湖南省第七届大学生计算机程序设计竞赛
The Seventh Hunan Collegiate Programming Contest

主办：湖南省教育厅
协办：湖南省高等教育学会计算机教育专业委员会
承办：湖南农业大学

2011 年 9 月 15 日

本次比赛 11 道题目，共 12 页。如有缺页，请立即通知赛场工作人员。
所有题目均采用标准输入输出，请不要读写任何文件。
所有题目的正确输出均是惟一的。你的输出只有和正确输出完全一致时才能通过。

题目 A

一二三

你弟弟刚刚学会写英语的一(one)、二(two)和三(three)。他在纸上写了好些一二三，可惜有些字母写错了。已知每个单词最多有一个字母写错了（单词长度肯定不会错），你能认出他写的啥吗？

输入

第一行为单词的个数（不超过 10）。以下每行为一个单词，单词长度正确，且最多有一个字母写错。所有字母都是小写的。

输出

对于每组测试数据，输出一行，即该单词的阿拉伯数字。输入保证只有一种理解方式。

样例输入	样例输出
3 owe too theee	1 2 3

题目 B

报数游戏

n 个人站成一行玩一个报数游戏。所有人从左到右编号为 1 到 *n*。游戏开始时，最左边的人报 1，他右边的人报 2，编号为 3 的人报 3，等等。当编号为 *n* 的人（即最右边的人）报完 *n* 之后，轮到他左边的人（即编号为 *n*-1 的人）报 *n*+1，然后编号为 *n*-2 的人报 *n*+2，以此类推。当最左边的人再次报数之后，报数方向又变成从左到右，依次类推。

为了防止游戏太无聊，报数时有一个特例：如果应该报的数包含数字 7 或者是 7 的倍数，他应当用拍手代替报数。下表是 *n*=4 的报数情况（X 表示拍手）。当编号为 3 的人第 4 次拍手的时候，他实际上数到了 35。

人	1	2	3	4	3	2	1	2	3
报数	1	2	3	4	5	6	X	8	9
人	4	3	2	1	2	3	4	3	2
报数	10	11	12	13	X	15	16	X	18
人	1	2	3	4	3	2	1	2	3
报数	19	20	X	22	23	24	25	26	X
人	4	3	2	1	2	3	4	3	2
报数	X	29	30	31	32	33	34	X	36

给定 *n*，*m* 和 *k*，你的任务是计算当编号为 *m* 的人第 *k* 次拍手时，他实际上数到了几。

输入

输入包含不超过 10 组数据。每组数据占一行，包含三个整数 *n*，*m* 和 *k*（2<=*n*<=100, 1<=*m*<=*n*, 1<=*k*<=100）。输入结束标志为 *n*=*m*=*k*=0。

输出

对于每组数据，输出一行，即编号为 *m* 的人第 *k* 次拍手时，他实际上数到的那个整数。

样例输入	样例输出
4 3 1 4 3 2 4 3 3 4 3 4 0 0 0	17 21 27 35

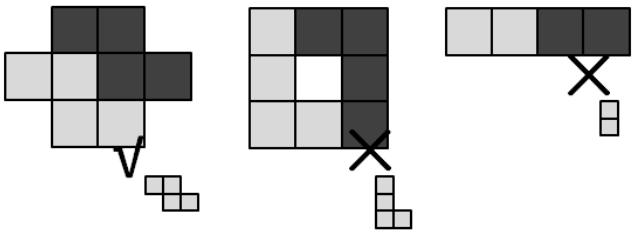
题目 C

多连块拼图

多连块是指由多个等大正方形边与边连接而成的平面连通图形。

-- 维基百科

给一个大多连块和小多连块，你的任务是判断大多连块是否可以由两个这样的小多连块拼成。小多连块只能平移，不能旋转或者翻转。两个小多连块不得重叠。左下图是一个合法的拼法，但右边两幅图都非法。中间那幅图的问题在于其中一个小多连块旋转了，而右图更离谱：拼在一起的那两个多连块根本就不是那个给定的小多连块（给定的小多连块画在右下方）。



