I’m stuck! **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201312-5 |
| 试题名称： | I’m stuck! |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定一个R行C列的地图，地图的每一个方格可能是'#', '+', '-', '|', '.', 'S', 'T'七个字符中的一个，分别表示如下意思： 　　'#': 任何时候玩家都不能移动到此方格； 　　'+': 当玩家到达这一方格后，下一步可以向上下左右四个方向相邻的任意一个非'#'方格移动一格； 　　'-': 当玩家到达这一方格后，下一步可以向左右两个方向相邻的一个非'#'方格移动一格； 　　'|': 当玩家到达这一方格后，下一步可以向上下两个方向相邻的一个非'#'方格移动一格； 　　'.': 当玩家到达这一方格后，下一步只能向下移动一格。如果下面相邻的方格为'#'，则玩家不能再移动； 　　'S': 玩家的初始位置，地图中只会有一个初始位置。玩家到达这一方格后，下一步可以向上下左右四个方向相邻的任意一个非'#'方格移动一格； 　　'T': 玩家的目标位置，地图中只会有一个目标位置。玩家到达这一方格后，可以选择完成任务，也可以选择不完成任务继续移动。如果继续移动下一步可以向上下左右四个方向相邻的任意一个非'#'方格移动一格。 　　此外，玩家不能移动出地图。 　　请找出满足下面两个性质的方格个数： 　　1. 玩家可以从初始位置移动到此方格； 　　2. 玩家不可以从此方格移动到目标位置。  **输入格式**  　　输入的第一行包括两个整数R 和C，分别表示地图的行和列数。(1 ≤ R, C ≤ 50)。 　　接下来的R行每行都包含C个字符。它们表示地图的格子。地图上恰好有一个'S'和一个'T'。  **输出格式**  　　如果玩家在初始位置就已经不能到达终点了，就输出“I'm stuck!”（不含双引号）。否则的话，输出满足性质的方格的个数。  **样例输入**  5 5 --+-+ ..|#. ..|## S-+-T ####.  **样例输出**  2  **样例说明**  　　如果把满足性质的方格在地图上用'X'标记出来的话，地图如下所示： 　　--+-+ 　　..|#X 　　..|## 　　S-+-T 　　####X |

有趣的数 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201312-4 |
| 试题名称： | 有趣的数 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　我们把一个数称为有趣的，当且仅当： 　　1. 它的数字只包含0, 1, 2, 3，且这四个数字都出现过至少一次。 　　2. 所有的0都出现在所有的1之前，而所有的2都出现在所有的3之前。 　　3. 最高位数字不为0。 　　因此，符合我们定义的最小的有趣的数是2013。除此以外，4位的有趣的数还有两个：2031和2301。 　　请计算恰好有n位的有趣的数的个数。由于答案可能非常大，只需要输出答案除以1000000007的余数。  **输入格式**  　　输入只有一行，包括恰好一个正整数n (4 ≤ n ≤ 1000)。  **输出格式**  　　输出只有一行，包括恰好n 位的整数中有趣的数的个数除以1000000007的余数。  **样例输入**  4  **样例输出**  3 |

最大的矩形 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201312-3 |
| 试题名称： | 最大的矩形 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在横轴上放了n个相邻的矩形，每个矩形的宽度是1，而第i（1 ≤ i ≤ n）个矩形的高度是hi。这n个矩形构成了一个直方图。例如，下图中六个矩形的高度就分别是3, 1, 6, 5, 2, 3。   http://118.190.20.162/RequireFile.do?fid=Fmhrjgtb 　　请找出能放在给定直方图里面积最大的矩形，它的边要与坐标轴平行。对于上面给出的例子，最大矩形如下图所示的阴影部分，面积是10。 http://118.190.20.162/RequireFile.do?fid=gNe4DHhD  **输入格式**  　　第一行包含一个整数n，即矩形的数量(1 ≤ n ≤ 1000)。 　　第二行包含n 个整数h1, h2, … , hn，相邻的数之间由空格分隔。(1 ≤ hi ≤ 10000)。hi是第i个矩形的高度。  **输出格式**  　　输出一行，包含一个整数，即给定直方图内的最大矩形的面积。  **样例输入**  6 3 1 6 5 2 3  **样例输出**  10 |

ISBN号码 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201312-2 |
| 试题名称： | ISBN号码 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　每一本正式出版的图书都有一个ISBN号码与之对应，ISBN码包括9位数字、1位识别码和3位分隔符，其规定格式如“x-xxx-xxxxx-x”，其中符号“-”是分隔符（键盘上的减号），最后一位是识别码，例如0-670-82162-4就是一个标准的ISBN码。ISBN码的首位数字表示书籍的出版语言，例如0代表英语；第一个分隔符“-”之后的三位数字代表出版社，例如670代表维京出版社；第二个分隔之后的五位数字代表该书在出版社的编号；最后一位为识别码。 　　识别码的计算方法如下： 　　首位数字乘以1加上次位数字乘以2……以此类推，用所得的结果mod 11，所得的余数即为识别码，如果余数为10，则识别码为大写字母X。例如ISBN号码0-670-82162-4中的识别码4是这样得到的：对067082162这9个数字，从左至右，分别乘以1，2，…，9，再求和，即0×1+6×2+……+2×9=158，然后取158 mod 11的结果4作为识别码。 　　编写程序判断输入的ISBN号码中识别码是否正确，如果正确，则仅输出“Right”；如果错误，则输出是正确的ISBN号码。  **输入格式**  　　输入只有一行，是一个字符序列，表示一本书的ISBN号码（保证输入符合ISBN号码的格式要求）。  **输出格式**  　　输出一行，假如输入的ISBN号码的识别码正确，那么输出“Right”，否则，按照规定的格式，输出正确的ISBN号码（包括分隔符“-”）。  **样例输入**  0-670-82162-4  **样例输出**  Right  **样例输入**  0-670-82162-0  **样例输出**  0-670-82162-4 |

出现次数最多的数 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201312-1 |
| 试题名称： | 出现次数最多的数 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定n个正整数，找出它们中出现次数最多的数。如果这样的数有多个，请输出其中最小的一个。  **输入格式**  　　输入的第一行只有一个正整数n(1 ≤ n ≤ 1000)，表示数字的个数。 　　输入的第二行有n个整数s1, s2, …, sn (1 ≤ si ≤ 10000, 1 ≤ i ≤ n)。相邻的数用空格分隔。  **输出格式**  　　输出这n个次数中出现次数最多的数。如果这样的数有多个，输出其中最小的一个。  **样例输入**  6 10 1 10 20 30 20  **样例输出**  10 |

任务调度 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201403-5 |
| 试题名称： | 任务调度 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　有若干个任务需要在一台机器上运行。它们之间没有依赖关系,因此 可以被按照任意顺序执行。 　　该机器有两个 CPU 和一个 GPU。对于每个任务,你可以为它分配不 同的硬件资源: 　　1. 在单个 CPU 上运行。 　　2. 在两个 CPU 上同时运行。 　　3. 在单个 CPU 和 GPU 上同时运行。 　　4. 在两个 CPU 和 GPU 上同时运行。 　　一个任务开始执行以后,将会独占它所用到的所有硬件资源,不得中 断,直到执行结束为止。第 i 个任务用单个 CPU,两个 CPU,单个 CPU 加 GPU,两个 CPU 加 GPU 运行所消耗的时间分别为 ai,bi,ci 和 di。 　　现在需要你计算出至少需要花多少时间可以把所有给定的任务完成。  **输入格式**  　　输入的第一行只有一个正整数 n(1 ≤ n ≤ 40), 是总共需要执行的任 务个数。 　　接下来的 n 行每行有四个正整数 ai, bi, ci, di(ai, bi, ci, di 均不超过 10), 以空格隔开。  **输出格式**  　　输出只有一个整数,即完成给定的所有任务所需的最少时间。  **样例输入**  3 4 4 2 2 7 4 7 4 3 3 3 3  **样例输出**  7  **样例说明**  　　有很多种调度方案可以在 7 个时间单位里完成给定的三个任务,以下是其中的一种方案: 　　同时运行第一个任务(单 CPU 加上 GPU)和第三个任务(单 CPU), 它们分别在时刻 2 和时刻 3 完成。在时刻 3 开始双 CPU 运行任务 2,在 时刻 7 完成。 |

无线网络 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201403-4 |
| 试题名称： | 无线网络 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　目前在一个很大的平面房间里有 n 个无线路由器,每个无线路由器都固定在某个点上。任何两个无线路由器只要距离不超过 r 就能互相建立网络连接。 　　除此以外,另有 m 个可以摆放无线路由器的位置。你可以在这些位置中选择至多 k 个增设新的路由器。 　　你的目标是使得第 1 个路由器和第 2 个路由器之间的网络连接经过尽量少的中转路由器。请问在最优方案下中转路由器的最少个数是多少?  **输入格式**  　　第一行包含四个正整数 n,m,k,r。(2 ≤ n ≤ 100,1 ≤ k ≤ m ≤ 100, 1 ≤ r ≤ 108)。 　　接下来 n 行,每行包含两个整数 xi 和 yi,表示一个已经放置好的无线 路由器在 (xi, yi) 点处。输入数据保证第 1 和第 2 个路由器在仅有这 n 个路由器的情况下已经可以互相连接(经过一系列的中转路由器)。 　　接下来 m 行,每行包含两个整数 xi 和 yi,表示 (xi, yi) 点处可以增设 一个路由器。 　　输入中所有的坐标的绝对值不超过 108,保证输入中的坐标各不相同。  **输出格式**  　　输出只有一个数,即在指定的位置中增设 k 个路由器后,从第 1 个路 由器到第 2 个路由器最少经过的中转路由器的个数。  **样例输入**  5 3 1 3 0 0 5 5 0 3 0 5 3 5 3 3 4 4 3 0  **样例输出**  2 |

命令行选项 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201403-3 |
| 试题名称： | 命令行选项 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　请你写一个命令行分析程序,用以分析给定的命令行里包含哪些选项。每个命令行由若干个字符串组成,它们之间恰好由一个空格分隔。这些字符串中的第一个为该命令行工具的名字,由小写字母组成,你的程序不用对它进行处理。在工具名字之后可能会包含若干选项,然后可能会包含一 些不是选项的参数。 　　选项有两类:带参数的选项和不带参数的选项。一个合法的无参数选项的形式是一个减号后面跟单个小写字母,如"-a" 或"-b"。而带参数选项则由两个由空格分隔的字符串构成,前者的格式要求与无参数选项相同,后者则是该选项的参数,是由小写字母,数字和减号组成的非空字符串。 　　该命令行工具的作者提供给你一个格式字符串以指定他的命令行工具需要接受哪些选项。这个字符串由若干小写字母和冒号组成,其中的每个小写字母表示一个该程序接受的选项。如果该小写字母后面紧跟了一个冒号,它就表示一个带参数的选项,否则则为不带参数的选项。例如, "ab:m:" 表示该程序接受三种选项,即"-a"(不带参数),"-b"(带参数), 以及"-m"(带参数)。 　　命令行工具的作者准备了若干条命令行用以测试你的程序。对于每个命令行,你的工具应当一直向后分析。当你的工具遇到某个字符串既不是合法的选项,又不是某个合法选项的参数时,分析就停止。命令行剩余的未分析部分不构成该命令的选项,因此你的程序应当忽略它们。  **输入格式**  　　输入的第一行是一个格式字符串,它至少包含一个字符,且长度不超过 52。格式字符串只包含小写字母和冒号,保证每个小写字母至多出现一次,不会有两个相邻的冒号,也不会以冒号开头。 　　输入的第二行是一个正整数 N(1 ≤ N ≤ 20),表示你需要处理的命令行的个数。 　　接下来有 N 行,每行是一个待处理的命令行,它包括不超过 256 个字符。该命令行一定是若干个由单个空格分隔的字符串构成,每个字符串里只包含小写字母,数字和减号。  **输出格式**  　　输出有 N 行。其中第 i 行以"Case i:" 开始,然后应当有恰好一个空格,然后应当按照字母升序输出该命令行中用到的所有选项的名称,对于带参数的选项,在输出它的名称之后还要输出它的参数。如果一个选项在命令行中出现了多次,只输出一次。如果一个带参数的选项在命令行中出 现了多次,只输出最后一次出现时所带的参数。  **样例输入**  albw:x 4 ls -a -l -a documents -b ls ls -w 10 -x -w 15 ls -a -b -c -d -e -l  **样例输出**  Case 1: -a -l Case 2: Case 3: -w 15 -x Case 4: -a -b |

窗口 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201403-2 |
| 试题名称： | 窗口 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **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 x1, y1, x2, y2,表示该窗口的一对顶点坐标分别为 (x1, y1) 和 (x2, y2)。保证 x1 < x2,y1 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。  **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。 |

拼图 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201409-5 |
| 试题名称： | 拼图 |
| 时间限制： | 3.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给出一个n×m的方格图，现在要用如下L型的积木拼到这个图中，使得方格图正好被拼满，请问总共有多少种拼法。其中，方格图的每一个方格正好能放积木中的一块。积木可以任意旋转。 http://118.190.20.162/RequireFile.do?fid=DNHB9nN9  **输入格式**  　　输入的第一行包含两个整数n, m，表示方格图的大小。  **输出格式**  　　输出一行，表示可以放的方案数，由于方案数可能很多，所以请输出方案数除以1,000,000,007的余数。  **样例输入**  6 2  **样例输出**  4  **样例说明**  　　四种拼法如下图所示： http://118.190.20.162/RequireFile.do?fid=2yqET7m8  **评测用例规模与约定**  　　在评测时将使用10个评测用例对你的程序进行评测。 　　评测用例1和2满足：1<=n<=30，m=2。 　　评测用例3和4满足：1<=n, m<=6。 　　评测用例5满足：1<=n<=100，1<=m<=6。 　　评测用例6和7满足：1<=n<=1000，1<=m<=6。 　　评测用例8、9和10满足：1<=n<=10^15，1<=m<=7。 |

