《数据结构》上机实验题目

（共 8 次，每次上机 4 小时）

第一阶段（线性部分）

《数据结构》第 1 次上机题 （线性表练习）

1．编程实现书 P12 ADT List 基本操作 13 个：

（1） 用顺序存储结构实现； （2）用链式存储结构实现；

2．设元素值为整型的线性表 L，分别采用顺序结构和链式结构存储，编写函数，实现线性表的就地逆置（书 P31 4）。

3．设线性表 L，元素值为整型的且存在相同值，分别采用顺序结构和链式结构存储，编写函数，利用原空间，删除重复的元素值。

4．CSP 题目

问题描述：一幅长宽分别为 n 个像素和 m 个像素的灰度图像可以表示为一个 n×m 大小的矩阵 A。

其中每个元素 A_{ij}（0≤i<n、0≤j<m）是一个[0,L)范围内的整数，表示对应位置像素的灰度值。

具体来说，一个 8 比特的灰度图像中每个像素的灰度范围是[0,128)。

一副灰度图像的灰度统计直方图（以下简称“直方图”）可以表示为一个长度为 L 的数组 h，其中 h[x]（0≤x<L）表示该图像中灰度值为 x 的像素个数。显然，h[0]到 h[L－1]的总和应等于图像中的像素总数 n·m。

已知一副图像的灰度矩阵 A，试计算其灰度直方图 h[0],h[1],…,h[L－1]。

输入格式：

输入共 n+1 行。

输入的第一行包含三个用空格分隔的正整数 n、m 和 L，含义如前文所述。

第二到第 n+1 行输入矩阵 A。

第 i+2（0≤i<n）行包含用空格分隔的 m 个整数，依次为 $A_{i0},A_{i1},\cdots,A_{i(m-1)}$。

输出格式：

输出仅一行，包含用空格分隔的 L 个整数 h[0],h[1],…,h[L－1]，表示输入图像的灰

度直方图。

样例输入：

4 4 16

0 1 2 3

4 5 6 7

8 9 10 11

12 13 14 15

样例输出：

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

样例输入：

7 11 8

0 7 0 0 0 7 0 0 7 7 0

7 0 7 0 7 0 7 0 7 0 7

7 0 0 0 7 0 0 0 7 0 7

7 0 0 0 0 7 0 0 7 7 0

7 0 0 0 0 0 7 0 7 0 0

7 0 7 0 7 0 7 0 7 0 0

0 7 0 0 0 7 0 0 7 0 0

样例输出：

48 0 0 0 0 0 0 29

评测用例规模与约定：

全部的测试数据满足 0<n,m≤500 且 4≤L≤256。

5. CSP 题目 碰撞的小球

问题描述：

数轴上有一条长度为 L（L 为偶数)的线段，左端点在原点，右端点在坐标 L 处。有 n 个不计体积的小球在线段上，开始时所有的小球都处在偶数坐标上，速度方向向右，速度大小为 1 单位长度每秒。

