第 1 组　1~3
第 2 组　4~6
第 3 组　7~10
第 4 组　11~14
第 5 组　15~18
第 6 组　19~22
第 7 组　23~26

CS 试题

id	9000000001
title	顺序表的插入运算（耿 2.4）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	已知顺序表 L 递增有序，编写程序，将 X 插入到线性表的适当位置上，以保持线性表的有序性。
input	第一行输入顺序表元素个数 elenum；(0<elenum<1000) 第二行输入顺序表 L； 第三行输入插入值 X。
output	输出插入 X 后的有序顺序表
sample_input	7 2 3 4 5 6 7 8 1
sample_output	1 2 3 4 5 6 7 8
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000002
title	线性表的就地逆置（耿 2.9）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	试分别以不同的存储结构实现线性表的就地逆置算法，即在原表的存储空间将线性表（a1,a2,…,an）逆置为(an,an-1,…,a1)。 (1) 以一维数组作存储结构。 (2) 以单链表作存储结构。
input	第一行输入线性表元素个数 elenum；(0<elenum<1000) 第二行输入 elenum 个数，作为线性表中的元素（a1,a2,…,an）。

output	分两行分别输出要求（1）和要求（2）的线性表逆置结果(an,an-1,…,a1)。
sample_input	5 2 5 3 7 15
sample_output	15 7 3 5 2 15 7 3 5 2
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000003
title	顺序表的删除（严 2.29）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	已知 A,B 和 C 为三个非递减有序的线性表，现要求对 A 表作如下操作：删去那些既在 B 表中出现又在 C 表中出现的元素。试对顺序表编写实现上述操作的算法。
input	第一行输入 3 个正整数 m,n,p(m,n,p<=100)，用空格分开，分别表示三个线性表中的元素个数，其后 3 行依次输入 A,B,C 表中的元素。
output	输出实现上述操作后的 A 表。
sample_input	8 5 6 1 2 3 4 5 6 6 7 2 3 5 9 12 2 4 5 6 12 13
sample_output	1 3 4 6 6 7
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000004
title	单链表的归并（耿 2.11）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	假设两个按元素值非递减有序排列的线性表 A 和 B，均以单链表作为存储结构，试编写程序，将 A 表和 B 表归并成一个按元素值非递增有序排列的线性表 C，并要求利用原表（即 A 表和 B 表的）结点空间存放表 C。
input	第一行输入两个正整数 m,n(m,n<=100)，用空格分开，表示线性表 A 和 B 中元素个数，其后两行分别输入单链表 A 和 B。

output	输出单链表 C。
sample_input	5 5 1 3 7 12 16 2 6 7 13 20
sample_output	20 16 13 12 7 7 6 3 2 1
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000005
title	单链表的删除（严 2.29）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	已知 A,B 和 C 为三个非递减有序的线性表，均以单链表作为存储结构。现要求对 A 表作如下操作：删去那些既在 B 表中出现又在 C 表中出现的元素。试对单链表编写实现上述操作的算法，并释放 A 表中的无用结点空间。
input	第一行输入 3 个正整数 m,n,p(m,n,p<=100)，用空格分开，表示三个线性表中的元素个数，其后 3 行依次输入 A,B,C 表中的元素。
output	输出实现上述操作后的 A 表。
sample_input	8 5 6 1 2 3 4 5 6 6 7 2 3 5 9 12 2 4 5 6 12 13
sample_output	1 3 4 6 6 7
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000006
title	LOCATE 操作（严 2.38）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	设有一个双向循环链表，每个结点中除有 pre，data 和 next 三个域外，还增设了一个访问频度域 freq。在链表被起用之前，频度域 freq 的值均初始化为零，而每当对链表进行一次 LOCATE(L,x)的操作后，被访问的结点（即元素值等于 x 的结点）中的频度域 freq 的值便增 1，同时调整链表中结点之间的次序，使其按访问频度非递增的次序