相反数 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201403-1 |
| 试题名称： | 相反数 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　有 N 个非零且各不相同的整数。请你编一个程序求出它们中有多少对相反数(a 和 -a 为一对相反数)。  **输入格式**  　　第一行包含一个正整数 N。(1 ≤ N ≤ 500)。 　　第二行为 N 个用单个空格隔开的非零整数,每个数的绝对值不超过1000,保证这些整数各不相同。  **输出格式**  　　只输出一个整数,即这 N 个数中包含多少对相反数。  **样例输入**  5 1 2 3 -1 -2  **样例输出**  2 |

最优配餐 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201409-4 |
| 试题名称： | 最优配餐 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　栋栋最近开了一家餐饮连锁店，提供外卖服务。随着连锁店越来越多，怎么合理的给客户送餐成为了一个急需解决的问题。 　　栋栋的连锁店所在的区域可以看成是一个n×n的方格图（如下图所示），方格的格点上的位置上可能包含栋栋的分店（绿色标注）或者客户（蓝色标注），有一些格点是不能经过的（红色标注）。 　　方格图中的线表示可以行走的道路，相邻两个格点的距离为1。栋栋要送餐必须走可以行走的道路，而且不能经过红色标注的点。  http://118.190.20.162/RequireFile.do?fid=383qHJjQ 　　送餐的主要成本体现在路上所花的时间，每一份餐每走一个单位的距离需要花费1块钱。每个客户的需求都可以由栋栋的任意分店配送，每个分店没有配送总量的限制。 　　现在你得到了栋栋的客户的需求，请问在最优的送餐方式下，送这些餐需要花费多大的成本。  **输入格式**  　　输入的第一行包含四个整数n, m, k, d，分别表示方格图的大小、栋栋的分店数量、客户的数量，以及不能经过的点的数量。 　　接下来m行，每行两个整数xi, yi，表示栋栋的一个分店在方格图中的横坐标和纵坐标。 　　接下来k行，每行三个整数xi, yi, ci，分别表示每个客户在方格图中的横坐标、纵坐标和订餐的量。（注意，可能有多个客户在方格图中的同一个位置） 　　接下来d行，每行两个整数，分别表示每个不能经过的点的横坐标和纵坐标。  **输出格式**  　　输出一个整数，表示最优送餐方式下所需要花费的成本。  **样例输入**  10 2 3 3 1 1 8 8 1 5 1 2 3 3 6 7 2 1 2 2 2 6 8  **样例输出**  29  **评测用例规模与约定**  　　前30%的评测用例满足：1<=n <=20。 　　前60%的评测用例满足：1<=n<=100。 　　所有评测用例都满足：1<=n<=1000，1<=m, k, d<=n^2。可能有多个客户在同一个格点上。每个客户的订餐量不超过1000，每个客户所需要的餐都能被送到。 |

画图 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201409-2 |
| 试题名称： | 画图 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在一个定义了直角坐标系的纸上，画一个(x1,y1)到(x2,y2)的矩形指将横坐标范围从x1到x2，纵坐标范围从y1到y2之间的区域涂上颜色。 　　下图给出了一个画了两个矩形的例子。第一个矩形是(1,1) 到(4, 4)，用绿色和紫色表示。第二个矩形是(2, 3)到(6, 5)，用蓝色和紫色表示。图中，一共有15个单位的面积被涂上颜色，其中紫色部分被涂了两次，但在计算面积时只计算一次。在实际的涂色过程中，所有的矩形都涂成统一的颜色，图中显示不同颜色仅为说明方便。 http://118.190.20.162/RequireFile.do?fid=qMtgAT3j 　　给出所有要画的矩形，请问总共有多少个单位的面积被涂上颜色。  **输入格式**  　　输入的第一行包含一个整数n，表示要画的矩形的个数。 　　接下来n行，每行4个非负整数，分别表示要画的矩形的左下角的横坐标与纵坐标，以及右上角的横坐标与纵坐标。  **输出格式**  　　输出一个整数，表示有多少个单位的面积被涂上颜色。  **样例输入**  2 1 1 4 4 2 3 6 5  **样例输出**  15  **评测用例规模与约定**  　　1<=n<=100，0<=横坐标、纵坐标<=100。 |

字符串匹配 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201409-3 |
| 试题名称： | 字符串匹配 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给出一个字符串和多行文字，在这些文字中找到字符串出现的那些行。你的程序还需支持大小写敏感选项：当选项打开时，表示同一个字母的大写和小写看作不同的字符；当选项关闭时，表示同一个字母的大写和小写看作相同的字符。  **输入格式**  　　输入的第一行包含一个字符串S，由大小写英文字母组成。 　　第二行包含一个数字，表示大小写敏感的选项，当数字为0时表示大小写不敏感，当数字为1时表示大小写敏感。 　　第三行包含一个整数n，表示给出的文字的行数。 　　接下来n行，每行包含一个字符串，字符串由大小写英文字母组成，不含空格和其他字符。  **输出格式**  　　输出多行，每行包含一个字符串，按出现的顺序依次给出那些包含了字符串S的行。  **样例输入**  Hello 1 5 HelloWorld HiHiHelloHiHi GrepIsAGreatTool HELLO HELLOisNOTHello  **样例输出**  HelloWorld HiHiHelloHiHi HELLOisNOTHello  **样例说明**  　　在上面的样例中，第四个字符串虽然也是Hello，但是大小写不正确。如果将输入的第二行改为0，则第四个字符串应该输出。  **评测用例规模与约定**  　　1<=n<=100，每个字符串的长度不超过100。 |

相邻数对 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201409-1 |
| 试题名称： | 相邻数对 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定n个不同的整数，问这些数中有多少对整数，它们的值正好相差1。  **输入格式**  　　输入的第一行包含一个整数n，表示给定整数的个数。 　　第二行包含所给定的n个整数。  **输出格式**  　　输出一个整数，表示值正好相差1的数对的个数。  **样例输入**  6 10 2 6 3 7 8  **样例输出**  3  **样例说明**  　　值正好相差1的数对包括(2, 3), (6, 7), (7, 8)。  **评测用例规模与约定**  　　1<=n<=1000，给定的整数为不超过10000的非负整数。 |

货物调度 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201412-5 |
| 试题名称： | 货物调度 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　某公司要处理一个周期性的物流问题。 　　有n个城市，第i个城市在每周的第j(1≤j≤7) 天会生产aij吨某种货物，同时需要消耗bij吨该种货物。已知每周的产量等于消耗量（即aij之和等于bij之和）。 　　城市之间有m条道路，第k条道路连接了城市sk和tk。一条道路上运输1吨货物有一个固定的成本ck。道路都可以双向使用。每天运输的货物量没有限制。城市之间的距离并不远，货物可以从任意一个城市运输到任意另一个城市并且在当天到达。 　　货物如果在当天没有被消耗掉，就需要存放在仓库里过夜。第i个城市的仓库容量为vi，存放1 吨货物过一夜所需的成本是wi。 　　请你计算该公司如果每周循环性地按照一个固定的流程调度货物的话，该公司在最优方案下每周需要为货物的运输和存储消耗多少成本。  **输入格式**  　　输入的第一行有两个正整数n和m，即城市的个数和道路的条数。 　　接下来有n行，每行包含16个整数，用以描述第i个城市的相关数据。其中第i行包含的数为ai1, ai2, ai3, ai4, ai5, ai6, ai7, bi1, bi2, bi3, bi4, bi5, bi6, bi7, vi, wi。 　　接下来有m行，每行包含3个整数，用以描述一条道路的相关数据。其中第k行包含的数为sk, tk和ck。 　　输入数据中城市的编号均为1到n之间。输入数据的每行的行首行尾均保证没有空格，两个数之间恰好被一个空格隔开。  **输出格式**  　　你只需要输出一个数，即最优方案下每周的支出。  **样例输入**  3 3 0 0 0 0 5 0 0 0 0 0 0 0 0 0 2 4 0 0 0 0 0 0 0 2 0 0 0 0 0 0 2 1 0 0 0 0 0 0 0 0 0 3 0 0 0 0 2 5 1 2 1 1 3 5 2 3 1  **样例输出**  67  **样例说明**  　　城市1 每周五生产5 吨货物，把其中2 吨运到存储费用低廉的城市2 存储，把1 吨运到城市3 存储，剩下的2 吨留在城市1。 　　在次周一的时候城市2 会消耗掉存放在那里的2 吨货物。为了节约存储成本，将囤放在城市1 的货物运到城市2 存放。周三再将所有货物运到城市3 以满足该城市的需求。 　　在此方案下，每周的运输成本为8，每周的存储成本为59，因此每周的总支出为67。  **评测用例规模与约定**  　　对于100%的数据，1≤n≤100，1≤m≤500，0≤aij,bij,vi≤100，1≤wi,ck≤100。 |

最优灌溉 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201412-4 |
| 试题名称： | 最优灌溉 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　雷雷承包了很多片麦田，为了灌溉这些麦田，雷雷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，雷雷需要建立一些水渠，以连接水井和麦田，雷雷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。 　　现在雷雷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雷雷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**  　　输出一行，包含一个整数，表示灌溉所有麦田所需要的最小费用。  **样例输入**  4 4 1 2 1 2 3 4 2 4 2 3 4 3  **样例输出**  6  **样例说明**  　　建立以下三条水渠：麦田1与麦田2、麦田2与麦田4、麦田4与麦田3。  **评测用例规模与约定**  　　前20%的评测用例满足：n≤5。 　　前40%的评测用例满足：n≤20。 　　前60%的评测用例满足：n≤100。 　　所有评测用例都满足：1≤n≤1000，1≤m≤100,000，1≤ci≤10,000。 |

集合竞价 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201412-3 |
| 试题名称： | 集合竞价 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　某股票交易所请你编写一个程序，根据开盘前客户提交的订单来确定某特定股票的开盘价和开盘成交量。 　　该程序的输入由很多行构成，每一行为一条记录，记录可能有以下几种： 　　1. buy p s 表示一个购买股票的买单，每手出价为p，购买股数为s。 　　2. sell p s 表示一个出售股票的卖单，每手出价为p，出售股数为s。 　　3. cancel i表示撤销第i行的记录。 　　如果开盘价为p0，则系统可以将所有出价至少为p0的买单和所有出价至多为p0的卖单进行匹配。因此，此时的开盘成交量为出价至少为p0的买单的总股数和所有出价至多为p0的卖单的总股数之间的较小值。 　　你的程序需要确定一个开盘价，使得开盘成交量尽可能地大。如果有多个符合条件的开盘价，你的程序应当输出最高的那一个。  **输入格式**  　　输入数据有任意多行，每一行是一条记录。保证输入合法。股数为不超过108的正整数，出价为精确到恰好小数点后两位的正实数，且不超过10000.00。  **输出格式**  　　你需要输出一行，包含两个数，以一个空格分隔。第一个数是开盘价，第二个是此开盘价下的成交量。开盘价需要精确到小数点后恰好两位。  **样例输入**  buy 9.25 100 buy 8.88 175 sell 9.00 1000 buy 9.00 400 sell 8.92 400 cancel 1 buy 100.00 50  **样例输出**  9.00 450  **评测用例规模与约定**  　　对于100%的数据，输入的行数不超过5000。 |

Z字形扫描 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201412-2 |
| 试题名称： | Z字形扫描 |
| 时间限制： | 2.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在图像编码的算法中，需要将一个给定的方形矩阵进行Z字形扫描(Zigzag Scan)。给定一个n×n的矩阵，Z字形扫描的过程如下图所示： http://118.190.20.162/RequireFile.do?fid=Nad2EgMG 　　对于下面的4×4的矩阵， 　　1 5 3 9 　　3 7 5 6 　　9 4 6 4 　　7 3 1 3 　　对其进行Z字形扫描后得到长度为16的序列： 　　1 5 3 9 7 3 9 5 4 7 3 6 6 4 1 3 　　请实现一个Z字形扫描的程序，给定一个n×n的矩阵，输出对这个矩阵进行Z字形扫描的结果。  **输入格式**  　　输入的第一行包含一个整数n，表示矩阵的大小。 　　输入的第二行到第n+1行每行包含n个正整数，由空格分隔，表示给定的矩阵。  **输出格式** 输出一行，包含n×n个整数，由空格分隔，表示输入的矩阵经过Z字形扫描后的结果。  **样例输入**  4 1 5 3 9 3 7 5 6 9 4 6 4 7 3 1 3  **样例输出** 1 5 3 9 7 3 9 5 4 7 3 6 6 4 1 3  **评测用例规模与约定**  　　1≤n≤500，矩阵元素为不超过1000的正整数。 |

门禁系统 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201412-1 |
| 试题名称： | 门禁系统 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　海涛最近要负责图书馆的管理工作，需要记录下每天读者的到访情况。每位读者有一个编号，每条记录用读者的编号来表示。给出读者的来访记录，请问每一条记录中的读者是第几次出现。  **输入格式**  　　输入的第一行包含一个整数n，表示涛涛的记录条数。 　　第二行包含n个整数，依次表示涛涛的记录中每位读者的编号。  **输出格式**  　　输出一行，包含n个整数，由空格分隔，依次表示每条记录中的读者编号是第几次出现。  **样例输入**  5 1 2 1 1 3  **样例输出**  1 1 2 3 1  **评测用例规模与约定**  　　1≤n≤1,000，读者的编号为不超过n的正整数。 |