当小球到达线段的端点（左端点或右端点）的时候，会立即向相反的方向移动，速度大小仍然为原来大小。

当两个小球撞到一起的时候，两个小球会分别向与自己原来移动的方向相反的方向，以原来的速度大小继续移动。

现在，告诉你线段的长度 L，小球数量 n，以及 n 个小球的初始位置，请你计算 t 秒之后，各个小球的位置。

提示：

因为所有小球的初始位置都为偶数，而且线段的长度为偶数，可以证明，不会有三个小球同时相撞，小球到达线段端点以及小球之间的碰撞时刻均为整数。

同时也可以证明两个小球发生碰撞的位置一定是整数（但不一定是偶数）。

输入格式：

输入的第一行包含三个整数 n，L，t，用空格分隔，分别表示小球的个数、线段长度和你需要计算 t 秒之后小球的位置。

第二行包含 n 个整数 a1，a2，…，an，用空格分隔，表示初始时刻 n 个小球的位置。

输出格式：

输出一行包含 n 个整数，用空格分隔，第 i 个整数代表初始时刻位于 ai 的小球，在 t 秒之后的位置。

样例输入：

3 10 5

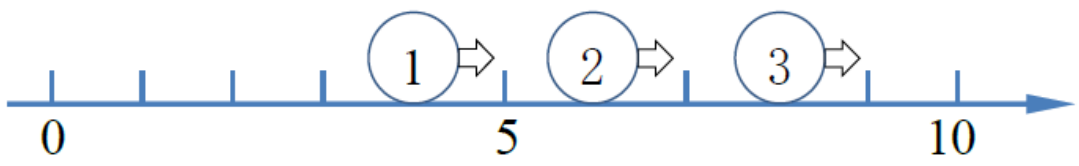
4 6 8

样例输出：

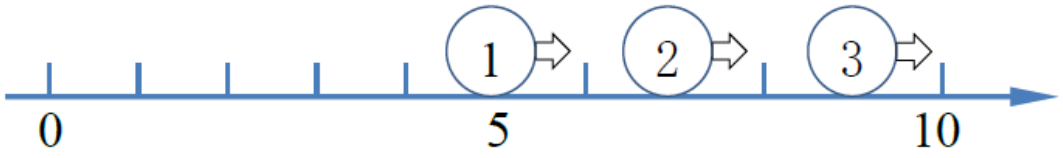
7 9 9

样例说明：

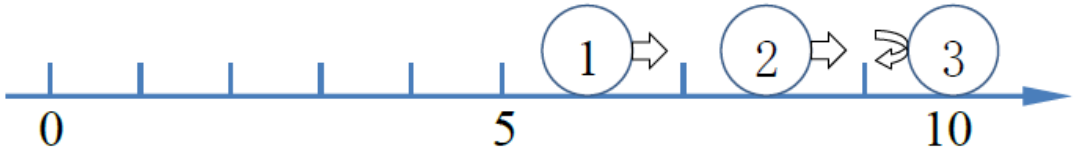
初始时，三个小球的位置分别为 4，6，8。



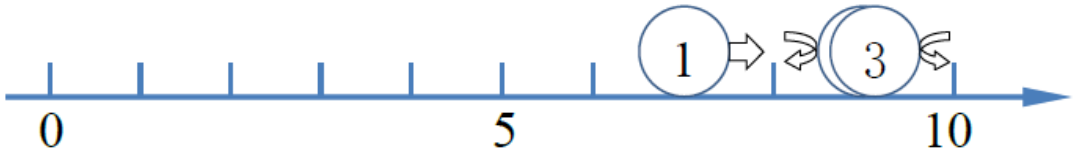
一秒后，三个小球的位置分别为 5，7，9。



两秒后，第三个小球碰到墙壁，速度反向，三个小球位置分别为 6，8，10。

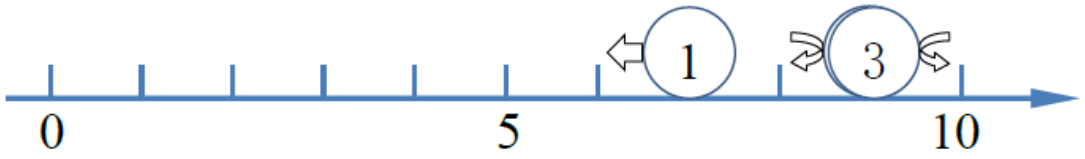


三秒后，第二个小球与第三个小球在位置 9 发生碰撞，速度反向（注意碰撞位置不一定为偶数），三个小球位置分别为 7，9，9。



四秒后，第一个小球与第二个小球在位置 8 发生碰撞，速度反向，第三个小球碰到墙壁，速度反向，三个小球位置分别为 8，8，10。

五秒后，三个小球的位置分别为 7，9，9。



样例输入：

10 22 30

14 12 16 6 10 2 8 20 18 4

样例输出：

6 6 8 2 4 0 4 12 10 2

问题分析：该问题中，只需维护好每个小球的当前位置，并随时间进行变化，最后按题设要求进行输出即可，本题目实质是一道简单的模拟题，与树、图无关。

附加题：习题集 1.19 1.20 2.19

注：习题集为严蔚敏数据结构题集

《数据结构》第 2 次上机题 （线性表练习）

1. 设元素值为整型的线性表 L，分别采用顺序结构和链式结构存储，编写函数，用选择/冒泡排序算法实现线性表的表排序。

2. 设线性表 A、B，元素值为整型，且递减有序，编写函数，实现下列功能：对采用顺序结构和链式结构 2 种存储结构，要求在 A 的空间上构成一个新线性表 C，其元素为 A 和 B 元素的并集，且表 C 中的元素值递减有序（互不相同）。

3. 输入正整数 n、m（m<n），设有 n 个人坐成一圈，从第 1 个人开始循环报数，报到 m 的人出列，然后再从下一个人开始报数，报到 m 的人又出列，如此重复，直到所有的人都出列为止。要求用链式结构和顺序结构实现，按出列的先后顺序输出每个人的信息。

4. CSP 题目

题目描述：

A1,A2,…,An 是一个由 n 个自然数（非负整数）组成的数组。我们称其中 Ai,…,Aj 是一个非零段，当且仅当以下条件同时满足：

- 1≤i≤j≤n；
- 对于任意的整数 k，若 i≤k≤j，则 Ak>0；
- i=1 或 Ai－1=0；
- j=n 或 Aj+1=0。

下面展示了几个简单的例子：

- A=[3,1,2,0,0,2,0,4,5,0,2]中的 4 个非零段依次为[3,1,2]、[2]、[4,5]和[2]；
- A=[2,3,1,4,5]仅有 1 个非零段；
- A=[0,0,0]则不含非零段（即非零段个数为 0）。