	排列，以便始终保持被频繁访问的结点总是靠近表头结点。试编写符合上述要求的 LOCATE 操作的程序。
input	第一行输入双向循环链表的节点数 m 和被访问的节点数 n， 第二行输入双向循环链表各节点的值， 第三行依次输入被访问的节点。
output	输出符合上述要求的双向循环链表。 输出经过 n 次 LOCATE 后的链表。
sample_input	7 1 a b c d e f g d
sample_output	d a b c e f g
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000007
title	表达式括号匹配（严 3.19）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	假设一个算术表达式中可以包含三种括号：圆括号“(”和“)”、方括号“[”和“]”和花括号“{”和“}”，且这三种括号可按任意的次序嵌套使用（如：…[…{…}…[…]…]…[…]…(…)…）。编写判别给定表达式中所含括号是否正确配对出现的程序（已知表达式已存入数据元素为字符的顺序表中）。
input	输入算术表达式，换行结束。
output	若给定表达式中所含括号正确配对，则输出 yes，否则输出 no。
sample_input	[5+(6-3)]-(2+3)]
sample_output	no
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000008
title	逆波兰式（耿 3.8）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	假设表达式由单字母变量和双目四则运算算符构成。试编写程序，将一个通常书写形

	式且书写正确的表达式转换为逆波兰式。
input	输入由单字母变量和双目四则运算算符构成的表达式。
output	输出其逆波兰式。
sample_input	(a+b)*c
sample_output	ab+c*
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000009
title	循环队列（严 3.30）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	假设将循环队列定义为：以域变量 rear 和 length 分别指示循环队列中队尾元素的位置和内含元素的个数。编写相应的入队列和出队列的程序，并判断循环队列是否队满（在出队列的算法中要返回队头元素）。
input	假设队列数组为 Queue[MAXSIZE]，第一行输入队列大小 N，第二行开始输入若干入队元素，队满时，停止入队。第三行输入出队元素。
output	输出入队出队操作后的循环队列，并返回出队元素的队头元素。
sample_input	5 3 4 6 2 7 4
sample_output	6 2 7 6
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000010
title	k 阶斐波那契数列（严 3.32）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	试利用循环队列编写 k 阶斐波那契数列中前 n+1 项 （f(0),f(1),…,f(n)）的程序，要求满足：f(n)<=max 而 f(n+1)>max，其中 max 为某个约定的常数。（注意：本题所用循环队列的容量仅为 k，则在程序执行结束时，留在循环队列中的元素应是所求 k 阶斐波那契序列中的最后 k 项 f(n-k+1),…,f(n))。

input	输入常数 max，阶数 k，用空格隔开。
output	输出 k 阶斐波那契数列中的最后 k 项 f(n-k+1),…,f(n)。
sample_input	14 2
sample_output	8 13
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000011
title	循环右移(耿 5.2)
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	编写程序，将一维数组 A（下标从 1 开始）中的元素循环右移 k 位，要求只用一个元素大小的附加存储。
input	第一行输入一维数组 A 的长度 n 和循环位移位数 k(0<n<100;0<k<100)，用空格分开。 第二行输入 n 个元素。
output	输出循环右移 k 位后的一维数组。
sample_input	6 3 1 2 3 4 5 6
sample_output	4 5 6 1 2 3
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000012
title	以三元组表为存储结构实现矩阵相加（耿 5.7）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	假设稀疏矩阵 A 和 B 均以三元组表作为存储结构。试编写矩阵相加的程序，另设三元组表 C 存放结果矩阵。矩阵大小为 m 行 n 列(0<m,n<100)
input	第一行输入 t1，t2(0<t1,t2<100) ,t1 和 t2 分别是矩阵 A 和 B 中非零元素的个数，后面 t1+t2 行分别输入 A 和 B 中的元素，用三元组表示。
output	输出三元组表 C。
sample_input	3 3 1 2 3 3 2 1