最小花费 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201503-5 |
| 试题名称： | 最小花费 |
| 时间限制： | 4.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　C国共有n个城市。有n-1条双向道路，每条道路连接两个城市，任意两个城市之间能互相到达。小R来到C国旅行，他共规划了m条旅行的路线，第i条旅行路线的起点是si，终点是ti。在旅行过程中，小R每行走一单位长度的路需要吃一单位的食物。C国的食物只能在各个城市中买到，而且不同城市的食物价格可能不同。 　　然而，小R不希望在旅行中为了购买较低价的粮食而绕远路，因此他总会选择最近的路走。现在，请你计算小R规划的每条旅行路线的最小花费是多少。  **输入格式**  　　第一行包含2个整数n和m。 　　第二行包含n个整数。第i个整数wi表示城市i的食物价格。 　　接下来n-1行，每行包括3个整数u, v, e，表示城市u和城市v之间有一条长为e的双向道路。 　　接下来m行，每行包含2个整数si和ti，分别表示一条旅行路线的起点和终点。  **输出格式**  　　输出m行，分别代表每一条旅行方案的最小花费。  **样例输入**  6 4 1 7 3 2 5 6 1 2 4 1 3 5 2 4 1 3 5 2 3 6 1 2 5 4 6 6 4 5 6  **样例输出**  35 16 26 13  **样例说明**  　　对于第一条路线，小R会经过2->1->3->5。其中在城市2处以7的价格购买4单位粮食，到城市1时全部吃完，并用1的价格购买7单位粮食，然后到达终点。  **评测用例规模与约定**  　　前10%的评测用例满足：n, m ≤ 20, wi ≤ 20； 　　前30%的评测用例满足：n, m ≤ 200； 　　另有40%的评测用例满足：一个城市至多与其它两个城市相连。 　　所有评测用例都满足：1 ≤ n, m ≤ 105，1 ≤ wi ≤ 106，1 ≤ e ≤ 10000。 |

网络延时 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201503-4 |
| 试题名称： | 网络延时 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定一个公司的网络，由n台交换机和m台终端电脑组成，交换机与交换机、交换机与电脑之间使用网络连接。交换机按层级设置，编号为1的交换机为根交换机，层级为1。其他的交换机都连接到一台比自己上一层的交换机上，其层级为对应交换机的层级加1。所有的终端电脑都直接连接到交换机上。 　　当信息在电脑、交换机之间传递时，每一步只能通过自己传递到自己所连接的另一台电脑或交换机。请问，电脑与电脑之间传递消息、或者电脑与交换机之间传递消息、或者交换机与交换机之间传递消息最多需要多少步。  **输入格式**  　　输入的第一行包含两个整数n, m，分别表示交换机的台数和终端电脑的台数。 　　第二行包含n - 1个整数，分别表示第2、3、……、n台交换机所连接的比自己上一层的交换机的编号。第i台交换机所连接的上一层的交换机编号一定比自己的编号小。 　　第三行包含m个整数，分别表示第1、2、……、m台终端电脑所连接的交换机的编号。  **输出格式**  　　输出一个整数，表示消息传递最多需要的步数。  **样例输入**  4 2 1 1 3 2 1  **样例输出**  4  **样例说明**  　　样例的网络连接模式如下，其中圆圈表示交换机，方框表示电脑： http://118.190.20.162/RequireFile.do?fid=F9GfBRHL 　　其中电脑1与交换机4之间的消息传递花费的时间最长，为4个单位时间。  **样例输入**  4 4 1 2 2 3 4 4 4  **样例输出**  4  **样例说明**  　　样例的网络连接模式如下： http://118.190.20.162/RequireFile.do?fid=LYDFDEbt 　　其中电脑1与电脑4之间的消息传递花费的时间最长，为4个单位时间。  **评测用例规模与约定**  　　前30%的评测用例满足：n ≤ 5, m ≤ 5。 　　前50%的评测用例满足：n ≤ 20, m ≤ 20。 　　前70%的评测用例满足：n ≤ 100, m ≤ 100。 　　所有评测用例都满足：1 ≤ n ≤ 10000，1 ≤ m ≤ 10000。 |

节日 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201503-3 |
| 试题名称： | 节日 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　有一类节日的日期并不是固定的，而是以“a月的第b个星期c”的形式定下来的，比如说母亲节就定为每年的五月的第二个星期日。 　　现在，给你a，b，c和y1, y2(1850 ≤ y1, y2 ≤ 2050)，希望你输出从公元y1年到公元y2年间的每年的a月的第b个星期c的日期。 　　提示：关于闰年的规则：年份是400的整数倍时是闰年，否则年份是4的倍数并且不是100的倍数时是闰年，其他年份都不是闰年。例如1900年就不是闰年，而2000年是闰年。 　　为了方便你推算，已知1850年1月1日是星期二。  **输入格式**  　　输入包含恰好一行，有五个整数a, b, c, y1, y2。其中c=1, 2, ……, 6, 7分别表示星期一、二、……、六、日。  **输出格式**  　　对于y1和y2之间的每一个年份，包括y1和y2，按照年份从小到大的顺序输出一行。 　　如果该年的a月第b个星期c确实存在，则以"yyyy/mm/dd"的格式输出，即输出四位数的年份，两位数的月份，两位数的日期，中间用斜杠“/”分隔，位数不足时前补零。 　　如果该年的a月第b个星期c并不存在，则输出"none"（不包含双引号)。  **样例输入**  5 2 7 2014 2015  **样例输出**  2014/05/11 2015/05/10  **评测用例规模与约定**  　　所有评测用例都满足：1 ≤ a ≤ 12，1 ≤ b ≤ 5，1 ≤ c ≤ 7，1850 ≤ y1, y2 ≤ 2050。 |

数字排序 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201503-2 |
| 试题名称： | 数字排序 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定n个整数，请统计出每个整数出现的次数，按出现次数从多到少的顺序输出。  **输入格式**  　　输入的第一行包含一个整数n，表示给定数字的个数。 　　第二行包含n个整数，相邻的整数之间用一个空格分隔，表示所给定的整数。  **输出格式**  　　输出多行，每行包含两个整数，分别表示一个给定的整数和它出现的次数。按出现次数递减的顺序输出。如果两个整数出现的次数一样多，则先输出值较小的，然后输出值较大的。  **样例输入**  12 5 2 3 3 1 3 4 2 5 2 3 5  **样例输出**  3 4 2 3 5 3 1 1 4 1  **评测用例规模与约定**  　　1 ≤ n ≤ 1000，给出的数都是不超过1000的非负整数。 |

图像旋转 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201503-1 |
| 试题名称： | 图像旋转 |
| 时间限制： | 5.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　旋转是图像处理的基本操作，在这个问题中，你需要将一个图像逆时针旋转90度。 　　计算机中的图像表示可以用一个矩阵来表示，为了旋转一个图像，只需要将对应的矩阵旋转即可。  **输入格式**  　　输入的第一行包含两个整数n, m，分别表示图像矩阵的行数和列数。 　　接下来n行每行包含m个整数，表示输入的图像。  **输出格式**  　　输出m行，每行包含n个整数，表示原始矩阵逆时针旋转90度后的矩阵。  **样例输入**  2 3 1 5 3 3 2 4  **样例输出**  3 4 5 2 1 3  **评测用例规模与约定**  　　1 ≤ n, m ≤ 1,000，矩阵中的数都是不超过1000的非负整数。 |

最佳文章 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201509-5 |
| 试题名称： | 最佳文章 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　小明最近在研究一门新的语言，叫做Q语言。Q语言单词和文章都可以用且仅用只含有小写英文字母的字符串表示，任何由这些字母组成的字符串也都是一篇合法的Q语言文章。 　　在Q语言的所有单词中，小明选出了他认为最重要的n个。使用这些单词，小明可以评价一篇Q语言文章的“重要度”。 　　文章“重要度”的定义为：在该文章中，所有重要的Q语言单词出现次数的总和。其中多次出现的单词，不论是否发生包含、重叠等情况，每次出现均计算在内。 　　例如，假设n = 2，小明选出的单词是gvagv和agva。在文章gvagvagvagv中，gvagv出现了3次，agva出现了2次，因此这篇文章的重要度为3+2=5。 　　现在，小明想知道，一篇由m个字母组成的Q语言文章，重要度最高能达到多少。  **输入格式**  　　输入的第一行包含两个整数n, m，表示小明选出的单词个数和最终文章包含的字母个数。 　　接下来n行，每行包含一个仅由英文小写字母构成的字符串，表示小明选出的这n个单词。  **输出格式**  　　输出一行一个整数，表示由m个字母组成的Q语言文章中，重要度最高的文章的重要度。  **样例输入**  3 15 agva agvagva gvagva  **样例输出**  11  **样例说明**  　　15个字母组成的重要度最高的文章为gvagvagvagvagva。 　　在这篇文章中，agva出现4次，agvagva出现3次，gvagva出现4次，共计4+3+4=11次。  **评测用例规模与约定**  　　在评测时将使用10个评测用例对你的程序进行评测。 　　设s为构成n个重要单词字母的总个数，例如在样例中，s=4+7+6=17；a为构成n个重要单词字母的种类数，例如在样例中，共有3中字母'a','g','v'，因此a=3。 　　评测用例1和2满足2 ≤ n ≤ 3，1500 ≤ m ≤ 2000，s = 40； 　　评测用例3和4满足m = 20，2 ≤ a ≤ 3； 　　评测用例5、6和7满足2000 ≤ m ≤ 100000； 　　评测用例8满足n = 2； 　　所有的评测用例满足1 ≤ s ≤ 100，1 ≤ m ≤ 1015，每个单词至少包含1个字母，保证单词中仅出现英文小写字母，输入中不含多余字符，不会出现重复的单词。 |

高速公路 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201509-4 |
| 试题名称： | 高速公路 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　某国有n个城市，为了使得城市间的交通更便利，该国国王打算在城市之间修一些高速公路，由于经费限制，国王打算第一阶段先在部分城市之间修一些单向的高速公路。 　　现在，大臣们帮国王拟了一个修高速公路的计划。看了计划后，国王发现，有些城市之间可以通过高速公路直接（不经过其他城市）或间接（经过一个或多个其他城市）到达，而有的却不能。如果城市A可以通过高速公路到达城市B，而且城市B也可以通过高速公路到达城市A，则这两个城市被称为便利城市对。 　　国王想知道，在大臣们给他的计划中，有多少个便利城市对。  **输入格式**  　　输入的第一行包含两个整数n, m，分别表示城市和单向高速公路的数量。 　　接下来m行，每行两个整数a, b，表示城市a有一条单向的高速公路连向城市b。  **输出格式**  　　输出一行，包含一个整数，表示便利城市对的数量。  **样例输入**  5 5 1 2 2 3 3 4 4 2 3 5  **样例输出**  3  **样例说明**  http://118.190.20.162/RequireFile.do?fid=4HG9GgbF 　　城市间的连接如图所示。有3个便利城市对，它们分别是(2, 3), (2, 4), (3, 4)，请注意(2, 3)和(3, 2)看成同一个便利城市对。  **评测用例规模与约定**  　　前30%的评测用例满足1 ≤ n ≤ 100, 1 ≤ m ≤ 1000； 　　前60%的评测用例满足1 ≤ n ≤ 1000, 1 ≤ m ≤ 10000； 　　所有评测用例满足1 ≤ n ≤ 10000, 1 ≤ m ≤ 100000。 |

模板生成系统 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201509-3 |
| 试题名称： | 模板生成系统 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　成成最近在搭建一个网站，其中一些页面的部分内容来自数据库中不同的数据记录，但是页面的基本结构是相同的。例如，对于展示用户信息的页面，当用户为 Tom 时，网页的源代码是 http://118.190.20.162/RequireFile.do?fid=Mbg3eNaq  　　而当用户为 Jerry 时，网页的源代码是 http://118.190.20.162/RequireFile.do?fid=69mbB6TA  　　这样的例子在包含动态内容的网站中还有很多。为了简化生成网页的工作，成成觉得他需要引入一套模板生成系统。 　　模板是包含特殊标记的文本。成成用到的模板只包含一种特殊标记，格式为 {{ VAR }}，其中 VAR 是一个变量。该标记在模板生成时会被变量 VAR 的值所替代。例如，如果变量 name = "Tom"，则 {{ name }} 会生成 Tom。具体的规则如下： 　　·变量名由大小写字母、数字和下划线 (\_) 构成，且第一个字符不是数字，长度不超过 16 个字符。 　　·变量名是大小写敏感的，Name 和 name 是两个不同的变量。 　　·变量的值是字符串。 　　·如果标记中的变量没有定义，则生成空串，相当于把标记从模板中删除。 　　·模板不递归生成。也就是说，如果变量的值中包含形如 {{ VAR }} 的内容，不再做进一步的替换。  **输入格式**  　　输入的第一行包含两个整数 m, n，分别表示模板的行数和模板生成时给出的变量个数。 　　接下来 m 行，每行是一个字符串，表示模板。 　　接下来 n 行，每行表示一个变量和它的值，中间用一个空格分隔。值是字符串，用双引号 (") 括起来，内容可包含除双引号以外的任意可打印 ASCII 字符（ASCII 码范围 32, 33, 35-126）。  **输出格式**  　　输出包含若干行，表示模板生成的结果。  **样例输入**  11 2 <!DOCTYPE html> <html> <head> <title>User {{ name }}</title> </head> <body> <h1>{{ name }}</h1> <p>Email: <a href="mailto:{{ email }}">{{ email }}</a></p> <p>Address: {{ address }}</p> </body> </html> name "David Beckham" email "david@beckham.com"  **样例输出**  <!DOCTYPE html> <html> <head> <title>User David Beckham</title> </head> <body> <h1>David Beckham</h1> <p>Email: <a href="mailto:david@beckham.com">david@beckham.com</a></p> <p>Address: </p> </body> </html>  **评测用例规模与约定**  　　0 ≤ m ≤ 100 　　0 ≤ n ≤ 100 　　输入的模板每行长度不超过 80 个字符（不包含换行符）。 　　输入保证模板中所有以 {{ 开始的子串都是合法的标记，开始是两个左大括号和一个空格，然后是变量名，结尾是一个空格和两个右大括号。 　　输入中所有变量的值字符串长度不超过 100 个字符（不包括双引号）。 　　保证输入的所有变量的名字各不相同。 |