现在我们可以对数组 A 进行如下操作：任选一个正整数 p，然后将 A 中所有小于 p 的数都变为 0。试选取一个合适的 p，使得数组 A 中的非零段个数达到最大。若输入的 A 所含非零段数已达最大值，可取 p=1，即不对 A 做任何修改。

输入格式：

从标准输入读入数据。

输入的第一行包含一个正整数 n。

输入的第二行包含 n 个用空格分隔的自然数 A1,A2,…,An。

输出格式：

输出到标准输出。

仅输出一个整数，表示对数组 A 进行操作后，其非零段个数能达到的最大值。

样例 1 输入：

11

3 1 2 0 0 2 0 4 5 0 2

样例 1 输出：

5

样例 1 解释：

p=2 时，A=[3,0,2,0,0,2,0,4,5,0,2]，5 个非零段依次为 [3]、[2]、[2]、[4,5] 和 [2]；此时非零段个数达到最大。

样例 2 输入：

14

5 1 20 10 10 10 10 15 10 20 1 5 10 15

样例 2 输出：

4

样例 2 解释：

p=12 时，A=[0,0,20,0,0,0,0,15,0,20,0,0,0,15]，4 个非零段依次为 [20]、[15]、[20] 和 [15]；此时非零段个数达到最大。

样例 3 输入：

3

1 0 0

样例 3 输出：

1

样例 3 解释：

p=1 时，A=[1,0,0]，此时仅有 1 个非零段 [1]，非零段个数达到最大。

样例 4 输入：

3

0 0 0

样例 4 输出：

0

样例 4 解释：

无论 p 取何值，A 都不含有非零段，故非零段个数至多为 0。

子任务：

70% 的测试数据满足 n≤1000；

全部的测试数据满足 n≤5×105，且数组 A 中的每一个数均不超过 104。

5. CSP 题目

问题描述：小 H 和小 W 来到了一条街上，两人分开买菜，他们买菜的过程可以描述为，去店里买一些菜然后去旁边的一个广场把菜装上车，两人都要买 n 种菜，所以也都要装 n 次车。具体的，对于小 H 来说有 n 个不相交的时间段 [a1,b1],[a2,b2]…[an,bn]在装车，对于小 W 来说有 n 个不相交的时间段 [c1,d1],[c2,d2]…[cn,dn]在装车。其中，一个时间段[s, t]表示的是从时刻 s 到时刻 t 这段时间，时长为 t-s。

由于他们是好朋友，他们都在广场上装车的时候会聊天，他们想知道他们可以聊多长时间。

输入格式：

输入的第一行包含一个正整数 n，表示时间段的数量。

接下来 n 行每行两个数 ai，bi，描述小 H 的各个装车的时间段。

接下来 n 行每行两个数 ci，di，描述小 W 的各个装车的时间段。

输出格式：

输出一行，一个正整数，表示两人可以聊多长时间。

样例输入：

4

1 3

5 6

9 13

14 15

2 4

5 7

10 11

13 14

样例输出：

3

数据规模和约定：

对于所有的评测用例，1 ≤ n ≤ 2000, ai < bi < ai+1，ci < di < ci+1,对于所有的 i(1 ≤ i ≤ n)有，1 ≤ ai，bi，ci，di ≤ 1000000。

给两个人设定两个数组 t[i]，t1[i] 当装车时置 1，当 t[i]=t1[i]时总数 sum++。

附加题：习题集 2.24 2.29 2.30

《数据结构》 第 3 次上机题（线性表复习，栈与队列练习）

1. 编程实现书 P32 ADT Stack 基本操作 9 个，用顺序存储结构实现；

2. 编程实现书 P48 ADT Queue 基本操作 9 个，用链式存储结构实现；

3. 利用栈操作实现八皇后问题求解。

4. CSP 题目

题目描述：

顿顿评估了 m 位同学上学期的安全指数，其中第 i（1≤i≤m）位同学的安全指数为 y_i，是一个[0,108]范围内的整数；同时，该同学上学期的挂科情况记作 $result_i \in 0,1$，其中 0 表示挂科、1 表示未挂科。

相应地，顿顿用 $predict_\theta(y)$ 表示根据阈值 θ 将安全指数 y 转化为的具体预测结果。

如果 $predict_\theta(y_j)$ 与 $result_j$ 相同，则说明阈值为 θ 时顿顿对第 j 位同学是否挂科预测正确；不同则说明预测错误。

$$\text{predict}_\theta(y) = \begin{cases} 0 & (y < \theta) \\ 1 & (y \geq \theta) \end{cases}$$

最后，顿顿设计了如下公式来计算最佳阈值 θ^*：

$$\theta^* = \max \operatorname*{argmax}_{\theta \in y_i} \sum_{j=1}^{m} (\text{predict}_\theta(y_j) == result_j)$$