输入

输入最多包含 20 组测试数据。每组数据第一行为两个整数 n 和 m($1<=m<=n<=10$)。以下 n 行描述大多连块，其中每行恰好包含 n 个字符*或者.，其中*表示属于多连块，.表示不属于。以下 m 行为小多连块，格式同大多连块。输入保证是合法的多连块（注意，多连块至少包含一个正方形）。输入结束标志为 $n=m=0$。

输出

对于每组测试数据，如果可以拼成，输出 1，否则输出 0。

样例输入	样例输出
4 3 .**. **** .**. .... **. .** ... 3 3 *** *.* *** *.. *.. **. 4 2 **** .... *. *. 0 0	1 0 0

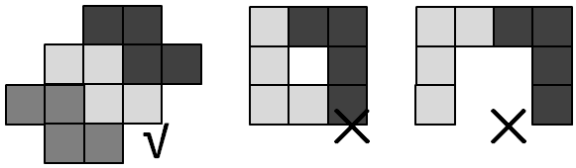
题目 D

多连块分解

多连块是指由多个等大正方形边与边连接而成的平面连通图形。

-- 维基百科

给定一个大多连块，你的任务是把它分解成至少两个全等（不能翻转或者旋转）的小多连块。下面的左图是一个合法的分解，右边两幅图不合法。中间那幅图的问题在于其中一个小多连块旋转了，而右图的问题在于其中一个小多连块翻转了。分解出的小多连块数量越少越好。注意：本题一定有解，因为至少可以分解成一大堆单位正方形。



输入

输入最多包含 30 组测试数据。每组数据第一行为一个整数 n(1<=n<=10)。以下 n 行描述大多连块，其中每行恰好包含 n 个字符*或者.，其中*表示属于多连块，.表示不属于。输入保证是合法的多连块。输入多连块至少包含一个正方形，至多包含二十个正方形。输入结束标志为 n=0。

输出

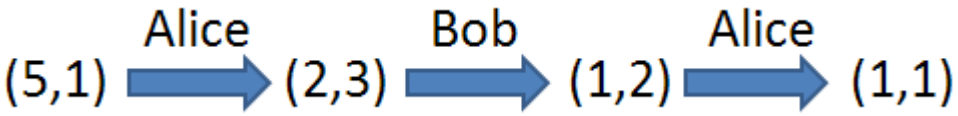
对于每组数据，输出大多连块的分解方案。每个正方形用一个大写字母表示它所在的小多连块。不同的小多连块应有不同的字母表示。如果有多组解，输出字典序最小的。换句话说，如果我们把输出的 n 行字符串逐行连接一个长字符串（比如样例 2，连接后的字符串为 AABB），这个字符串的字典序应该尽量小。每组测试数据的输出后打印一个空行。

样例输入	样例输出
5 ..**. .**** ****. .**.. ..... 2 ** ** 0	..AA. .AABB AABB. .BB.. ..... AA BB

题目 E

盒子游戏

有两个相同的盒子，其中一个装了 n 个球，另一个装了一个球。Alice 和 Bob 发明了一个游戏，规则如下：Alice 和 Bob 轮流操作，Alice 先操作。每次操作时，游戏者先看看哪个盒子里的球的数目比较少，然后清空这个盒子（盒子里的球直接扔掉），然后把另一个盒子里的球拿一些到这个盒子中，使得两个盒子都至少有一个球。如果一个游戏者无法进行操作，他（她）就输了。下图是一个典型的游戏：



面对两个各装一个球的盒子，Bob 无法继续操作，因此 Alice 获胜。你的任务是找出谁会获胜。假定两人都很聪明，总是采取最优策略。

输入

输入最多包含 300 组测试数据。每组数据仅一行，包含一个整数 n（$2<=n<=10^9$）。输入结束标志为 n=0。

输出

对于每组数据，输出胜者的名字。

样例输入	样例输出
2 3 4 0	Alice Bob Alice

题目 F

打怪升级

对于多数 RPG 游戏来说，除了剧情就是打怪升级。本题的任务是用最短的时间取得所有战斗的胜利。这些战斗必须按照特定的顺序进行，每打赢一场，都可能会获得一些补药，用来提升力量。本题只有两种补药："加 1 药"和"乘 2 药"，分别让你的力量值加 1 和乘以 2。