日期计算 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201509-2 |
| 试题名称： | 日期计算 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定一个年份y和一个整数d，问这一年的第d天是几月几日？ 　　注意闰年的2月有29天。满足下面条件之一的是闰年： 　　1） 年份是4的整数倍，而且不是100的整数倍； 　　2） 年份是400的整数倍。  **输入格式**  　　输入的第一行包含一个整数y，表示年份，年份在1900到2015之间（包含1900和2015）。 　　输入的第二行包含一个整数d，d在1至365之间。  **输出格式**  　　输出两行，每行一个整数，分别表示答案的月份和日期。  **样例输入**  2015 80  **样例输出**  3 21  **样例输入**  2000 40  **样例输出**  2 9 |

数列分段 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201509-1 |
| 试题名称： | 数列分段 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定一个整数数列，数列中连续相同的最长整数序列算成一段，问数列中共有多少段？  **输入格式**  　　输入的第一行包含一个整数n，表示数列中整数的个数。 　　第二行包含n个整数a1, a2, …, an，表示给定的数列，相邻的整数之间用一个空格分隔。  **输出格式**  　　输出一个整数，表示给定的数列有多个段。  **样例输入**  8 8 8 8 0 12 12 8 0  **样例输出**  5  **样例说明**  　　8 8 8是第一段，0是第二段，12 12是第三段，倒数第二个整数8是第四段，最后一个0是第五段。  **评测用例规模与约定**  　　1 ≤ n ≤ 1000，0 ≤ ai ≤ 1000。 |

矩阵 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201512-5 |
| 试题名称： | 矩阵 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　创造一个世界只需要定义一个初状态和状态转移规则。 　　宏观世界的物体运动规律始终跟物体当前的状态有关，也就是说只要知道物体足够多的状态信息，例如位置、速度等，我们就能知道物体之后任意时刻的状态。 　　现在小M创造了一个简化的世界。 　　这个世界中，时间是离散的，物理规律是线性的：世界的初始状态可以用一个m维向量b(0)表示，状态的转移方式用m×m的矩阵A表示。 　　若已知这个世界当前的状态是b，那么下一时刻就等于b左乘状态转移矩阵A，即Ab。 　　这个世界中，物体的状态也是离散的，也就是说可以用整数表示。再进一步，整数都可以用二进制编码拆分为有限位0和1。因此，这里的矩阵A和向量b的每个元素都是0或1，矩阵乘法中的加法运算视为异或运算（xor），乘法运算视为与运算（and）。 　　具体地，设矩阵A第i行第j列的元素为ai, j，向量b的第i个元素为bi。那么乘法Ab所得的第k个元素为 　　(ak,1 and b1) xor (ak,2 and b2) xor ⋯ xor (ak,m and bm) 　　矩阵和矩阵的乘法也有类似的表达。 　　小M发现，这样的矩阵运算也有乘法结合律，例如有A(Ab)=(AA)b=A2b。 　　为了保证自己创造的世界维度不轻易下降，小M保证了矩阵A可逆，也就是说存在一个矩阵A-1，使得对任意向量d，都有A-1Ad=d。 　　小M想了解自己创造的世界是否合理，他希望知道这个世界在不同时刻的状态。 　　具体地，小M有n组询问，每组询问会给出一个非负整数k，小M希望你帮他求出Akb。  **输入格式**  　　输入第一行包含一个整数m，表示矩阵和向量的规模。 　　接下来m行，每行包含一个长度为m的01串，表示矩阵A。 　　接下来一行，包含一个长度为m的01串，表示初始向量b(0)。（b(0)是列向量，这里表示它的转置） 　　注意：01串两个相邻的数字之间均没有空格。 　　接下来一行，包含一个正整数n，表示询问的个数。 　　最后n行，每行包含一个非负整数k，表示询问Akb(0)。 　　注意：k可能为0，此时是求A0b(0) =b(0)。  **输出格式**  　　输出n行，每行包含一个01串，表示对应询问中Akb(0)的结果。 　　注意：01串两个相邻的数字之间不要输出空格。  **样例输入**  3 110 011 111 101 10 0 2 3 14 1 1325 6 124124 151 12312  **样例输出**  101 010 111 101 110 010 100 101 001 100  **评测用例规模与约定**  　　本题使用10个评测用例来测试你的程序。 　　对于评测用例1，m = 10，n = 100，k ≤ 103。 　　对于评测用例2，m = 10，n = 100，k ≤ 104。 　　对于评测用例3，m = 30，n = 100，k ≤ 105。 　　对于评测用例4，m = 180，n = 100，k ≤ 105。 　　对于评测用例5，m = 10，n = 100，k ≤ 109。 　　对于评测用例6，m = 30，n = 100，k ≤ 109。 　　对于评测用例7，m = 180，n = 100，k ≤ 109。 　　对于评测用例8，m = 600，n = 100，k ≤ 109。 　　对于评测用例9，m = 800，n = 100，k ≤ 109。 　　对于评测用例10，m = 1000，n = 100，k ≤ 109。 |

送货 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201512-4 |
| 试题名称： | 送货 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　为了增加公司收入，F公司新开设了物流业务。由于F公司在业界的良好口碑，物流业务一开通即受到了消费者的欢迎，物流业务马上遍及了城市的每条街道。然而，F公司现在只安排了小明一个人负责所有街道的服务。 　　任务虽然繁重，但是小明有足够的信心，他拿到了城市的地图，准备研究最好的方案。城市中有n个交叉路口，m条街道连接在这些交叉路口之间，每条街道的首尾都正好连接着一个交叉路口。除开街道的首尾端点，街道不会在其他位置与其他街道相交。每个交叉路口都至少连接着一条街道，有的交叉路口可能只连接着一条或两条街道。 　　小明希望设计一个方案，从编号为1的交叉路口出发，每次必须沿街道去往街道另一端的路口，再从新的路口出发去往下一个路口，直到所有的街道都经过了正好一次。  **输入格式**  　　输入的第一行包含两个整数n, m，表示交叉路口的数量和街道的数量，交叉路口从1到n标号。 　　接下来m行，每行两个整数a, b，表示和标号为a的交叉路口和标号为b的交叉路口之间有一条街道，街道是双向的，小明可以从任意一端走向另一端。两个路口之间最多有一条街道。  **输出格式**  　　如果小明可以经过每条街道正好一次，则输出一行包含m+1个整数p1, p2, p3, ..., pm+1，表示小明经过的路口的顺序，相邻两个整数之间用一个空格分隔。如果有多种方案满足条件，则输出字典序最小的一种方案，即首先保证p1最小，p1最小的前提下再保证p2最小，依此类推。 　　如果不存在方案使得小明经过每条街道正好一次，则输出一个整数-1。  **样例输入**  4 5 1 2 1 3 1 4 2 4 3 4  **样例输出**  1 2 4 1 3 4  **样例说明**  　　城市的地图和小明的路径如下图所示。 http://118.190.20.162/RequireFile.do?fid=HgNYQ5G9  **样例输入**  4 6 1 2 1 3 1 4 2 4 3 4 2 3  **样例输出**  -1  **样例说明**  　　城市的地图如下图所示，不存在满足条件的路径。 http://118.190.20.162/RequireFile.do?fid=67NLAqAY  **评测用例规模与约定**  　　前30%的评测用例满足：1 ≤ n ≤ 10, n-1 ≤ m ≤ 20。 　　前50%的评测用例满足：1 ≤ n ≤ 100, n-1 ≤ m ≤ 10000。 　　所有评测用例满足：1 ≤ n ≤ 10000，n-1 ≤ m ≤ 100000。 |

画图 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201512-3 |
| 试题名称： | 画图 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　用 ASCII 字符来画图是一件有趣的事情，并形成了一门被称为 ASCII Art 的艺术。例如，下图是用 ASCII 字符画出来的 CSPRO 字样。 　　..\_\_\_\_.\_\_\_\_..\_\_\_\_..\_\_\_\_...\_\_\_.. 　　./.\_\_\_/.\_\_\_||..\_.\|..\_.\./.\_.\. 　　|.|...\\_\_\_.\|.|\_).|.|\_).|.|.|.| 　　|.|\_\_\_.\_\_\_).|..\_\_/|..\_.<|.|\_|.| 　　.\\_\_\_\_|\_\_\_\_/|\_|...|\_|.\\_\\\_\_\_/. 　　本题要求编程实现一个用 ASCII 字符来画图的程序，支持以下两种操作： 　　 画线：给出两个端点的坐标，画一条连接这两个端点的线段。简便起见题目保证要画的每条线段都是水平或者竖直的。水平线段用字符 - 来画，竖直线段用字符 | 来画。如果一条水平线段和一条竖直线段在某个位置相交，则相交位置用字符 + 代替。 　　 填充：给出填充的起始位置坐标和需要填充的字符，从起始位置开始，用该字符填充相邻位置，直到遇到画布边缘或已经画好的线段。注意这里的相邻位置只需要考虑上下左右 4 个方向，如下图所示，字符 @ 只和 4 个字符 \* 相邻。 　　.\*. 　　\*@\* 　　.\*.  **输入格式**  　　第1行有三个整数m, n和q。m和n分别表示画布的宽度和高度，以字符为单位。q表示画图操作的个数。 　　第2行至第q + 1行，每行是以下两种形式之一： 　　 0 x1 y1 x2 y2：表示画线段的操作，(x1, y1)和(x2, y2)分别是线段的两端，满足要么x1 = x2 且y1 ≠ y2，要么 y1 = y2 且 x1 ≠ x2。 　　 1 x y c：表示填充操作，(x, y)是起始位置，保证不会落在任何已有的线段上；c 为填充字符，是大小写字母。 　　画布的左下角是坐标为 (0, 0) 的位置，向右为x坐标增大的方向，向上为y坐标增大的方向。这q个操作按照数据给出的顺序依次执行。画布最初时所有位置都是字符 .（小数点）。  **输出格式**  　　输出有n行，每行m个字符，表示依次执行这q个操作后得到的画图结果。  **样例输入**  4 2 3 1 0 0 B 0 1 0 2 0 1 0 0 A  **样例输出**  AAAA A--A  **样例输入**  16 13 9 0 3 1 12 1 0 12 1 12 3 0 12 3 6 3 0 6 3 6 9 0 6 9 12 9 0 12 9 12 11 0 12 11 3 11 0 3 11 3 1 1 4 2 C  **样例输出**  ................ ...+--------+... ...|CCCCCCCC|... ...|CC+-----+... ...|CC|......... ...|CC|......... ...|CC|......... ...|CC|......... ...|CC|......... ...|CC+-----+... ...|CCCCCCCC|... ...+--------+... ................  **评测用例规模与约定**  　　所有的评测用例满足：2 ≤ m, n ≤ 100，0 ≤ q ≤ 100，0 ≤ x < m（x表示输入数据中所有位置的x坐标），0 ≤ y < n（y表示输入数据中所有位置的y坐标）。 |

消除类游戏 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201512-2 |
| 试题名称： | 消除类游戏 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　消除类游戏是深受大众欢迎的一种游戏，游戏在一个包含有n行m列的游戏棋盘上进行，棋盘的每一行每一列的方格上放着一个有颜色的棋子，当一行或一列上有连续三个或更多的相同颜色的棋子时，这些棋子都被消除。当有多处可以被消除时，这些地方的棋子将同时被消除。 　　现在给你一个n行m列的棋盘，棋盘中的每一个方格上有一个棋子，请给出经过一次消除后的棋盘。 　　请注意：一个棋子可能在某一行和某一列同时被消除。  **输入格式**  　　输入的第一行包含两个整数n, m，用空格分隔，分别表示棋盘的行数和列数。 　　接下来n行，每行m个整数，用空格分隔，分别表示每一个方格中的棋子的颜色。颜色使用1至9编号。  **输出格式**  　　输出n行，每行m个整数，相邻的整数之间使用一个空格分隔，表示经过一次消除后的棋盘。如果一个方格中的棋子被消除，则对应的方格输出0，否则输出棋子的颜色编号。  **样例输入**  4 5 2 2 3 1 2 3 4 5 1 4 2 3 2 1 3 2 2 2 4 4  **样例输出**  2 2 3 0 2 3 4 5 0 4 2 3 2 0 3 0 0 0 4 4  **样例说明**  　　棋盘中第4列的1和第4行的2可以被消除，其他的方格中的棋子均保留。  **样例输入**  4 5 2 2 3 1 2 3 1 1 1 1 2 3 2 1 3 2 2 3 3 3  **样例输出**  2 2 3 0 2 3 0 0 0 0 2 3 2 0 3 2 2 0 0 0  **样例说明**  　　棋盘中所有的1以及最后一行的3可以被同时消除，其他的方格中的棋子均保留。  **评测用例规模与约定**  　　所有的评测用例满足：1 ≤ n, m ≤ 30。 |