该公式亦可等价地表述为如下规则：

1. 最佳阈值仅在 y_i 中选取，即与某位同学的安全指数相同；
2. 按照该阈值对这 m 位同学上学期的挂科情况进行预测，预测正确的次数最多（即准确率最高）；
3. 多个阈值均可以达到最高准确率时，选取其中最大的。

输入格式：

从标准输入读入数据。

输入的第一行包含一个正整数 m。

接下来输入 m 行，其中第 i（1≤i≤m）行包括用空格分隔的两个整数 y_i 和 $result_i$，含义如上文所述。

输出格式：

输出到标准输出。

输出一个整数，表示最佳阈值 θ^*。

样例 1 输入：

6

0 0

1 0

1 1

3 1

5 1

7 1

样例 1 输出：

3

样例 1 解释：

按照规则一，最佳阈值的选取范围为 0,1,3,5,7。

θ=0 时，预测正确次数为 4；

θ=1 时，预测正确次数为 5；

θ=3 时，预测正确次数为 5；

θ=5 时，预测正确次数为 4；

θ=7 时，预测正确次数为 3。

阈值选取为 1 或 3 时，预测准确率最高；

所以按照规则二，最佳阈值的选取范围缩小为 1,3。

依规则三，θ*=max1,3=3。

样例 2 输入：

8

5 1

5 0

5 0

2 1

3 0

4 0

100000000 1

1 0

样例 2 输出：

100000000

子任务：

70%的测试数据保证 m≤200；

全部的测试数据保证 2≤m≤10^5。

5. CSP 题目

问题描述：

在某图形操作系统中,有 N 个窗口,每个窗口都是一个两边与坐标轴分别平行的矩形区域。窗口的边界上的点也属于该窗口。窗口之间有层次的区别,在多于一个窗口重叠的区域里,只会显示位于顶层的窗口里的内容。

当你点击屏幕上一个点的时候,你就选择了处于被点击位置的最顶层窗口,并且这个窗口就会被移到所有窗口的最顶层,而剩余的窗口的层次顺序不变。如果你点击的位置不属于任何窗口,则系统会忽略你这次点击。

现在我们希望你写一个程序模拟点击窗口的过程。

输入格式：

输入的第一行有两个正整数,即 N 和 M。(1 ≤ N ≤ 10,1 ≤ M ≤ 10)

接下来 N 行按照从最下层到最顶层的顺序给出 N 个窗口的位置。每行包含四个非负整数 x_1, y_1, x_2, y_2,表示该窗口的一对顶点坐标分别为 (x_1, y_1) 和 (x_2, y_2)。保证 $x_1<x_2$, $y_1<y_2$。

接下来 M 行每行包含两个非负整数 x, y,表示一次鼠标点击的坐标。

题目中涉及到的所有点和矩形的顶点的 x, y 坐标分别不超过 2559 和 1439。

问题分析：这个问题可以用链式线性表来实现。

输出格式：输出包括 M 行,每一行表示一次鼠标点击的结果。如果该次鼠标点击选择了一个窗口,则输出这个窗口的编号(窗口按照输入中的顺序从 1 编号到 N);如果没有,则输出"IGNORED"(不含双引号)。

样例输入：

3 4

0 0 4 4

1 1 5 5

2 2 6 6

1 1

0 0

4 4

0 5

样例输出：

2

1

1

IGNORED

样例说明：

第一次点击的位置同时属于第 1 和第 2 个窗口,但是由于第 2 个窗口在上面,它被选择并且被置于顶层。

第二次点击的位置只属于第 1 个窗口,因此该次点击选择了此窗口并将其置于顶层。现在的三个窗口的层次关系与初始状态恰好相反了。

第三次点击的位置同时属于三个窗口的范围,但是由于现在第 1 个窗口处于顶层,它被选择。

最后点击的 (0, 5) 不属于任何窗口。

附加题：习题集 2.32 2.37 2.38

《数据结构》 第 4 次上机题（线性结构练习）

1. 输入稀疏矩阵，建立稀疏矩阵三元组顺序结构，实现矩阵的列序遍历转置和快速转置算法。

2. 求矩阵的马鞍点。（书 P69 7）

3. CSP 题目

问题描述：

一次放学的时候，小明已经规划好了自己回家的路线，并且能够预测经过各个路段的时间。同时，小明通过学校里安装的“智慧光明”终端，看到了**出发时刻路上**经过的所有红绿灯的指示状态。请帮忙计算小明此次回家所需要的时间。

输入格式：

输入的第一行包含空格分隔的三个正整数 r、y、g，表示红绿灯的设置。这三个数均不超过 10^6。

输入的第二行包含一个正整数 n，表示小明总共经过的道路段数和路过的红绿灯数目。

接下来的 n 行，每行包含空格分隔的两个整数 k、t。k=0 表示经过了一段道路，将会耗时 t 秒，此处 t 不超过 10^6；k=1、2、3 时，分别表示**出发时刻**，此处的红绿灯状态是红灯、黄灯、绿灯，且倒计时显示牌上显示的数字是 t，此处 t 分别不会超过 r、y、g。