战斗时间取决于你的力量。每场战斗可以用 6 个参数描述：$p_1, p_2, t_1, t_2, w_1, w_2$。如果你的力量小于 p_1，你将输掉战斗；如果你的力量大于 p_2，需要 t_2 秒赢得战斗；如果力量位于 p_1 和 p_2（包括 p_1 和 p_2），战斗时间从 t_1 线性递减到 t_2。比如 p_1=50，p_2=75，t_1=40，t_2=15，你的力量为 55，则战斗获胜需要 35 秒。注意，战斗时间可能不是整数。最后两个参数 w_1 和 w_2 分别表示战斗胜利后获得的"加 1 药"和"乘 2 药"的数量。注意，你不一定要立刻使用这些补药，可以在需要的时候再用，但不能在战斗中使用补药。

按顺序给出每场战斗的参数，输出赢得所有战斗所需的最短总时间。战斗必须按顺序进行，且不能跳过任何一场战斗。

输入

输入最多包含 25 组测试数据。每组数据第一行为两个整数 n 和 p（$1<=n<=1000$, $1<=p<=100$），即战斗的场数和你的初始力量值。以下 n 行每行 6 个整数 $p_1, p_2, t_1, t_2, w_1, w_2$（$1<=p_1<p_2<=100$, $1<=t_2<t_1<=100$, $0<=w_1,w_2<=10$），按顺序给出各场战斗的参数。输入结束标志为 $n=p=0$。

输出

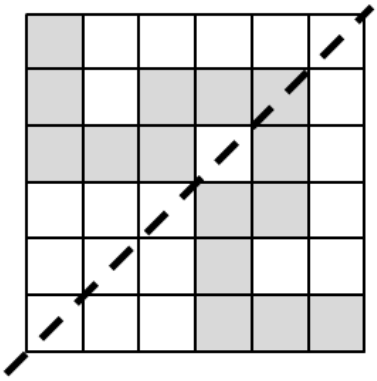
对于每组数据，输出最短总时间（单位：秒），保留两位小数。如果无解，输出"Impossible"（不含引号）。

样例输入	样例输出
1 55 50 75 40 15 10 0 2 55 50 75 40 15 10 0 50 75 40 15 10 0 3 1 1 2 2 1 0 5 1 2 2 1 1 0 1 100 100 1 0 0 1 7 4 15 35 23 0 0 1 1 2 3 2 1 0 0 0 0	35.00 60.00 41.00 31.73 Impossible

题目 G

最优对称路径

给一个 *n* 行 *n* 列的网格，每个格子里有一个 1 到 9 的数字。你需要从左上角走到右下角，其中每一步只能往上、下、左、右四个方向之一走到相邻格子，不能斜着走，也不能走出网格，但可以重复经过一个格子。为了美观，你经过的路径还必须关于“左下-右上”这条对角线对称。下图是一个 6x6 网格上的对称路径。



你的任务是统计所有合法路径中，数字之和最小的路径有多少条。

输入

输入最多包含 25 组测试数据。每组数据第一行为一个整数 *n*（2<=*n*<=100）。以下 *n* 行每行包含 *n* 个 1 到 9 的数字，表示输入网格。输入结束标志为 *n*=0。

输出

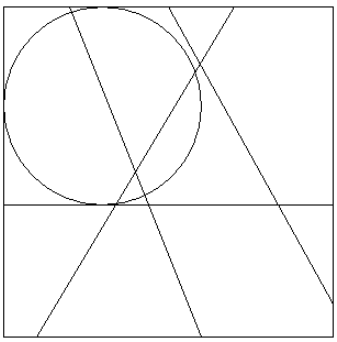
对于每组数据，输出合法路径中，数字之和最小的路径条数除以 1,000,000,009 的余数。

样例输入	样例输出
2 1 1 1 1 3 1 1 1 1 1 1 2 1 1 0	2 3

Problem H
Pieces and Discs

There is a rectangle on the Cartesian plane, whose bottom-left corner is (0,0), top-right corner is (L,W). You draw some line segments to divide the rectangle into pieces. Each line segment connects two points on the boundary of the original rectangle (these two points are guaranteed to be on different sides of the rectangle).

