×4.5+4000×6=122300(t×km)**

如果选矿厂建在顶点1处，则总的运输量表示为g(1)，它等于

g(1)=3000×3+2000×0+7000×2+1000×3.3+5000×6.3+1000×1.5+4000×3=68300(t×km)

显然，选矿厂建在顶点1要比建在顶点0处好，因为建在顶点1处时运输量较小。当然，顶点1是不是最好的还不能确定，应把g(0)，g(1)，…，g(6)都算出来再比较一下，才可以选出一个最好的建厂地点。

一般情况下，给定无向网G=(V,E)，采用邻接矩阵存储结构，每条边上的权值均非负，G的每个顶点i还有一个“产量”A(i)，对于每个顶点i，令：

**g(i)=A(0)×d(0,i)+ A(1)×d(1,i)+……+A(n-1)×d(n-1,i)**

其中d(k,i)（0≤k≤n-1）表示顶点k到顶点i的最短路径长度。

g(i)代表了把各个点的物资运到顶点i处所花费的t×km，对于具有n个顶点的无向网G来说，使得g(i)达到**最小**的那个顶点i就称为该网的**中央点**，利用dijkstra算法编程求解给定无向网的中央点。

例如，上述无向网的中央点为顶点2。

**23、DNA排序**

一个字符串的逆序数定义为：该字符串中出现的次序相反的字符对（按照字母表顺序）的数目。

例如：字符串“DAABEC”的逆序数是5，因为D比它右边的4个字母大，而E比它右边的1个字母大。字符串“AACEDGG”的逆序数则是1（E和D），几乎已经排好序了。字符串“ZWQM”的逆序数是6，完全没有排好序。

DNA字符串中只包含4种字符，即字母A，G，C，T。给定若干个DNA字符串，将这些DNA字符串按照每个字符串的逆序数从小到大排序。

首先输入字符串的个数n，然后输入每个字符串的长度m，接下来输入n个DNA字符串，输出按照每个字符串的逆序数从小到大排好序后的n个字符串。

例如：

若输入:

6 10

AACATGAAGG

TTTTGGCCAA

TTTGGCCAAA

GATCAGATTT

CCCGGGGGGA

ATCGATGCAT

则输出：

CCCGGGGGGA

AACATGAAGG

GATCAGATTT

ATCGATGCAT

TTTTGGCCAA

TTTGGCCAAA

**24、字符串排序**

给定n个字符串，在这n个字符串中有相同的字符串，不同的字符串只有num个。要求首先输入字符串的个数n，然后输入n个字符串，将这n个字符串中num个不同的字符串按照字典序排序，并输出每个字符串在这n个字符串中所占的比例，精确到4位小数。

例如：

若输入：

29

Red Alder

Ash

Aspen

Basswood

Ash

Beech

Yellow Birch

Ash

Cherry

Cottonwood

Ash

Cypress

Red Elm

Gum

Hackberry

White Oak

Hickory

Pecan

Hard Maple

White Oak

Soft Maple

Red Oak

Red Oak

White Oak

Poplan

Sassafras

Sycamore

Black Walnut

Willow

则输出：

Ash 13.7931

Aspen 3.4483

Basswood 3.4483

Beech 3.4483

Black Walnut 3.4483

Cherry 3.4483

Cottonwood 3.4483

Cypress 3.4483

Gum 3.4483

Hackberry 3.4483

Hard Maple 3.4483

Hickory 3.4483

Pecan 3.4483

Poplan 3.4483

Red Alder 3.4483

Red Elm 3.4483

Red Oak 6.8966

Sassafras 3.4483

Soft Maple 3.4483

Sycamore 3.4483

White Oak 10.3448

Willow 3.4483

Yellow Birch 3.4483

25、整理单词

本题的任务是编写一个整理单词的程序。

问题的输入包含4个部分：

（1）字典，包含至少一个、至多100个单词，每个单词一行；

（2）一行内容为XXXXXX，表示字典结束；

（3）一个或多个你要整理的“字符串”；

（4）一行内容为XXXXXX，表示输入结束。

字典中的单词以及要整理的字符串都是由小写英文字母组成的，每个单词或字符串至少包含1个字母，至多包含6个字母（XXXXXX是由大写的X组成的），字典中的每个单词只出现一次。

对于输入中每个要整理的“字符串”，输出在字典里存在的单词，该单词与要整理的字符串长度相同，且包含的字母是完全相同的，只是字母的排列可以不同。如果在字典里找到不止一个单词与要整理的字符串相对应时，要把它们按照字典序排序，每个单词占一行。如果在字典里没找到相对应的单词则输出“NOT A VALID WORD”。每输出对应的一组单词或“NOT A VALID WORD”后要输出“\*\*\*\*\*\*”。