	3 4 2 1 1 4 3 2 5 3 4 1
sample_output	1 1 4 1 2 3 3 2 6 3 4 3
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000013
title	以十字链表为存储结构实现矩阵相加（严 5.27）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	以十字链表为存储结构，编写程序，将稀疏矩阵 B 加到稀疏矩阵 A 上。
input	第一行输入四个正整数，分别为稀疏矩阵 A 和稀疏矩阵 B 的行数 m、列数 n、稀疏矩阵 A 的非零元素个数 t1 和稀疏矩阵 B 的非零元素个数 t2。接下来的 t1+t2 行三元组表示，其中第一个元素表示非零元素所在的行值，第二个元素表示非零元素所在的列值，第三个元素表示非零元素的值。
output	输出相加后的矩阵三元组。
sample_input	3 4 3 2 1 1 1 1 3 1 2 2 2 1 2 1 2 2 3
sample_output	1 1 1 1 2 1 1 3 1 2 2 5
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000014
title	求广义表深度（严 5.30）
time_limit	3000MS

memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	试按表头、表尾的分析方法编写求广义表的深度的递归程序。
input	输入一串以‘(’开始，以‘(’结束的字符串，并且输入的左右括号必须匹配，如：(),(())……
output	分别输出按表头、表尾分析方法求广义表深度的结果，每个结果占一行。
sample_input	((a,b,(c,(d,e),f)),g)
sample_output	4 4
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000015
title	建立二叉树的二叉链表存储结构（严 6.70）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	如果用大写字母标识二叉树结点，则一颗二叉树可以用符合下面语法图的字符序列表示。试编写递归程序，由这种形式的字符序列，建立相应的二叉树的二叉链表存储结构(附图见《严蔚敏：数据结构题集（C 语言版）》第 45 页 6.70)。
input	输入如图所示的字符序列。
output	建立相应二叉树的二成叉链表存储结构，并先序遍历输出。
sample_input	A(B(#,D),C(E(#,F),#))
sample_output	AB#DCE#F#
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000016
title	计算二叉树叶子结点数目（耿 6.14）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	二叉树按照二叉链表方式存储，编写程序，计算二叉树中叶子结点的数目。
input	按先序输入二叉树各结点，其中#表示取消建立子树结点。
output	输出二叉树中叶子节点的数目。

sample_input	ABD##EH###CF#I##G##
sample_output	4
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000017
title	输出以二叉树表示的算术表达式（严 6.51）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	编写程序，输出以二叉树表示的算术表达式，若该表达式中含有括号，则在输出时应添上。
input	按先序输入一行字符，其中#表示取消建立子树结点，即所有叶子节点均为#。
output	输出该二叉树所表示的算术表达式（若表达式中含有括号，则在输出时应添上）。
sample_input	*+a(###b#)##c##
sample_output	(a+b)*c
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000018
title	建立二叉树的二叉链表（严 6.65）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	已知一棵二叉树的前序序列和中序序列分别存于两个一维数组中，试编写算法建立该二叉树的二叉链表。
input	分两行分别输入一棵二叉树的前序序列和中序序列。
output	输出该二叉树的后序序列。
sample_input	ABDFGCEH BFDGACEH
sample_output	FGDBHECA
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000019
title	基于图的深度优先搜索策略（耿 7.10）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	试基于图的深度优先搜索策略编写程序，判别以邻接表方式存储的有向图中，是否存在由顶点 vi 到顶点 vj 的路径（i 不等于 j）。注意：程序中涉及的图的基本操作必须在此存储结构上实现。
input	第一行输入有向图的顶点数 n 和边数 m，用空格隔开；第二行输入顶点信息；分 m 行输入有向图边的信息，例如顶点对 1,2 表示从顶点 1 到顶点 2 的一条弧。最后一行输入待判别的顶点对 vi,vj.(0<m,n<100)
output	若有向图中存在由顶点 vi 到顶点 vj 的路径（i 不等于 j），则输出 yes；否则输出 no。
sample_input	4 4 1 2 3 4 1 2 1 3 1 4 2 3 2 3
sample_output	yes
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000020
title	基于图的广度优先搜索策略（耿 7.11）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	试基于图的广度优先搜索策略编写程序，判别以邻接表方式存储的有向图中，是否存在由顶点 vi 到顶点 vj（i 不等于 j）。注意：程序中涉及的图的基本操作必须在此存储结构上实现。
input	第一行输入有向图的顶点数 n 和边数 m，用空格隔开；第二行输入顶点信息；分 m 行输入有向图边的信息，例如顶点对 1,2 表示从顶点 1 到顶点 2 的一条弧。最后一行输入待判别的顶点对 vi,vj。(0<m,n<100)
output	若有向图中存在由顶点 vi 到顶点 vj 的路径（i 不等于 j），则输出 yes；否则输出 no。
sample_input	4 4 1 2 3 4 1 2