数位之和 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201512-1 |
| 试题名称： | 数位之和 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定一个十进制整数n，输出n的各位数字之和。  **输入格式**  　　输入一个整数n。  **输出格式**  　　输出一个整数，表示答案。  **样例输入**  20151220  **样例输出**  13  **样例说明**  　　20151220的各位数字之和为2+0+1+5+1+2+2+0=13。  **评测用例规模与约定**  　　所有评测用例满足：0 ≤ n ≤ 1000000000。 |

网络连接 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201604-5 |
| 试题名称： | 网络连接 |
| 时间限制： | 5.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　某校新建的大楼中有n台设备，学校需要利用这些设备搭建一个网络。我们用1到n的整数给这些设备编号。 　　这些设备之间一共可以建立m条连线，建立每条连线会消耗一定的费用。连接建立后两台设备就可以相互通信了。两台设备可以借助其他设备进行通信，即通信关系可传递：如果设备A和设备B都能与设备C相互通信，那么设备A和设备B也能相互通信。 　　由于大楼的拓扑结构所限，可以建立连线的两台设备，一定满足其编号之差的绝对值不超过p。 　　这n台设备中的一部分属于用户设备。学校要求在最终的网络中，用户设备必须两两能够相互通信，其他设备则可以根据需要选择连线或不连线。 　　现在问要达到学校的要求最少要消耗多少的费用。  **输入格式**  　　输入的第一行包含一个正整数T，表示数据的组数，保证T=5。接下来依次描述每组数据。 　　每组数据的第一行包含3个正整数n, m, p，表示设备的总数，可以建立的连线数量和拓扑结构的参数。 　　第二行包含一个长度为n的01字符串，依次表示这n台设备是否为用户设备；为1表示是，为0表示不是。相邻字符之间无空格隔开，保证不会出现除了0和1之外的字符。保证至少有2个设备是用户设备。 　　接下来m行，每行包含3个非负整数u, v, w，表示设备u和设备v可以消耗w的费用建立连线。其中0 < u < v ≤ n，v – u ≤ p，w ≤ 106。 　　除第二行外，所有的数之间用一个空格隔开。 　　保证两台设备之间最多只有一条可以建立的连线，保证至少存在一种方案能够满足学校的要求。  **输出格式**  　　对于每组数据，输出一行一个整数，表示能达到要求所需的最少费用。  **样例输入**  1 20 11 6 10000100001100000000 1 6 300 1 3 100 3 6 100 4 10 100 4 6 100 6 10 400 10 15 100 11 15 100 10 11 500 12 15 100 15 20 100  **样例输出**  700  **样例说明**  　　用户设备分别是1、6、11、12。最优的方案需要选择以下连线：设备1和3，费用100；设备3和6，费用100；设备4和6，费用100；设备4和10，费用100；设备10和15，费用100；设备11和15，费用100；设备12和15，费用100。共计700。  **子任务**  　　每个测试点的每组数据分别都满足以下限制（其中c表示用户设备总数）：   |  |  |  |  | | --- | --- | --- | --- | | 编号 | n | p | c | | 1 | =500 | =6 | =n | | 2 | =500 | =6 | =2 | | 3 | =500 | =6 | =4 | | 4 | =500 | =6 | =6 | | 5 | =500 | =6 | =10 | | 6 | =500 | =2 | 无限制 | | 7 | =500 | =2 | 无限制 | | 8 | =500 | =3 | 无限制 | | 9 | =500 | =6 | 无限制 | | 10 | =500 | =6 | 无限制 |   **样例输入**  5 20 93 6 11111111111111111111 1 2 727765 2 3 263881 1 4 514909 2 4 131844 3 4 933178 1 5 438613 2 5 603733 3 5 859906 4 5 38725 1 6 599082 2 6 225184 3 6 365575 4 6 126126 5 6 25505 2 7 546601 3 7 63931 4 7 419406 5 7 531790 6 7 265271 2 8 724126 3 8 163879 4 8 59317 5 8 545880 6 8 126779 7 8 938859 4 9 387231 5 9 600880 6 9 927742 7 9 56690 8 9 987324 4 10 23106 5 10 328383 6 10 757082 7 10 787341 8 10 218539 9 10 282814 5 11 506005 6 11 257575 7 11 524808 8 11 223439 9 11 402018 10 11 580131 6 12 711292 7 12 314072 8 12 752595 9 12 141042 10 12 339498 11 12 321201 7 13 100337 8 13 969444 9 13 683912 10 13 973527 11 13 776443 12 13 436444 8 14 826460 9 14 453950 10 14 231374 11 14 667343 13 14 754318 10 15 46540 11 15 741941 12 15 909374 13 15 901791 14 15 246817 10 16 34192 11 16 866848 12 16 488324 13 16 824636 14 16 319615 15 16 472537 11 17 661823 12 17 595773 13 17 912773 14 17 674733 15 17 880787 16 17 168668 12 18 621903 13 18 894662 14 18 994829 15 18 80344 16 18 343121 17 18 697098 13 19 569872 14 19 599332 15 19 417377 16 19 837022 18 19 270641 14 20 748579 15 20 724926 16 20 367572 17 20 103857 18 20 443540 19 20 932348 20 92 6 00000010000000001000 1 2 113099 1 3 587351 2 3 209419 1 4 412024 2 4 526550 3 4 74575 1 5 627809 2 5 488889 3 5 78667 4 5 853953 1 6 914999 2 6 60305 3 6 52666 4 6 39394 5 6 243622 1 7 843636 2 7 752452 3 7 206953 4 7 209536 5 7 105503 6 7 179097 2 8 938375 3 8 449664 4 8 464591 5 8 312120 6 8 943383 7 8 798642 3 9 759837 4 9 655902 5 9 367024 6 9 560174 7 9 421235 8 9 552637 4 10 251605 5 10 453809 6 10 942947 7 10 378285 8 10 944748 9 10 761726 5 11 7926 6 11 237034 8 11 860297 9 11 224152 10 11 62671 7 12 518894 9 12 265459 10 12 399585 11 12 16993 7 13 935813 8 13 675125 9 13 986248 10 13 801332 11 13 775825 12 13 483974 8 14 879925 9 14 568657 10 14 539371 11 14 392819 12 14 673171 13 14 593981 9 15 893512 10 15 459005 11 15 935151 13 15 984026 14 15 352506 10 16 847405 11 16 902823 12 16 924659 14 16 31185 15 16 83802 11 17 225697 12 17 697702 13 17 431672 14 17 934227 15 17 64055 16 17 714592 12 18 429691 13 18 69217 16 18 916446 17 18 859123 13 19 675753 14 19 427831 15 19 3386 16 19 179655 17 19 148897 18 19 307554 14 20 291004 15 20 497622 16 20 669001 17 20 742221 18 20 662204 19 20 966218 20 93 6 00000101001110000010 1 2 710445 1 3 390108 2 3 871795 1 4 323940 2 4 1557 3 4 853515 1 5 987497 2 5 791951 3 5 565856 4 5 401932 1 6 49223 2 6 59889 3 6 300678 4 6 123895 5 6 160344 1 7 196308 3 7 787087 5 7 224478 6 7 104395 3 8 344509 4 8 173377 5 8 100930 7 8 544113 3 9 475119 4 9 673881 5 9 234187 6 9 191885 7 9 35559 8 9 872991 4 10 339910 5 10 624600 6 10 442925 7 10 821981 8 10 805172 9 10 463188 5 11 745511 6 11 444346 7 11 856170 8 11 10706 9 11 582739 10 11 666926 6 12 838187 7 12 711050 8 12 386931 9 12 886249 10 12 356669 11 12 14156 7 13 339092 8 13 578454 9 13 195419 10 13 600017 11 13 504670 12 13 753722 8 14 219160 9 14 281523 10 14 730539 11 14 890253 12 14 789909 13 14 995665 9 15 227552 10 15 125146 11 15 737706 12 15 697809 13 15 753633 14 15 248107 10 16 15962 11 16 178154 12 16 586355 13 16 32073 14 16 912542 15 16 913266 11 17 158548 12 17 35601 14 17 818081 15 17 39970 16 17 330500 12 18 207654 13 18 729538 14 18 642935 15 18 73734 16 18 676418 17 18 455930 13 19 945333 15 19 723755 16 19 67978 17 19 128589 18 19 647790 14 20 294150 15 20 681124 16 20 749167 17 20 480962 18 20 133266 19 20 928077 20 34 2 10000111010111001100 1 2 175025 1 3 372293 2 3 723523 2 4 828462 3 4 96614 3 5 496039 4 5 435856 4 6 728072 5 6 26736 5 7 658831 6 7 294479 6 8 971178 7 8 755365 7 9 467364 8 9 912893 8 10 88692 9 10 132051 9 11 939008 10 11 986048 11 12 676052 11 13 287187 12 13 304051 13 14 305223 14 15 610873 14 16 540274 15 16 494014 15 17 90777 16 17 157248 16 18 211384 17 18 718096 17 19 822166 18 19 366585 18 20 250962 19 20 231236 20 89 6 00110010100101111001 1 2 758268 1 3 205716 2 3 295263 1 4 207862 2 4 69876 3 4 179798 2 5 479889 3 5 953848 4 5 68337 1 6 833892 2 6 234107 4 6 772558 5 6 675566 2 7 942986 3 7 253909 4 7 95158 5 7 119424 6 7 316766 2 8 477692 3 8 406515 4 8 987774 5 8 808702 6 8 184409 7 8 500278 3 9 887393 4 9 469162 6 9 293078 7 9 205359 8 9 777079 5 10 313301 6 10 636761 7 10 846529 8 10 315392 9 10 477016 5 11 29206 6 11 840608 7 11 151527 8 11 877001 9 11 439665 10 11 907197 7 12 815137 8 12 530916 9 12 922868 11 12 827381 7 13 887939 8 13 638323 9 13 311582 10 13 169399 11 13 763999 12 13 695711 8 14 686748 9 14 400272 10 14 39935 11 14 528026 12 14 776064 13 14 965438 10 15 457267 12 15 318406 13 15 3489 14 15 217063 11 16 763755 12 16 711741 13 16 617167 14 16 469274 15 16 845875 11 17 776936 12 17 464487 13 17 682511 14 17 11831 15 17 447702 16 17 10247 12 18 10991 13 18 388512 14 18 312431 15 18 427326 16 18 846191 17 18 977005 13 19 941472 14 19 709497 15 19 343667 16 19 346238 17 19 777055 18 19 703066 14 20 779760 15 20 559561 16 20 810681 17 20 436275 18 20 371192 19 20 42384  **样例输出**  2402044 339126 523926 4622029 1721412 |

游戏 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201604-4 |
| 试题名称： | 游戏 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　小明在玩一个电脑游戏，游戏在一个n×m的方格图上进行，小明控制的角色开始的时候站在第一行第一列，目标是前往第n行第m列。 　　方格图上有一些方格是始终安全的，有一些在一段时间是危险的，如果小明控制的角色到达一个方格的时候方格是危险的，则小明输掉了游戏，如果小明的角色到达了第n行第m列，则小明过关。第一行第一列和第n行第m列永远都是安全的。 　　每个单位时间，小明的角色必须向上下左右四个方向相邻的方格中的一个移动一格。 　　经过很多次尝试，小明掌握了方格图的安全和危险的规律：每一个方格出现危险的时间一定是连续的。并且，小明还掌握了每个方格在哪段时间是危险的。 　　现在，小明想知道，自己最快经过几个时间单位可以达到第n行第m列过关。  **输入格式**  　　输入的第一行包含三个整数n, m, t，用一个空格分隔，表示方格图的行数n、列数m，以及方格图中有危险的方格数量。 　　接下来t行，每行4个整数r, c, a, b，表示第r行第c列的方格在第a个时刻到第b个时刻之间是危险的，包括a和b。游戏开始时的时刻为0。输入数据保证r和c不同时为1，而且当r为n时c不为m。一个方格只有一段时间是危险的（或者说不会出现两行拥有相同的r和c）。  **输出格式**  　　输出一个整数，表示小明最快经过几个时间单位可以过关。输入数据保证小明一定可以过关。  **样例输入**  3 3 3 2 1 1 1 1 3 2 10 2 2 2 10  **样例输出**  6  **样例说明**  　　第2行第1列时刻1是危险的，因此第一步必须走到第1行第2列。 　　第二步可以走到第1行第1列，第三步走到第2行第1列，后面经过第3行第1列、第3行第2列到达第3行第3列。  **评测用例规模与约定**  　　前30%的评测用例满足：0 < n, m ≤ 10，0 ≤ t < 99。 　　所有评测用例满足：0 < n, m ≤ 100，0 ≤ t < 9999，1 ≤ r ≤ n，1 ≤ c ≤ m，0 ≤ a ≤ b ≤ 100。 |