输出格式：

输出一个数字，表示此次小明放学回家所用的时间。

样例输入：

30 3 30

8

0 10

1 5

0 11

2 2

0 6

0 3

3 10

0 3

样例输出：

46

样例说明：

小明先经过第一段路，用时 10 秒。第一盏红绿灯出发时是红灯，还剩 5 秒；小明到达路口时，这个红绿灯已经变为绿灯，不用等待直接通过。接下来经过第二段路，用时 11 秒。第二盏红绿灯出发时是黄灯，还剩两秒；小明到达路口时，这个红绿灯已经变为红灯，还剩 11 秒。接下来经过第三、第四段路，用时 9 秒。第三盏红绿灯出发时是绿灯，还剩 10 秒；小明到达路口时，这个红绿灯已经变为红灯，还剩两秒。接下来经过最后一段路，用时 3 秒。共计 10+11+11+9+2+3=46 秒。

评测用例规模与约定：

有些测试点具有特殊的性质：

* 前 2 个测试点中不存在任何信号灯。

测试点的输入数据规模：

* 前 6 个测试点保证 $n \leqslant 10^3$。

* 所有测试点保证 $n \leqslant 10^5$。

4. CSP 题目

问题描述：请实现一个铁路购票系统的简单座位分配算法，来处理一节车厢的座位分配。

假设一节车厢有 20 排、每一排 5 个座位。为方便起见，我们用 1 到 100 来给所有的座位编号，第一排是 1 到 5 号，第二排是 6 到 10 号，依次类推，第 20 排是 96 到 100 号。

购票时，一个人可能购一张或多张票，最多不超过 5 张。如果这几张票可以安排在同一排编号相邻的座位，则应该安排在编号最小的相邻座位。否则应该安排在编号最小的几个空座位中（不考虑是否相邻）。

假设初始时车票全部未被购买，现在给了一些购票指令，请你处理这些指令。

输入格式：

对于所有评测用例，$1 \leqslant n \leqslant 100$，所有购票数量之和不超过 100。

输入的第一行包含一个整数 n，表示购票指令的数量。

第二行包含 n 个整数，每个整数 p 在 1 到 5 之间，表示要购入的票数，相邻的两个数之间使用一个空格分隔。

输出格式：

输出 n 行，每行对应一条指令的处理结果。

对于购票指令 p，输出 p 张车票的编号，按从小到大排序。

问题分析：

这个问题可以用顺序结构或链式结构实现。

样例输入：

4

2 5 4 2

样例输出：

1 2

6 7 8 9 10

11 12 13 14

3 4

附加题：习题集 3.20 3.28 3.32

第一阶段总结

第二阶段（树与图部分）

《数据结构》 第 5 次上机题目 （二叉树练习 ）

1. 编程实现书 P75 ADT BinaryTree 基本操作 20 个，用二叉链表结构实现；

2.实现二叉树的先序、中序、后序遍历，用递归和非递归方法；实现层次遍历。

3. 设二叉树采用二叉链表存储,编写函数,对二叉树中每个元素值为 x 的结点，删除以它为根的子树，并释放相应空间。（习题集 6.45）

4. 编写函数，判断给定的二叉树是否是完全二叉树。（习题集 6.49）

5. CSP 题目

问题描述：

待处理的灰度图像长宽皆为 n 个像素，可以表示为一个 n×n 大小的矩阵 A，其中每个元素是一个[0,L)范围内的整数，表示对应位置像素的灰度值。

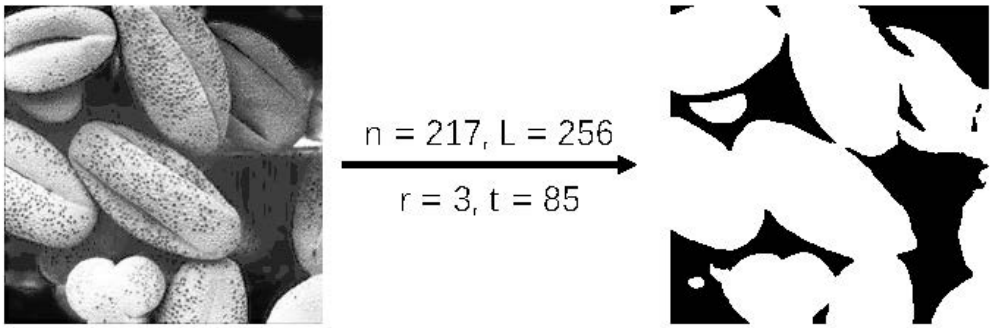
对于矩阵中任意一个元素 A_{ij}（0≤i,j<n），其**邻域**定义为附近若干元素的集和：