	1 3 1 4 2 3 2 3
sample_output	yes
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000021
title	逆波兰表达式（严 7.38）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	一个四则运算算术表达式以有向无环图的邻接表方式存储，每个操作数原子都由单个字母表示。编写程序输出其逆波兰表达式。
input	输入四则运算算术表达式。
output	输出其逆波兰表达式。
sample_input	(a+b)*c
sample_output	ab+c*
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000022
title	Dijkstra 算法（严 7.42）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	编写程序，实现以邻接表作存储结构，求从源点到其余各顶点的最短路径的 Dijkstra 算法。
input	第一行输入顶点数 n 和边数 m；第二行输入顶点信息；分 m 行输入 m 对顶点 vi，vj（表示由顶点 vi 到顶点 vj（i 不等于 j）的边）以及该弧的权值。(0<m,n<100)
output	输出从源点到其余各顶点的最短路径（不可达用-1 表示）。
sample_input	6 11 1 2 50 1 3 10 1 5 45

	2 3 15 2 5 10 3 1 20 3 4 15 4 2 20 4 5 35 5 4 30 6 4 3
sample_output	1 3 10 1 4 25 1 2 45 1 5 45 1 6 -1
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000023
title	构造哈希表（耿 8.12）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	选取哈希函数 H(k)=(3k)%11，用线性探测再散列法处理冲突。试在 0-10 的散列地址空间中，编写程序，对关键字序列 (22,41,53 46,30,13,01,67)构造哈希表，并求等概率情况下查找成功的平均查找长度。
input	无
output	输出等概率情况下查找成功的平均查找长度。
sample_input	无
sample_output	2
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000024
title	二叉排序树的判别（耿 8.6）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10

description	试编写程序，判别给定的二叉树是否为二叉排序树。设此二叉树以二叉链表作存储结构，且树中结点的关键字均不同。
input	按先序输入二叉树各结点（结点值大于 0），其中-1 表示取消建立子树结点。
output	若该二叉树为二叉排序树，则输出 yes；否则，输出 no。
sample_input	12 8 4 -1 -1 10 -1 -1 16 13 -1 -1 18 -1 -1
sample_output	yes
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000025
title	二叉排序树的插入和删除（严 9.35、9.36 和 9.37）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple
difficulty	10
description	假设二叉排序树以后继线索链表作存储结构，编写程序，满足以下要求： 1. 输出该二叉排序树中所有大于 a 小于 b 的关键字； 2. 在二叉排序树中插入一个关键字； 3. 在二叉排序树中删除一个关键字。
input	第一行按先序输入二叉排序树各结点（结点值大于 0），其中-1 表示取消建立子树结点；第二行输入要求 1 中 a、b，用空格隔开；第三行输入要求 2 中要插入的关键字；第四行输入要求 3 中要删除的关键字。
output	按照中序序列，分三行输出要求 1、要求 2 和要求 3 的结果。
sample_input	12 8 4 -1 -1 10 -1 -1 16 13 -1 -1 18 -1 -1 10 17 6 12
sample_output	12 13 16 4 6 8 10 12 13 16 18 4 8 10 13 16 18
hint	
source	INPOJ
solution_language	CS
solution	

CS 试题

id	9000000026
title	二叉排序树的合并（严 9.38）
time_limit	3000MS
memory_limit	10000KB
filesize_limit	0
category	PF_Simple

difficulty	10
description	试编写程序，将两棵二叉排序树合并为一棵二叉排序树。
input	按照先序序列，分两行输入两棵二叉排序树各结点（结点值大于 0），其中-1 表示取消建立子树结点。
output	按照中序序列输出合并后的二叉排序树。
sample_input	12 8 4 -1 -1 10 -1 -1 16 13 -1 -1 18 -1 -1 17 6 2 -1 -1 9 -1 -1 24 19 -1 -1 26 -1 -1
sample_output	2 4 6 8 9 10 12 13 16 17 18 19 24 26
hint	
source	INPOJ
solution_language	CS
solution	