路径解析 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201604-3 |
| 试题名称： | 路径解析 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在操作系统中，数据通常以文件的形式存储在文件系统中。文件系统一般采用层次化的组织形式，由目录（或者文件夹）和文件构成，形成一棵树的形状。文件有内容，用于存储数据。目录是容器，可包含文件或其他目录。同一个目录下的所有文件和目录的名字各不相同，不同目录下可以有名字相同的文件或目录。 　　为了指定文件系统中的某个文件，需要用路径来定位。在类 Unix 系统（Linux、Max OS X、FreeBSD等）中，路径由若干部分构成，每个部分是一个目录或者文件的名字，相邻两个部分之间用 / 符号分隔。 　　有一个特殊的目录被称为根目录，是整个文件系统形成的这棵树的根节点，用一个单独的 / 符号表示。在操作系统中，有当前目录的概念，表示用户目前正在工作的目录。根据出发点可以把路径分为两类： 　　 绝对路径：以 / 符号开头，表示从根目录开始构建的路径。 　　 相对路径：不以 / 符号开头，表示从当前目录开始构建的路径。  　　例如，有一个文件系统的结构如下图所示。在这个文件系统中，有根目录 / 和其他普通目录 d1、d2、d3、d4，以及文件 f1、f2、f3、f1、f4。其中，两个 f1 是同名文件，但在不同的目录下。 http://118.190.20.162/RequireFile.do?fid=aMLNafRq 　　对于 d4 目录下的 f1 文件，可以用绝对路径 /d2/d4/f1 来指定。如果当前目录是 /d2/d3，这个文件也可以用相对路径 ../d4/f1 来指定，这里 .. 表示上一级目录（注意，根目录的上一级目录是它本身）。还有 . 表示本目录，例如 /d1/./f1 指定的就是 /d1/f1。注意，如果有多个连续的 / 出现，其效果等同于一个 /，例如 /d1///f1 指定的也是 /d1/f1。 　　本题会给出一些路径，要求对于每个路径，给出正规化以后的形式。一个路径经过正规化操作后，其指定的文件不变，但是会变成一个不包含 . 和 .. 的绝对路径，且不包含连续多个 / 符号。如果一个路径以 / 结尾，那么它代表的一定是一个目录，正规化操作要去掉结尾的 /。若这个路径代表根目录，则正规化操作的结果是 /。若路径为空字符串，则正规化操作的结果是当前目录。  **输入格式**  　　第一行包含一个整数 P，表示需要进行正规化操作的路径个数。 　　第二行包含一个字符串，表示当前目录。 　　以下 P 行，每行包含一个字符串，表示需要进行正规化操作的路径。  **输出格式**  　　共 P 行，每行一个字符串，表示经过正规化操作后的路径，顺序与输入对应。  **样例输入**  7 /d2/d3 /d2/d4/f1 ../d4/f1 /d1/./f1 /d1///f1 /d1/ /// /d1/../../d2  **样例输出**  /d2/d4/f1 /d2/d4/f1 /d1/f1 /d1/f1 /d1 / /d2  **评测用例规模与约定**  　　1 ≤ P ≤ 10。 　　文件和目录的名字只包含大小写字母、数字和小数点 .、减号 - 以及下划线 \_。 　　不会有文件或目录的名字是 . 或 .. ，它们具有题目描述中给出的特殊含义。 　　输入的所有路径每个长度不超过 1000 个字符。 　　输入的当前目录保证是一个经过正规化操作后的路径。 　　对于前 30% 的测试用例，需要正规化的路径的组成部分不包含 . 和 .. 。 　　对于前 60% 的测试用例，需要正规化的路径都是绝对路径。 |

俄罗斯方块 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201604-2 |
| 试题名称： | 俄罗斯方块 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **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 |

折点计数 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201604-1 |
| 试题名称： | 折点计数 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定n个整数表示一个商店连续n天的销售量。如果某天之前销售量在增长，而后一天销售量减少，则称这一天为折点，反过来如果之前销售量减少而后一天销售量增长，也称这一天为折点。其他的天都不是折点。如下图中，第3天和第6天是折点。 http://118.190.20.162/RequireFile.do?fid=gbLRbhR7 　　给定n个整数a1, a2, …, an表示销售量，请计算出这些天总共有多少个折点。 　　为了减少歧义，我们给定的数据保证：在这n天中相邻两天的销售量总是不同的，即ai-1≠ai。注意，如果两天不相邻，销售量可能相同。  **输入格式**  　　输入的第一行包含一个整数n。 　　第二行包含n个整数，用空格分隔，分别表示a1, a2, …, an。  **输出格式**  　　输出一个整数，表示折点出现的数量。  **样例输入**  7 5 4 1 2 3 6 4  **样例输出**  2  **评测用例规模与约定**  　　所有评测用例满足：1 ≤ n ≤ 1000，每天的销售量是不超过10000的非负整数。 |

祭坛 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-5 |
| 试题名称： | 祭坛 |
| 时间限制： | 2.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在遥远的Dgeak大陆，生活着一种叫做Dar-dzo-nye的怪物。每当这种怪物降临，人们必须整夜对抗怪物而不能安睡。为了乞求这种怪物不再降临，人们决定建造祭坛。 　　Dgeak大陆可以看成一个用平面直角坐标系表示的巨大平面。在这个平面上，有 n 个Swaryea水晶柱，每个水晶柱可以用一个点表示。 　　如果 4 个水晶柱依次相连可以构成一个四边形，满足其两条对角线分别平行于 x 轴和 y 轴，并且对角线的交点位于四边形内部（不包括边界），那么这 4 个水晶柱就可以建立一个结界。其中，对角线的交点称作这个结界的中心。 　　例如下左图中，水晶柱 ABCD 可以建立一个结界，其中心为 O。 http://118.190.20.162/RequireFile.do?fid=TyfBBLAehttp://118.190.20.162/RequireFile.do?fid=Lej3jN8N 　　为了起到抵御Dar-dzo-nye的最佳效果，人们会把祭坛修建在最多层结界的保护中。其中不同层的结界必须有共同的中心，这些结界的边界不能有任何公共点，并且中心处也不能有水晶柱。这里共同中心的结界数量叫做结界的层数。 　　为了达成这个目的，人们要先利用现有的水晶柱建立若干个结界，然后在某些结界的中心建立祭坛。 　　例如上右图中，黑色的点表示水晶柱（注意 P 和 O 点不是水晶柱）。祭坛的一个最佳位置为 O 点，可以建立在 3 层结界中，其结界的具体方案见下左图。当然，建立祭坛的最佳位置不一定是唯一，在上右图中，O 点左侧 1 单位的点 P 也可以建立一个在 3 层结界中的祭坛，见下右图。 http://118.190.20.162/RequireFile.do?fid=EmBb9j4ghttp://118.190.20.162/RequireFile.do?fid=RM5T9rDf  　　现在人们想知道： 　　1. 祭坛最佳选址地点所在的结界层数； 　　2. 祭坛最佳的选址地点共有多少个。  **输入格式**  　　输入的第一行包含两个正整数 n,q，表示水晶柱的个数和问题的种类。保证 q=1 或 2，其意义见输出格式。 　　接下来 n 行，每行包含两个非负整数 x,y，表示每个水晶柱的坐标。保证相同的坐标不会重复出现。  **输出格式**  　　若 q=1，输出一行一个整数，表示祭坛最多可以位于多少个结界的中心；若 q=2，输出一行一个整数，表示结界数最多的方案有多少种。  **样例1输入**  　　26 1 　　0 5 　　1 1 　　1 5 　　1 9 　　3 5 　　3 10 　　4 0 　　4 1 　　4 2 　　4 4 　　4 6 　　4 9 　　4 11 　　5 0 　　5 2 　　5 4 　　5 8 　　5 9 　　5 10 　　5 11 　　6 5 　　7 5 　　8 5 　　9 10 　　10 2 　　10 5  **样例1输出**  　　3  **样例2输入**  　　26 2 　　0 5 　　1 1 　　1 5 　　1 9 　　3 5 　　3 10 　　4 0 　　4 1 　　4 2 　　4 4 　　4 6 　　4 9 　　4 11 　　5 0 　　5 2 　　5 4 　　5 8 　　5 9 　　5 10 　　5 11 　　6 5 　　7 5 　　8 5 　　9 10 　　10 2 　　10 5  **样例2输出**  　　2  **样例说明**  　　样例即为题目描述中的例子，两个样例数据相同，分别询问最多的结界数量和达到最多结界数量的方案数。 　　其中图片的左下角为原点，右和上分别是 x 轴和 y 轴的正方向，一个格子的长度为单位长度。 　　以图中的 O 点建立祭坛，祭坛最多可以位于 3 个结界的中心。不存在更多结界的方案，因此样例1的答案为 3。 　　在 O 点左侧 1 单位的点 (4,5) 也可以建立一个在 3 个结界中的祭坛，因此样例2的答案为 2。  **评测用例规模与约定**  　　对于所有的数据，保证存在至少一种方案，使得祭坛建造在至少一层结界中，即不存在无论如何祭坛都无法建造在结界中的情况。 　　数据分为 8 类，各类之间互相没有交集，分别有以下特点： 　　1. 占数据的 10%，n=200，x,y≤n； 　　2. 占数据的 10%，n=200，x,y≤109； 　　3. 占数据的 10%，n=1000，x,y≤n； 　　4. 占数据的 10%，n=1000，x,y≤109； 　　5. 占数据的 10%，n=5000，x,y≤n； 　　6. 占数据的 10%，n=5000，x,y≤109； 　　7. 占数据的 20%，n=300000，x,y≤n； 　　8. 占数据的 20%，n=300000，x,y≤109。  　　此外，每类数据中，q=1 与 q=2 各占恰好一半。 |

交通规划 3 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-4 |
| 试题名称： | 交通规划 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　G国国王来中国参观后，被中国的高速铁路深深的震撼，决定为自己的国家也建设一个高速铁路系统。 　　建设高速铁路投入非常大，为了节约建设成本，G国国王决定不新建铁路，而是将已有的铁路改造成高速铁路。现在，请你为G国国王提供一个方案，将现有的一部分铁路改造成高速铁路，使得任何两个城市间都可以通过高速铁路到达，而且从所有城市乘坐高速铁路到首都的最短路程和原来一样长。请你告诉G国国王在这些条件下最少要改造多长的铁路。  **输入格式**  　　输入的第一行包含两个整数n, m，分别表示G国城市的数量和城市间铁路的数量。所有的城市由1到n编号，首都为1号。 　　接下来m行，每行三个整数a, b, c，表示城市a和城市b之间有一条长度为c的双向铁路。这条铁路不会经过a和b以外的城市。  **输出格式**  　　输出一行，表示在满足条件的情况下最少要改造的铁路长度。  **样例输入**  4 5 1 2 4 1 3 5 2 3 2 2 4 3 3 4 2  **样例输出**  11  **评测用例规模与约定**  　　对于20%的评测用例，1 ≤ n ≤ 10，1 ≤ m ≤ 50； 　　对于50%的评测用例，1 ≤ n ≤ 100，1 ≤ m ≤ 5000； 　　对于80%的评测用例，1 ≤ n ≤ 1000，1 ≤ m ≤ 50000； 　　对于100%的评测用例，1 ≤ n ≤ 10000，1 ≤ m ≤ 100000，1 ≤ a, b ≤ n，1 ≤ c ≤ 1000。输入保证每个城市都可以通过铁路达到首都。 |

**问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-3 |
| 试题名称： | 炉石传说 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　《炉石传说：魔兽英雄传》（Hearthstone: Heroes of Warcraft，简称炉石传说）是暴雪娱乐开发的一款集换式卡牌游戏（如下图所示）。游戏在一个战斗棋盘上进行，由两名玩家轮流进行操作，本题所使用的炉石传说游戏的简化规则如下： http://118.190.20.162/RequireFile.do?fid=RT2fdRLm 　　\* 玩家会控制一些角色，每个角色有自己的生命值和攻击力。当生命值小于等于 0 时，该角色死亡。角色分为英雄和随从。 　　\* 玩家各控制一个英雄，游戏开始时，英雄的生命值为 30，攻击力为 0。当英雄死亡时，游戏结束，英雄未死亡的一方获胜。 　　\* 玩家可在游戏过程中召唤随从。棋盘上每方都有 7 个可用于放置随从的空位，从左到右一字排开，被称为战场。当随从死亡时，它将被从战场上移除。 　　\* 游戏开始后，两位玩家轮流进行操作，每个玩家的连续一组操作称为一个回合。 　　\* 每个回合中，当前玩家可进行零个或者多个以下操作： 　　1) 召唤随从：玩家召唤一个随从进入战场，随从具有指定的生命值和攻击力。 　　2) 随从攻击：玩家控制自己的某个随从攻击对手的英雄或者某个随从。 　　3) 结束回合：玩家声明自己的当前回合结束，游戏将进入对手的回合。该操作一定是一个回合的最后一个操作。 　　\* 当随从攻击时，攻击方和被攻击方会同时对彼此造成等同于自己攻击力的伤害。受到伤害的角色的生命值将会减少，数值等同于受到的伤害。例如，随从 X 的生命值为 HX、攻击力为 AX，随从 Y 的生命值为 HY、攻击力为 AY，如果随从 X 攻击随从 Y，则攻击发生后随从 X 的生命值变为 HX - AY，随从 Y 的生命值变为 HY - AX。攻击发生后，角色的生命值可以为负数。 　　本题将给出一个游戏的过程，要求编写程序模拟该游戏过程并输出最后的局面。  **输入格式**  　　输入第一行是一个整数 n，表示操作的个数。接下来 n 行，每行描述一个操作，格式如下： 　　<action> <arg1> <arg2> ... 　　其中<action>表示操作类型，是一个字符串，共有 3 种：summon表示召唤随从，attack表示随从攻击，end表示结束回合。这 3 种操作的具体格式如下： 　　\* summon <position> <attack> <health>：当前玩家在位置<position>召唤一个生命值为<health>、攻击力为<attack>的随从。其中<position>是一个 1 到 7 的整数，表示召唤的随从出现在战场上的位置，原来该位置及右边的随从都将顺次向右移动一位。 　　\* attack <attacker> <defender>：当前玩家的角色<attacker>攻击对方的角色 <defender>。<attacker>是 1 到 7 的整数，表示发起攻击的本方随从编号，<defender>是 0 到 7 的整数，表示被攻击的对方角色，0 表示攻击对方英雄，1 到 7 表示攻击对方随从的编号。 　　\* end：当前玩家结束本回合。 　　注意：随从的编号会随着游戏的进程发生变化，当召唤一个随从时，玩家指定召唤该随从放入战场的位置，此时，原来该位置及右边的所有随从编号都会增加 1。而当一个随从死亡时，它右边的所有随从编号都会减少 1。任意时刻，战场上的随从总是从1开始连续编号。  **输出格式**  　　输出共 5 行。 　　第 1 行包含一个整数，表示这 n 次操作后（以下称为 T 时刻）游戏的胜负结果，1 表示先手玩家获胜，-1 表示后手玩家获胜，0 表示游戏尚未结束，还没有人获胜。 　　第 2 行包含一个整数，表示 T 时刻先手玩家的英雄的生命值。 　　第 3 行包含若干个整数，第一个整数 p 表示 T 时刻先手玩家在战场上存活的随从个数，之后 p 个整数，分别表示这些随从在 T 时刻的生命值（按照从左往右的顺序）。 　　第 4 行和第 5 行与第 2 行和第 3 行类似，只是将玩家从先手玩家换为后手玩家。  **样例输入**  8 summon 1 3 6 summon 2 4 2 end summon 1 4 5 summon 1 2 1 attack 1 2 end attack 1 1  **样例输出**  0 30 1 2 30 1 2  **样例说明**  　　按照样例输入从第 2 行开始逐行的解释如下： 　　1. 先手玩家在位置 1 召唤一个生命值为 6、攻击力为 3 的随从 A，是本方战场上唯一的随从。 　　2. 先手玩家在位置 2 召唤一个生命值为 2、攻击力为 4 的随从 B，出现在随从 A 的右边。 　　3. 先手玩家回合结束。 　　4. 后手玩家在位置 1 召唤一个生命值为 5、攻击力为 4 的随从 C，是本方战场上唯一的随从。 　　5. 后手玩家在位置 1 召唤一个生命值为 1、攻击力为 2 的随从 D，出现在随从 C 的左边。 　　6. 随从 D 攻击随从 B，双方均死亡。 　　7. 后手玩家回合结束。 　　8. 随从 A 攻击随从 C，双方的生命值都降低至 2。  **评测用例规模与约定**  　　\* 操作的个数0 ≤ n ≤ 1000。 　　\* 随从的初始生命值为 1 到 100 的整数，攻击力为 0 到 100 的整数。 　　\* 保证所有操作均合法，包括但不限于： 　　1) 召唤随从的位置一定是合法的，即如果当前本方战场上有 m 个随从，则召唤随从的位置一定在 1 到 m + 1 之间，其中 1 表示战场最左边的位置，m + 1 表示战场最右边的位置。 　　2) 当本方战场有 7 个随从时，不会再召唤新的随从。 　　3) 发起攻击和被攻击的角色一定存在，发起攻击的角色攻击力大于 0。 　　4) 一方英雄如果死亡，就不再会有后续操作。 　　\* 数据约定： 　　前 20% 的评测用例召唤随从的位置都是战场的最右边。 　　前 40% 的评测用例没有 attack 操作。 　　前 60% 的评测用例不会出现随从死亡的情况。 |