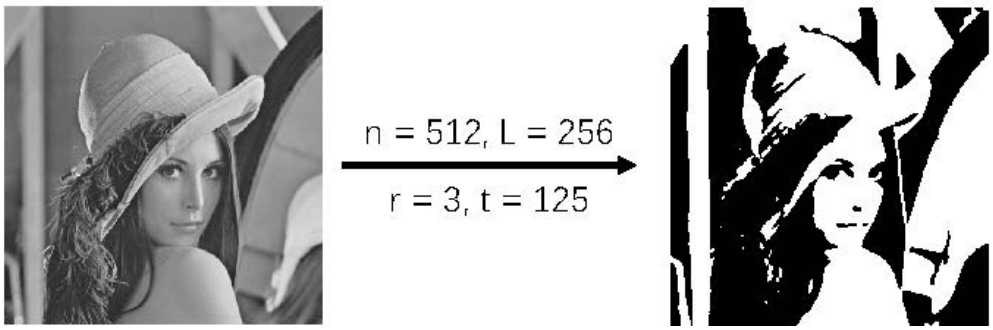
$$Neighbor(i,j,r) = \{A_{xy} | 0 \le x,y < n \text{ and } |x-i| \le r \text{ and } |y-j| \le r\}$$

这里使用了一个额外的参数 r 来指明 A_{ij} 附近元素的具体范围。根据定义，易知 Neighbor(i,j,r)最多有$(2r+1)^2$个元素。

如果元素 A_{ij} **邻域**中所有元素的**平均值**小于或等于一个给定的阈值 t，我们就认为该元素对应位置的像素处于**较暗区域**。

下图给出了两个例子，左侧图像的较暗区域在右侧图像中展示为黑色，其余区域展示为白色。





现给定邻域参数 r 和阈值 t，试统计输入灰度图像中有多少像素处于**较暗区域**。

输入格式：

输入共 n+1 行。

输入的第一行包含四个用空格分隔的正整数 n、L、r 和 t，含义如前文所述。

第二到第 n+1 行输入矩阵 A。

第 i+2（0≤i<n）行包含用空格分隔的 n 个整数，依次为 Ai0,Ai1,…,Ai(n−1)。

输出格式：

输出一个整数，表示输入灰度图像中处于较暗区域的像素总数。

样例输入：

4 16 1 6

0 1 2 3

4 5 6 7

8 9 10 11

12 13 14 15

样例输出：

7

样例输入：

11 8 2 2

0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0

0 7 0 0 0 7 0 0 7 7 0

7 0 7 0 7 0 7 0 7 0 7

7 0 0 0 7 0 0 0 7 0 7

7 0 0 0 0 7 0 0 7 7 0

7 0 0 0 0 0 7 0 7 0 0

7 0 7 0 7 0 7 0 7 0 0

0 7 0 0 0 7 0 0 7 0 0

0 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0

样例输出：

83

评测用例规模与约定：

70% 的测试数据满足 n≤100、r≤10。

全部的测试数据满足 0<n≤600、0<r≤100 且 2≤t<L≤256。

附加题：习题集 6.42 6.45 6.47

《数据结构》 第 6 次上机题目 （ 二叉树、树、图练习 ）

1. 编程实现书 P96 ADT Graph 基本操作 11 个，用邻接矩阵存储结构实现；

2. 输入 N 个权值（1-100 正整数），建立哈夫曼树。

3. 编写函数，对二叉链表结构的二叉树，求宽度。（书 P94 4）

4. 编写函数，对一棵以孩子-兄弟链表表示的树，输出第 i 层的所有元素。

5. CSP 题目

问题描述：

俄罗斯方块是俄罗斯人阿列克谢·帕基特诺夫发明的一款休闲游戏。

游戏在一个 15 行 10 列的方格图上进行，方格图上的每一个格子可能已经放置了方块，或者没有放置方块。每一轮，都会有一个新的由 4 个小方块组成的板块从方格图的上方落下，玩家可以操作板块左右移动放到合适的位置，当板块中某一个方块的下边缘与方格图上的方块上边缘重合或者达到下边界时，板块不再移动，如果此时方格图的某一行全放满了方块，则该行被消除并得分。

在这个问题中，你需要写一个程序来模拟板块下落，你不需要处理玩家的操作，也不需要处理消行和得分。

具体的，给定一个初始的方格图，以及一个板块的形状和它下落的初始位置，你要给出最终的方格图。

输入格式：

输入的前 15 行包含初始的方格图，每行包含 10 个数字，相邻的数字用空格分隔。如果一个数字是 0，表示对应的方格中没有方块，如果数字是 1，则表示初始的时候有方块。输入保证前 4 行中的数字都是 0。

输入的第 16 至第 19 行包含新加入的板块的形状，每行包含 4 个数字，组成了板块图案，同样 0 表示没方块，1 表示有方块。输入保证板块的图案中正好包含 4 个方块，且 4 个方块是连在一起的（准确的说，4 个方块是四连通的，即给定的板块是俄罗斯方块的标准板块）。