例如：

若输入：

tarp

given

score

refund

only

trap

work

earn

course

pepper

part

XXXXXX

resco

nfudre

aptr

sett

oresuc

XXXXXX

则输出：

score

\*\*\*\*\*\*

refund

\*\*\*\*\*\*

part

tarp

trap

\*\*\*\*\*\*

NOT A VALID WORD

\*\*\*\*\*\*

course

\*\*\*\*\*\*

26、字符串种类统计

输入n（1=<n<=100）个字符串，每个字符串的长度都相同，均为m（1=<m<=20），这n个字符串中有相同的，要求编写程序输出n行数据，第1行输出n个字符串中只出现1次的字符串种类数，第2行输出n个字符串中出现2次的字符串种类数，第3行输出n个字符串中出现3次的字符串种类数，依此类推，第i行输出n个字符串中出现i次的字符串种类数。

要求，首先输入n和m的值，然后再输入n个字符串，输出n行，第i（1=<i<=n）行输出出现i次的字符串种类数。

例如：

若输入：

9 6

AAAAAA

ACACAC

GTTTTG

ACACAC

GTTTTG

ACACAC

ACACAC

TCCCCC

TCCCCC

则输出：

1

2

0

1

0

0

0

0

0

若输入：

4 5

qqqqq

qqqqq

qqqqq

qqqqq

则输出：

0

0

0

1

若输入：

4 5

asgdf

sadgg

sdgfg

zfghf

则输出：

4

0

0

0

若输入：

6 6

aaaaaa

cccccc

bbbbbb

bbbbbb

cccccc

cccccc

则输出：

1

1

1

0

0

0

**27、上下文无关文法的表示**

问题描述：一个上下文无关文法G的四元组定义如下：

G=（VN, VT, P, S）

其中：VN——称为非终结符集，

VT——称为终结符集

P——称为产生式集，其每个产生式为形如A->β。其中A∈VN，而β是由终结符和非终结符组在的任意符号串（包括空串）。

S——为开始符号，S∈VN

给出上下文无关文法的数据结构表示，并实现其有关基本操作。文法的基本操作有：

（1）locateVT(G, a)：求文法G中终结符a的位置。

（2）locateVN(G, A)：求文法G中非终结符A的位置。

（3）outputPonID(G, i)：输出编号为i的产生式。

（4）outputPonVN(G, A)：输出所有左部为A的产生式。（即产生式为：A->β）

（5）creatG(G)：根据输入信息创建文法G。

（6）outputG(G)：输出文法G。

例如，文法的四元组表示：

G=（{S, A, B}, {a, b}, P, S），其中P中的产生式有：

(1) S->ab (5) A->bAA

(2) S->bA (6) B->b

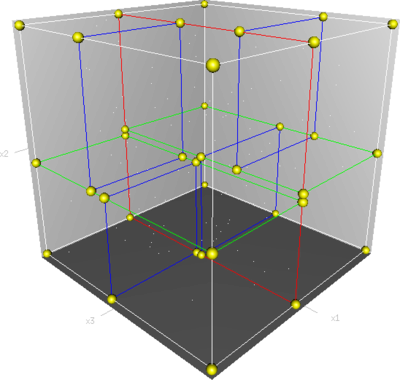
(3) A->a (7) B->bS

(4) A->aS (8) B->aBB

**28.k-d树（扩展内容）**

在[计算机科学](http://baike.baidu.com/view/92404.htm)里，k-d树（k-维树的缩写）是在k维[欧几里德空间](http://baike.baidu.com/view/454406.htm)组织点的[数据结构](http://baike.baidu.com/view/9900.htm)。k-d树可以使用在多种应用场合，如多维键值搜索。k-d树是[二叉树](http://baike.baidu.com/view/88806.htm)的一种特殊情况。

k-d树是每个节点都为k维点的[二叉树](http://baike.baidu.com/view/88806.htm)。所有非[叶子节点](http://baike.baidu.com/view/544292.htm)可以视作用一个超平面把空间分割成两部分。在超平面左边的点代表节点的左子树，在超平面右边的点代表节点的右子树。超平面的方向可以用下述方法来选择：每个节点都与k维中垂直于超平面的那一维有关。因此，如果选择按照x轴划分，所有x值小于指定值的节点都会出现在左子树，所有x值大于指定值的节点都会出现在右子树。这样，超平面可以用该x值来确定，其法矢为x轴的单位向量。



一个三维k-d树。第一次划分（红色）把根节点（白色）划分成两个节点，然后它们分别再次被划分（绿色）为两个子节点。最后这四个子节点的每一个都被划分（蓝色）为两个子节点。至此已经不能再被划分，最后得到的八个节点称为叶子节点。

有兴趣的同学可参考以下博客

<http://underthehood.blog.51cto.com/2531780/687160>

阅读完材料后可以自行设计设计题目

说明：

**（1）对于中等难度及以上的题目，至少选择两道题目来进行设计；对于简单的题目，要求至少选择三道题目来进行设计。具体题目的选择可与指导教师商议确定。**

**（2）除上述题目外，经指导教师同意后，也可以自选一些新颖的题目。**