炉石传说 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-3 |
| 试题名称： | 炉石传说 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　《炉石传说：魔兽英雄传》（Hearthstone: Heroes of Warcraft，简称炉石传说）是暴雪娱乐开发的一款集换式卡牌游戏（如下图所示）。游戏在一个战斗棋盘上进行，由两名玩家轮流进行操作，本题所使用的炉石传说游戏的简化规则如下： http://118.190.20.162/RequireFile.do?fid=RT2fdRLm 　　\* 玩家会控制一些角色，每个角色有自己的生命值和攻击力。当生命值小于等于 0 时，该角色死亡。角色分为英雄和随从。 　　\* 玩家各控制一个英雄，游戏开始时，英雄的生命值为 30，攻击力为 0。当英雄死亡时，游戏结束，英雄未死亡的一方获胜。 　　\* 玩家可在游戏过程中召唤随从。棋盘上每方都有 7 个可用于放置随从的空位，从左到右一字排开，被称为战场。当随从死亡时，它将被从战场上移除。 　　\* 游戏开始后，两位玩家轮流进行操作，每个玩家的连续一组操作称为一个回合。 　　\* 每个回合中，当前玩家可进行零个或者多个以下操作： 　　1) 召唤随从：玩家召唤一个随从进入战场，随从具有指定的生命值和攻击力。 　　2) 随从攻击：玩家控制自己的某个随从攻击对手的英雄或者某个随从。 　　3) 结束回合：玩家声明自己的当前回合结束，游戏将进入对手的回合。该操作一定是一个回合的最后一个操作。 　　\* 当随从攻击时，攻击方和被攻击方会同时对彼此造成等同于自己攻击力的伤害。受到伤害的角色的生命值将会减少，数值等同于受到的伤害。例如，随从 X 的生命值为 HX、攻击力为 AX，随从 Y 的生命值为 HY、攻击力为 AY，如果随从 X 攻击随从 Y，则攻击发生后随从 X 的生命值变为 HX - AY，随从 Y 的生命值变为 HY - AX。攻击发生后，角色的生命值可以为负数。 　　本题将给出一个游戏的过程，要求编写程序模拟该游戏过程并输出最后的局面。  **输入格式**  　　输入第一行是一个整数 n，表示操作的个数。接下来 n 行，每行描述一个操作，格式如下： 　　<action> <arg1> <arg2> ... 　　其中<action>表示操作类型，是一个字符串，共有 3 种：summon表示召唤随从，attack表示随从攻击，end表示结束回合。这 3 种操作的具体格式如下： 　　\* summon <position> <attack> <health>：当前玩家在位置<position>召唤一个生命值为<health>、攻击力为<attack>的随从。其中<position>是一个 1 到 7 的整数，表示召唤的随从出现在战场上的位置，原来该位置及右边的随从都将顺次向右移动一位。 　　\* attack <attacker> <defender>：当前玩家的角色<attacker>攻击对方的角色 <defender>。<attacker>是 1 到 7 的整数，表示发起攻击的本方随从编号，<defender>是 0 到 7 的整数，表示被攻击的对方角色，0 表示攻击对方英雄，1 到 7 表示攻击对方随从的编号。 　　\* end：当前玩家结束本回合。 　　注意：随从的编号会随着游戏的进程发生变化，当召唤一个随从时，玩家指定召唤该随从放入战场的位置，此时，原来该位置及右边的所有随从编号都会增加 1。而当一个随从死亡时，它右边的所有随从编号都会减少 1。任意时刻，战场上的随从总是从1开始连续编号。  **输出格式**  　　输出共 5 行。 　　第 1 行包含一个整数，表示这 n 次操作后（以下称为 T 时刻）游戏的胜负结果，1 表示先手玩家获胜，-1 表示后手玩家获胜，0 表示游戏尚未结束，还没有人获胜。 　　第 2 行包含一个整数，表示 T 时刻先手玩家的英雄的生命值。 　　第 3 行包含若干个整数，第一个整数 p 表示 T 时刻先手玩家在战场上存活的随从个数，之后 p 个整数，分别表示这些随从在 T 时刻的生命值（按照从左往右的顺序）。 　　第 4 行和第 5 行与第 2 行和第 3 行类似，只是将玩家从先手玩家换为后手玩家。  **样例输入**  8 summon 1 3 6 summon 2 4 2 end summon 1 4 5 summon 1 2 1 attack 1 2 end attack 1 1  **样例输出**  0 30 1 2 30 1 2  **样例说明**  　　按照样例输入从第 2 行开始逐行的解释如下： 　　1. 先手玩家在位置 1 召唤一个生命值为 6、攻击力为 3 的随从 A，是本方战场上唯一的随从。 　　2. 先手玩家在位置 2 召唤一个生命值为 2、攻击力为 4 的随从 B，出现在随从 A 的右边。 　　3. 先手玩家回合结束。 　　4. 后手玩家在位置 1 召唤一个生命值为 5、攻击力为 4 的随从 C，是本方战场上唯一的随从。 　　5. 后手玩家在位置 1 召唤一个生命值为 1、攻击力为 2 的随从 D，出现在随从 C 的左边。 　　6. 随从 D 攻击随从 B，双方均死亡。 　　7. 后手玩家回合结束。 　　8. 随从 A 攻击随从 C，双方的生命值都降低至 2。  **评测用例规模与约定**  　　\* 操作的个数0 ≤ n ≤ 1000。 　　\* 随从的初始生命值为 1 到 100 的整数，攻击力为 0 到 100 的整数。 　　\* 保证所有操作均合法，包括但不限于： 　　1) 召唤随从的位置一定是合法的，即如果当前本方战场上有 m 个随从，则召唤随从的位置一定在 1 到 m + 1 之间，其中 1 表示战场最左边的位置，m + 1 表示战场最右边的位置。 　　2) 当本方战场有 7 个随从时，不会再召唤新的随从。 　　3) 发起攻击和被攻击的角色一定存在，发起攻击的角色攻击力大于 0。 　　4) 一方英雄如果死亡，就不再会有后续操作。 　　\* 数据约定： 　　前 20% 的评测用例召唤随从的位置都是战场的最右边。 　　前 40% 的评测用例没有 attack 操作。 　　前 60% 的评测用例不会出现随从死亡的情况。 |

火车购票 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-2 |
| 试题名称： | 火车购票 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **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**  　　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。  **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  **样例说明**  　　1) 购2张票，得到座位1、2。 　　2) 购5张票，得到座位6至10。 　　3) 购4张票，得到座位11至14。 　　4) 购2张票，得到座位3、4。  **评测用例规模与约定**  　　对于所有评测用例，1 ≤ n ≤ 100，所有购票数量之和不超过100。 |

最大波动 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201609-1 |
| 试题名称： | 最大波动 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　小明正在利用股票的波动程度来研究股票。小明拿到了一只股票每天收盘时的价格，他想知道，这只股票连续几天的最大波动值是多少，即在这几天中某天收盘价格与前一天收盘价格之差的绝对值最大是多少。  **输入格式**  　　输入的第一行包含了一个整数n，表示小明拿到的收盘价格的连续天数。 　　第二行包含n个正整数，依次表示每天的收盘价格。  **输出格式**  　　输出一个整数，表示这只股票这n天中的最大波动值。  **样例输入**  6 2 5 5 7 3 5  **样例输出**  4  **样例说明**  　　第四天和第五天之间的波动最大，波动值为|3-7|=4。  **评测用例规模与约定**  　　对于所有评测用例，2 ≤ n ≤ 1000。股票每一天的价格为1到10000之间的整数。 |

**问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201612-5 |
| 试题名称： | 卡牌游戏 |
| 时间限制： | 3.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　小Q和小M是游戏数值策划师，他们最近在测试自己新设计的卡牌对战游戏。游戏总共有 n 张卡牌，用 1 到 n 的正整数编号。最开始小Q和小M各会拥有其中的一部分。 　　每一局游戏，小Q和小M都需要从自己拥有的卡牌中选出一张进行对战，获胜的一方会获得双方选出的两张卡牌。游戏会一直进行下去，直到其中一个人获得了所有的卡牌，此时获得所有卡牌的一方赢得了最终的胜利。 　　对于一对特定的卡牌 i 和 j，i 战胜 j 的概率为 Pi, j。此概率与其他事件独立，只与选出的这两张卡牌有关系；每次对战一定会决出胜负，因此有 Pi,j + Pj,i = 1。 　　小Q和小M都没有好好学习博弈论，已经忘了混合决策那套理论。经过商量，他们采取了同一套看起来合理的选牌方式： 　　1. 对于自己的卡牌 i，计算出这张卡牌能赢得对方每张卡牌的概率之和 Si=∑j是对方的卡牌Pi, j； 　　2. 令自己选出卡牌i的概率正比于Si，即选出i的概率为 Si/∑k是自己的卡牌Sk。 　　小M想知道，对于给出的 m 种初始状态，他最终获胜的概率是多少。  **输入格式**  　　从标准输入读入数据。 　　输入的第一行包含两个正整数 n, m 表示卡牌的数量和初始状态的数量。 　　接下来 n-1 行。其中的第 i 行（1 ≤ i < n）包含 n - i 个恰好含有 2 位小数的浮点数；该行的第 j 个（1 ≤ j ≤ n - i）数表示 Pi, i+j。 　　保证上述每个 Pi,j 均是直接调用伪随机数生成函数生成一个 [10, 90] 上的整数，然后除以 100 得到；即可以认为每个数都是从 [0.10, 0.90] 上的所有 2 位小数中，独立等概率取得的。 　　接下来 m 行，每行包含 n 个 0 或 1 的整数，描述一个初始状态。这 n 个数中的第 i 个如果是 1 表示第 i 张牌最初在小M手中，否则表示这张牌在小Q手中。 　　保证询问两两不同。  **输出格式**  　　输出到标准输出。 　　输出 m 行，每行输出一个小数部分长度恰好为5的浮点数，表示小M的每种初始状况最终获胜的概率四舍五入后的结果。 　　你答案中的每个数必须和参考答案完全一样才能获得相应测试点的分数。 　　保证参考答案与真实答案的差值不超过 4 × 10-6。  **样例输入**  3 4 0.46 0.21 0.86 0 0 0 1 1 0 0 0 1 1 1 1  **样例输出**  0.00000 0.83488 0.16512 1.00000  **样例说明**   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | 局面编号 | 小M的牌 | 小Q的牌 | S1 | S2 | S3 | 小M的 Si 之和 | 小Q的 Si 之和 | | x1 | 1 | 2,3 | 0.67 | 0.54 | 0.79 | 0.67 | 1.33 | | x2 | 2 | 1,3 | 0.46 | 1.40 | 0.14 | 1.40 | 0.60 | | x3 | 3 | 1,2 | 0.21 | 0.86 | 0.93 | 0.93 | 1.07 |   　　我们设 P (x) 为当前局面为 x，最终小M获胜的概率，例如 P (x1) 表示小M手里的牌为 1 最终获胜的概率。 　　定义事件 ^x 为将局面 x 中双方手里的牌互换的局面，根据对称性易得 P( ^x)=1 - P (x)。 　　对于全部牌都在小M手里的情况，小M已经赢得了胜利，此种情况下小M获胜概率为 1；相反，牌全部在小Q手里的话，小M获胜概率为 0。 　　对于局面 x1：小M由于只有一张牌 1，因此他必须出这张牌；小Q根据之前的选牌策略，他有 0.54/1.33 的概率出 2，有 0.79/1.33 的概率出 3。若小Q出 2：则小M有 0.46 的概率会赢得这张牌，从而进入局面 ^x3；有 1-0.46 的概率会输掉手中最后一张 1。若小Q出 3：则小M有 0.21 的概率会赢得这张牌，从而进入局面 ^x2；有 1-0.21 的概率会输掉最后一张 1。因此，可以得出 http://118.190.20.162/RequireFile.do?fid=Y6DfN3M5   　　同理，可以求出 P (x2) 和 P (x3) 的表达式： http://118.190.20.162/RequireFile.do?fid=e5hJTQfy   　　注意到 0.2484/1.33， 0.1204/0.60 等数均小于 1，我们可以将三个式子互相不断代入各自的等式右边，得到一个收敛的级数。对这个级数求和就能求出样例的答案。  **样例输入**  2 4 0.34 0 0 1 0 0 1 1 1  **样例输出**  0.00000 0.34000 0.66000 1.00000  **样例输入**  4 8 0.81 0.34 0.73 0.85 0.50 0.22 0 0 0 0 1 0 1 0 0 1 1 0 1 1 1 0 0 0 0 1 1 0 0 1 0 1 0 1 1 1 1 1  **样例输出**  0.00000 0.61095 0.38546 0.80232 0.19768 0.61454 0.38905 1.00000  **样例输入**  5 20 0.45 0.28 0.48 0.59 0.61 0.88 0.66 0.19 0.67 0.11 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 1 0 1 0 0 0 1 1 0 1 0 1 1 0 1 1 1 1 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 0 1 0 1 0 1 1 0 1 0 1 0 1 1 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1  **样例输出**  0.00000 0.15693 0.30992 0.16074 0.35568 0.28030 0.63788 0.31579 0.52478 0.85575 0.14425 0.47522 0.68421 0.36212 0.71970 0.64432 0.83926 0.69008 0.84307 1.00000  **评测用例规模与约定**  　　总共 20 组评测数据。对于第 i 组数据（1 ≤ i ≤ 20）：若 i ≤ 10，n=⌈i / 2⌉；若 i > 10，n = i-5。 |