第 20 行包含一个 1 到 7 之间的整数，表示板块图案最左边开始的时候是在方格图的哪一列中。注意，这里的板块图案指的是 16 至 19 行所输入的板块图案，如果板块图案的最左边一列全是 0，则它的左边和实际所表示的板块的左边是不一致的（见样例）

输出格式：

输出 15 行，每行 10 个数字，相邻的数字之间用一个空格分隔，表示板块下落后的方格图。注意，你不需要处理最终的消行。

样例输入：

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 1 0 0

0 0 0 0 0 0 1 0 0 0

0 0 0 0 0 0 1 0 0 0

1 1 1 0 0 0 1 1 1 1

0 0 0 0 1 0 0 0 0 0

0 0 0 0

0 1 1 1

0 0 0 1

0 0 0 0

3

样例输出：

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 1 0 0

0 0 0 0 0 0 1 0 0 0

0 0 0 0 0 0 1 0 0 0

1 1 1 1 1 1 1 1 1 1

0 0 0 0 1 1 0 0 0 0

附加题：习题集 6.49 6.60 6.62

《数据结构》 第 7 次上机题目 （ 图 练习 ）

1. 图的深度优先和广度优先遍历；

2. 编程实现 Dijkstra 算法；

3. CSP 题目

题目描述：

小明在他的果园里种了一些苹果树，这些苹果树排列成一个圆。为了保证苹果的品质，在种植过程中要进行疏果操作。为了更及时地完成疏果操作，小明会不时地检查每棵树的状态，根据需要进行疏果。检查时，如果发现可能有苹果从树上掉落，小明会重新统计树上的苹果个数（然后根据之前的记录就可以判断是否有苹果掉落了）。在全部操作结束后，请帮助小明统计相关的信息。

输入格式：

从标准输入读入数据。

第 1 行包含一个正整数 N，表示苹果树的棵数。

第 1+i 行（$1 \leqslant i \leqslant N$），每行的格式为 $m_i, a_{i1}, a_{i2}, \ldots, a_{imi}$，。其中，第一个正整数 m_i 表示本行后面的整数个数。后续的 m_i 个整数表示小明对第 i 棵苹果树的操作记录。若 a_{ij}（$1 \leqslant j \leqslant m_i$）为正整数，则表示小明进行了重新统计该棵树上

的苹果个数的操作，统计的苹果个数为 a_{ij};若为零或负整数，则表示一次疏果操作，去掉的苹果个数是$|a_{ij}|$。

输入保证一定是正确的，满足:

a. $a_{i1}>0$，即对于每棵树的记录，第一个操作一定是统计苹果个数（初始状态，此时不用判断是否有苹果掉落）;

b. 每次疏果操作保证操作后树上的苹果个数仍为正。

输出格式：

输出到标准输出。

输出只有一行，包含三个整数 T、D、E。其中，

a. T 为全部疏果操作结束后所有苹果树上剩下的苹果总数（假设每棵苹果树在最后一次统计苹果个数操作后苹果不会因为疏果以外的原因减少）;

b. D 为发生苹果掉落的苹果树的棵数;

c. E 为相邻连续三棵树发生苹果掉落情况的组数。

对于第三个统计量的解释：N 棵苹果树 $A_1,A_2,...,A_N$排列成一个圆，那么 A_1与 A_2相邻，A_2与 A_3相邻，.....，A_{N-1}与 A_N相邻，A_N与 A_1相邻。如果 A_{i-1}，A_i，A_{i+1}这三棵树都发生了苹果掉落的情况，则记为一组。形式化的，有

$$E = \left|\{A_i|\text{Drop}(\text{Pred}(A_i)) \wedge \text{Drop}(A_i) \wedge \text{Drop}(\text{Succ}(A_i))\}\right|.$$

其中，Drop(A_i）表示苹果树 A_i是否发生苹果掉落的情况，Pred(A_i)表示 A_i的前一棵树 A_{i-1}（如果 i>1）或者 A_N（如果 i=1)，Succ(A_i)表示 A_i的后一棵树 A_{i+1}（如果 i<N）或者 A_1（如果 i=N）

样例 1 输入：

```
4
4 74 -7 -12 -5
5 73 -8 -6 59 -4
5 76 -5 -10 60 -2
5 80 -6 -15 59 0
```

样例 1 输出：

```
222 1 0
```

样例 1 解释：

全部操作结束后，第 1 棵树上剩下的苹果个数为 74-7-12-5=50，第 2 棵为 59-4=55，第 3 棵为 60-2=58，第 4 棵为 59-0=59。因此 T=50+55+58+59=222。