Finally, you draw some discs (a disc is a circle with its interior), and your task is to find out all the pieces each disc is intersecting with (i.e. pieces that have non-zero intersection area with the disc), and output their areas in increasing order. An example picture is shown below:



Input
There will be at most 100 test cases. Each test case begins with four integer n, m, L, W ($1<=n,m<=20$, $1<=L,W<=100$), where n is the number of line segments, m is the number of discs. Each of the next n lines describes a line segment with for integers x_1, y_1, x_2, y_2, that is a segment connecting (x_1,y_1) and (x_2,y_2). These two points are guaranteed to be on different sides of the original rectangle. Each of the last m line contains three integers x, y, R ($0<=x<=L$, $0<=y<=W$, $1<=R<=100$), indicating that the disc is centered at (x, y), whose radius is R. No two segments will be the same. Input is terminated by $n=m=L=W=0$.

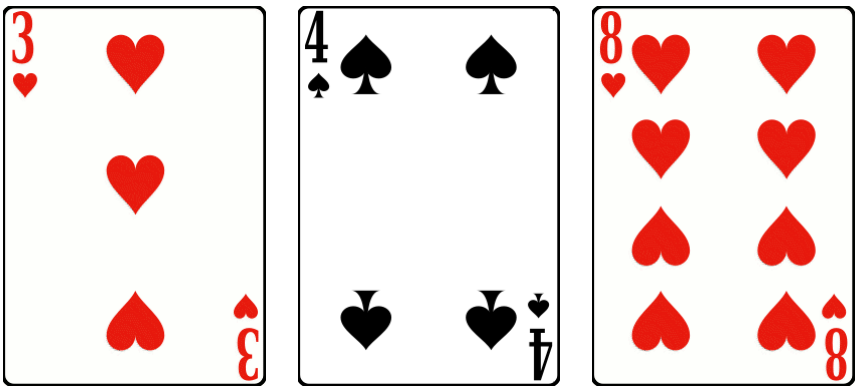
Output
For each disc (same order as in input), print the number of pieces that the disc is intersecting with, and the areas of these pieces in a single line. The areas should be sorted in increasing order, and each area should be rounded to 2 digits after the decimal point. **Print a blank line after each test case.**

Sample Input	Output for the Sample Input
4 1 10 10 0 4 10 4 1 0 7 10 5 10 10 1 2 10 6 0 3 7 3 0 0 0 0	4 0.50 10.03 10.77 18.70

Problem I
Super Poker

I have a set of super poker cards, consisting of an infinite number of cards. For each positive integer *p*, there are exactly four cards whose value is *p*: Spade(S), Heart(H), Club(C) and Diamond(D). There are no cards of other values.

Given two positive integers *n* and *k*, how many ways can you pick up at most *k* cards whose values sum to *n*? For example, if *n*=15 and *k*=3, one way is 3H + 4S + 8H, shown below:



Input
There will be at most 20 test cases, each with two integers *n* and *k* ($1<=n<=10^9$, $1<=k<=10$). The input is terminated by *n*=*k*=0.

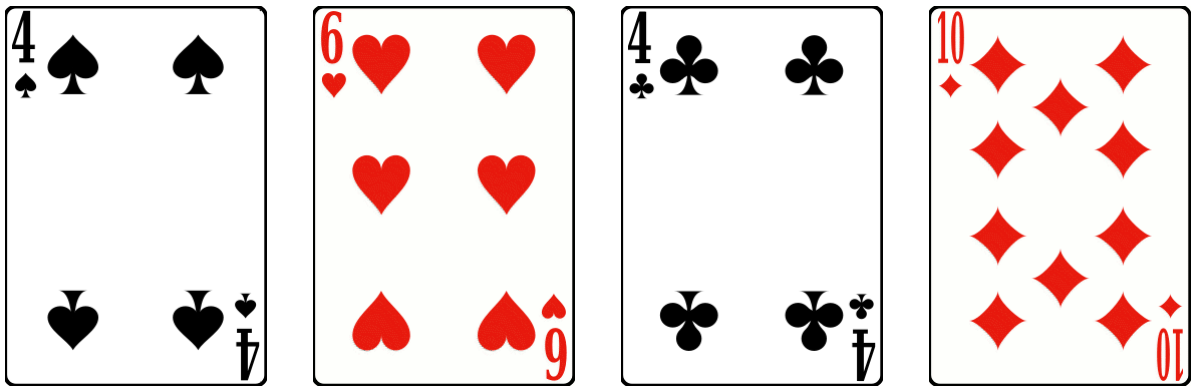
Output
For each test case, print the number of ways, modulo 1,000,000,009.