压缩编码 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201612-4 |
| 试题名称： | 压缩编码 |
| 时间限制： | 3.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　给定一段文字，已知单词a1, a2, …, an出现的频率分别t1, t2, …, tn。可以用01串给这些单词编码，即将每个单词与一个01串对应，使得任何一个单词的编码（对应的01串）不是另一个单词编码的前缀，这种编码称为前缀码。 　　使用前缀码编码一段文字是指将这段文字中的每个单词依次对应到其编码。一段文字经过前缀编码后的长度为： 　　L=a1的编码长度×t1+a2的编码长度×t2+…+ an的编码长度×tn。 　　定义一个前缀编码为字典序编码，指对于1 ≤ i < n，ai的编码（对应的01串）的字典序在ai+1编码之前，即a1, a2, …, an的编码是按字典序升序排列的。 　　例如，文字E A E C D E B C C E C B D B E中， 5个单词A、B、C、D、E出现的频率分别为1, 3, 4, 2, 5，则一种可行的编码方案是A:000, B:001, C:01, D:10, E:11，对应的编码后的01串为1100011011011001010111010011000111，对应的长度L为3×1+3×3+2×4+2×2+2×5=34。 　　在这个例子中，如果使用哈夫曼(Huffman)编码，对应的编码方案是A:000, B:01, C:10, D:001, E:11，虽然最终文字编码后的总长度只有33，但是这个编码不满足字典序编码的性质，比如C的编码的字典序不在D的编码之前。 　　在这个例子中，有些人可能会想的另一个字典序编码是A:000, B:001, C:010, D:011, E:1，编码后的文字长度为35。 　　请找出一个字典序编码，使得文字经过编码后的长度L最小。在输出时，你只需要输出最小的长度L，而不需要输出具体的方案。在上面的例子中，最小的长度L为34。  **输入格式**  　　输入的第一行包含一个整数n，表示单词的数量。 　　第二行包含n个整数，用空格分隔，分别表示a1, a2, …, an出现的频率，即t1, t2, …, tn。请注意a1, a2, …, an具体是什么单词并不影响本题的解，所以没有输入a1, a2, …, an。  **输出格式**  　　输出一个整数，表示文字经过编码后的长度L的最小值。  **样例输入**  5 1 3 4 2 5  **样例输出**  34  **样例说明**  　　这个样例就是问题描述中的例子。如果你得到了35，说明你算得有问题，请自行检查自己的算法而不要怀疑是样例输出写错了。  **评测用例规模与约定**  　　对于30%的评测用例，1 ≤ n ≤ 10，1 ≤ ti ≤ 20； 　　对于60%的评测用例，1 ≤ n ≤ 100，1 ≤ ti ≤ 100； 　　对于100%的评测用例，1 ≤ n ≤ 1000，1 ≤ ti ≤ 10000。 |

权限查询 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201612-3 |
| 试题名称： | 权限查询 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　授权 (authorization) 是各类业务系统不可缺少的组成部分，系统用户通过授权机制获得系统中各个模块的操作权限。 　　本题中的授权机制是这样设计的：每位用户具有若干角色，每种角色具有若干权限。例如，用户 david 具有 manager 角色，manager 角色有 crm:2 权限，则用户 david 具有 crm:2 权限，也就是 crm 类权限的第 2 等级的权限。 　　具体地，用户名和角色名称都是由小写字母组成的字符串，长度不超过 32。权限分为分等级权限和不分等级权限两大类。分等级权限由权限类名和权限等级构成，中间用冒号“:”分隔。其中权限类名也是由小写字母组成的字符串，长度不超过 32。权限等级是一位数字，从 0 到 9，数字越大表示权限等级越高。系统规定如果用户具有某类某一等级的权限，那么他也将自动具有该类更低等级的权限。例如在上面的例子中，除 crm:2 外，用户 david 也具有 crm:1 和 crm:0 权限。不分等级权限在描述权限时只有权限类名，没有权限等级（也没有用于分隔的冒号）。 　　给出系统中用户、角色和权限的描述信息，你的程序需要回答多个关于用户和权限的查询。查询可分为以下几类： 　　\* 不分等级权限的查询：如果权限本身是不分等级的，则查询时不指定等级，返回是否具有该权限； 　　\* 分等级权限的带等级查询：如果权限本身分等级，查询也带等级，则返回是否具有该类的该等级权限； 　　\* 分等级权限的不带等级查询：如果权限本身分等级，查询不带等级，则返回具有该类权限的等级；如果不具有该类的任何等级权限，则返回“否”。  **输入格式**  　　输入第一行是一个正整数 p，表示不同的权限类别的数量。紧接着的 p 行被称为 P 段，每行一个字符串，描述各个权限。对于分等级权限，格式为 <category>:<level>，其中 <category> 是权限类名，<level> 是该类权限的最高等级。对于不分等级权限，字符串只包含权限类名。 　　接下来一行是一个正整数 r，表示不同的角色数量。紧接着的 r 行被称为 R 段，每行描述一种角色，格式为 　　<role> <s> <privilege 1> <privilege 2> ... <privilege s> 　　其中 <role> 是角色名称，<s> 表示该角色具有多少种权限。后面 <s> 个字符串描述该角色具有的权限，格式同 P 段。 　　接下来一行是一个正整数 u，表示用户数量。紧接着的 u 行被称为 U 段，每行描述一个用户，格式为 　　<user> <t> <role 1> <role 2> ... <role t> 　　其中 <user> 是用户名，<t> 表示该用户具有多少种角色。后面 <t> 个字符串描述该用户具有的角色。 　　接下来一行是一个正整数 q，表示权限查询的数量。紧接着的 q 行被称为 Q 段，每行描述一个授权查询，格式为 <user> <privilege>，表示查询用户 <user> 是否具有 <privilege> 权限。如果查询的权限是分等级权限，则查询中的 <privilege> 可指定等级，表示查询该用户是否具有该等级的权限；也可以不指定等级，表示查询该用户具有该权限的等级。对于不分等级权限，只能查询该用户是否具有该权限，查询中不能指定等级。  **输出格式**  　　输出共 q 行，每行为 false、true，或者一个数字。false 表示相应的用户不具有相应的权限，true 表示相应的用户具有相应的权限。对于分等级权限的不带等级查询，如果具有权限，则结果是一个数字，表示该用户具有该权限的（最高）等级。如果用户不存在，或者查询的权限没有定义，则应该返回 false。  **样例输入**  3 crm:2 git:3 game 4 hr 1 crm:2 it 3 crm:1 git:1 game dev 2 git:3 game qa 1 git:2 3 alice 1 hr bob 2 it qa charlie 1 dev 9 alice game alice crm:2 alice git:0 bob git bob poweroff charlie game charlie crm charlie git:3 malice game  **样例输出**  false true false 2 false true false true false  **样例说明**  　　样例输入描述的场景中，各个用户实际的权限如下： 　　\* 用户 alice 具有 crm:2 权限 　　\* 用户 bob 具有 crm:1、git:2 和 game 权限 　　\* 用户 charlie 具有 git:3 和 game 权限 　　\* 用户 malice 未描述，因此不具有任何权限  **评测用例规模与约定**  　　评测用例规模： 　　\* 1 ≤ p, r, u ≤ 100 　　\* 1 ≤ q ≤ 10 000 　　\* 每个用户具有的角色数不超过 10，每种角色具有的权限种类不超过 10 　　约定： 　　\* 输入保证合法性，包括： 　　1) 角色对应的权限列表（R 段）中的权限都是之前（P 段）出现过的，权限可以重复出现，如果带等级的权限重复出现，以等级最高的为准 　　2) 用户对应的角色列表（U 段）中的角色都是之前（R 段）出现过的，如果多个角色都具有某一分等级权限，以等级最高的为准 　　3) 查询（Q 段）中的用户名和权限类名不保证在之前（U 段和 P 段）出现过 　　\* 前 20% 的评测用例只有一种角色 　　\* 前 50% 的评测用例权限都是不分等级的，查询也都不带等级 |

工资计算 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201612-2 |
| 试题名称： | 工资计算 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　小明的公司每个月给小明发工资，而小明拿到的工资为交完个人所得税之后的工资。假设他一个月的税前工资（扣除五险一金后、未扣税前的工资）为S元，则他应交的个人所得税按如下公式计算： 　　1） 个人所得税起征点为3500元，若S不超过3500，则不交税，3500元以上的部分才计算个人所得税，令A=S-3500元； 　　2） A中不超过1500元的部分，税率3%； 　　3） A中超过1500元未超过4500元的部分，税率10%； 　　4） A中超过4500元未超过9000元的部分，税率20%； 　　5） A中超过9000元未超过35000元的部分，税率25%； 　　6） A中超过35000元未超过55000元的部分，税率30%； 　　7） A中超过55000元未超过80000元的部分，税率35%； 　　8） A中超过80000元的部分，税率45%； 　　例如，如果小明的税前工资为10000元，则A=10000-3500=6500元，其中不超过1500元部分应缴税1500×3%=45元，超过1500元不超过4500元部分应缴税(4500-1500)×10%=300元，超过4500元部分应缴税(6500-4500)×20%=400元。总共缴税745元，税后所得为9255元。 　　已知小明这个月税后所得为T元，请问他的税前工资S是多少元。  **输入格式**  　　输入的第一行包含一个整数T，表示小明的税后所得。所有评测数据保证小明的税前工资为一个整百的数。  **输出格式**  　　输出一个整数S，表示小明的税前工资。  **样例输入**  9255  **样例输出**  10000  **评测用例规模与约定**  　　对于所有评测用例，1 ≤ T ≤ 100000。 |

中间数 **问题描述**

|  |  |
| --- | --- |
| 试题编号： | 201612-1 |
| 试题名称： | 中间数 |
| 时间限制： | 1.0s |
| 内存限制： | 256.0MB |
| 问题描述： | **问题描述**  　　在一个整数序列a1, a2, …, an中，如果存在某个数，大于它的整数数量等于小于它的整数数量，则称其为中间数。在一个序列中，可能存在多个下标不相同的中间数，这些中间数的值是相同的。 　　给定一个整数序列，请找出这个整数序列的中间数的值。  **输入格式**  　　输入的第一行包含了一个整数n，表示整数序列中数的个数。 　　第二行包含n个正整数，依次表示a1, a2, …, an。  **输出格式**  　　如果约定序列的中间数存在，则输出中间数的值，否则输出-1表示不存在中间数。  **样例输入**  6 2 6 5 6 3 5  **样例输出**  5  **样例说明**  　　比5小的数有2个，比5大的数也有2个。  **样例输入**  4 3 4 6 7  **样例输出**  -1  **样例说明**  　　在序列中的4个数都不满足中间数的定义。  **样例输入**  5 3 4 6 6 7  **样例输出**  -1  **样例说明**  　　在序列中的5个数都不满足中间数的定义。  **评测用例规模与约定**  　　对于所有评测用例，1 ≤ n ≤ 1000，1 ≤ ai ≤ 1000。 |