其中，第 3 棵树在第 2 次统计之前剩下的苹果个数为 76-5-10=61>60，因此发生了苹果掉落的情况。可以检验其他的树没有这种情况，因此 D=1。

没有连续三棵树都发生苹果掉落的情况，因此 E=0。

样例 2 输入：

5

4 10 0 9 0

4 10 -2 7 0

2 10 0

4 10 -3 5 0

4 10 -1 8 0

样例 2 输出：

39 4 2

样例 2 解释：

第 1、2、4、5 棵树发生了苹果掉落的情况，因此 D=4。其中，连续三棵树都发生苹果掉落情况的有(5,1,2)和(4,5,1)，因此 E=2。

4. CSP 题目

问题描述：小刘承包了很多片麦田，为了灌溉这些麦田，小刘在第一个麦田挖了一口很深的水井，所有的麦田都从这口井来引水灌溉。 为了灌溉，小刘需要建立一些水渠，以连接水井和麦田，小刘也可以利用部分麦田作为“中转站”，利用水渠连接不同的麦田，这样只要一片麦田能被灌溉，则与其连接的麦田也能被灌溉。现在小刘知道哪些麦田之间可以建设水渠和建设每个水渠所需要的费用（注意不是所有麦田之间都可以建立水渠）。请问灌溉所有麦田最少需要多少费用来修建水渠。

输入格式：输入的第一行包含两个正整数 n， m，分别表示麦田的片数和小刘可以建立的水渠的数量。麦田使用 1，2，3，……依次标号。接下来 m 行，每行包含三个整数 ai， bi， ci，表示第 ai 片麦田与第 bi 片麦田之间可以建立一条水渠，所需要的费用为 ci。

输出格式：输出一个整数，表示灌溉所有麦田所需要的最小费用，及水渠连接说明。

问题分析：这个问题可以用最小生成树算法实现。

输入样例:

4 4

1 2 1

2 3 4

2 4 2

3 4 3

输出样例:

6

建立以下 3 条水渠：麦田 1 与麦田 2、麦田 2 与麦田 4、麦田 4 与麦田 3。

附加题：习题集 7.27 7.34 7.37

第二阶段总结

第三阶段（查找与排序部分）

《数据结构》 第 8 次上机题目 （ 查找 排序 练习 ）

1. 实现二叉排序树的插入和删除。

2. 实现交换、选择、归并等简单排序算法；

3. 实现快速排序算法；

4. 实现堆排序算法；

5. CSP 题目

题目背景：

开学了，可是校园里堆积了不少垃圾杂物。

热心的同学们纷纷自发前来清理，为学校注入正能量～

题目描述：

通过无人机航拍我们已经知晓了 n 处尚待清理的垃圾位置，其中第 i (1≤i≤n) 处的坐标为(xi,yi)，保证所有的坐标均为整数。

我们希望在垃圾集中的地方建立些回收站。具体来说，对于一个位置(x,y)是否

适合建立回收站，我们主要考虑以下几点：

a. (x,y)必须是整数坐标，且该处存在垃圾；

b. 上下左右四个邻居位置，即(x,y+1)、(x,y-1)、(x+1,y)和(x-1,y)处，必须全部存在垃圾；

c. 进一步地，我们会对满足上述两个条件的选址进行评分分数为不大于 4 的自然数，表示在(x±1,y±1)四个对角位置中有几处存在垃圾。

现在，请你统计一下每种得分的选址个数。

输入格式：

从标准输入读入数据。

输入总共有 n+1 行。

第 1 行包含一个正整数 n，表示已查明的垃圾点个数。

第 1+i 行（1≤i≤n）包含由一个空格分隔的两个整数 xi 和 yi，表示第 i 处垃圾的坐标。

保证输入的 n 个坐标互不相同。

输出格式：

输出到标准输出。

输出共五行，每行一个整数，依次表示得分为 0、1、2、3 和 4 的回收站选址个数。

样例 1 输入：

```
7
1 2
2 1
0 0
1 1
1 0
2 0
0 1
```

样例 1 输出：

```
0
```

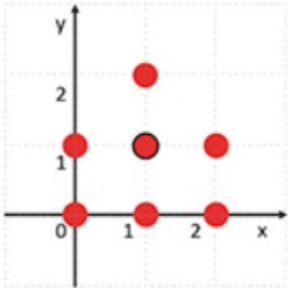
0

1

0

0

样例 1 解释：



如图所示，仅有(1,1)可选为回收站地址，评分为 2。

样例 2 输入：

2

0 0

-100000 10

样例 2 输出：

0

0

0

0

0

样例 2 解释：

不存在可选地址。

样例 3 输入：

11

9 10

10 10

11 10

12 10

13 10

11 9

11 8

12 9

10 9

10 11

12 11

样例 3 输出：

0

2

1

0

0

样例 3 解释：

1 分选址：(10,10)和(12,10)；

2 分选址：(11,9)。

提示：

本题中所涉及的坐标皆为整数，且保证输入的坐标两两不同。

附加题：习题集 9.32 10.32 10.34

第三阶段总结