Sample Input	Output for the Sample Input
2 1 2 2 2 3 50 5 0 0	4 10 10 1823966

Problem J
Super Poker II

I have a set of super poker cards, consisting of an infinite number of cards. For each positive composite integer *p*, there are exactly four cards whose value is *p*: Spade(S), Heart(H), Club(C) and Diamond(D). There are no cards of other values. By "composite integer", we mean integers that have more than 2 divisors. For example, 6 is a composite integer, since it has 4 divisors: 1, 2, 3, 6; 7 is not a composite number, since 7 only has 2 divisors: 1 and 7. Note that 1 is not composite (it has only 1 divisor).

Given a positive integer *n*, how many ways can you pick up exactly one card from each suit (i.e. exactly one spade card, one heart card, one club card and one diamond card), so that the card values sum to *n*? For example, if *n*=24, one way is 4S+6H+4C+10D, shown below:



Unfortunately, some of the cards are lost, but this makes the problem more interesting. To further make the problem even more interesting (and challenging!), I'll give you two other positive integers *a* and *b*, and you need to find out all the answers for *n*=*a*, *n*=*a*+1, …, *n*=*b*.

Input
The input contains at most 25 test cases. Each test case begins with 3 integers *a*, *b* and *c*, where *c* is the number of lost cards. The next line contains *c* strings, representing the lost cards. Each card is formatted as *value*S, *value*H, *value*C or *value*D, where value is a composite integer. No two lost cards are the same. The input is terminated by *a*=*b*=*c*=0. There will be at most one test case where *a*=1, *b*=50,000 and *c*<=10,000. For other test cases, 1<=*a*<=*b*<=100, 0<=*c*<=10.

Output
For each test case, print *b*-*a*+1 integers, one in each line. Since the numbers might be large, you should output each integer modulo 1,000,000. **Print a blank line after each test case.**

Sample Input	Output for the Sample Input
12 20 2 4S 6H 0 0 0	0 0 0 0 0 0 1 0 3

Problem K
RMQ with Shifts

In the traditional RMQ (Range Minimum Query) problem, we have a static array A. Then for each query (L, R) (L<=R), we report the minimum value among A[L], A[L+1], …, A[R]. Note that the indices start from 1, i.e. the left-most element is A[1].

In this problem, the array A is no longer static: we need to support another operation shift(i_1, i_2, i_3, …, i_k) ($i_1<i_2<...<i_k$, k>1): we do a left "circular shift" of A[i_1], A[i_2], …, A[i_k].

For example, if A={6, 2, 4, 8, 5, 1, 4}, then shift(2, 4, 5, 7) yields {6, 8, 4, 5, 4, 1, 2}. After that, shift(1,2) yields {8, 6, 4, 5, 4, 1, 2}.

Input
There will be only one test case, beginning with two integers *n*, *q* (1<=*n*<=100,000, 1<=*q*<=120,000), the number of integers in array A, and the number of operations. The next line contains *n* positive integers not greater than 100,000, the initial elements in array A. Each of the next *q* lines contains an operation. Each operation is formatted as a string having no more than 30 characters, with no space characters inside. All operations are guaranteed to be valid. **Warning:** The dataset is large, better to use faster I/O methods.

Output
For each query, print the minimum value (rather than index) in the requested range.

Sample Input	Output for the Sample Input
7 5 6 2 4 8 5 1 4 query(3,7) shift(2,4,5,7) query(1,4) shift(1,2) query(2,2)	1 4 6