湖南省第十届“蓝狐网络杯”
大学生计算机程序设计竞赛

主办: 湖南省教育厅
协办: 湖南省高教学会计算机教育专业委员会
承办: 湖南科技大学

2014 年 10 月 11 日-12 日

题目 A. 点到圆弧的距离

输入一个点 P 和一条圆弧（圆周的一部分），你的任务是计算 P 到圆弧的最短距离。换句话说，你需要在圆弧上找一个点，到 P 点的距离最小。

提示：请尽量使用精确算法。相比之下，近似算法更难通过本题的数据。

输入格式

输入包含最多 10000 组数据。每组数据包含 8 个整数 x1, y1, x2, y2, x3, y3, xp, yp。圆弧的起点是 A(x1,y1)，经过点 B(x2,y2)，结束位置是 C(x3,y3)。点 P 的位置是 (xp,yp)。输入保证 A, B, C 各不相同且不会共线。上述所有点的坐标绝对值不超过 20。

输出格式

对于每组数据，输出测试点编号和 P 到圆弧的距离，保留三位小数。你的输出和标准输出之间最多能有 0.001 的误差。

样例输入	样例输出
0 0 1 1 2 0 1 -1 3 4 0 5 -3 4 0 1	Case 1: 1.414 Case 2: 4.000

题目 B. 积木玩具

你想在一个长方形棋盘上放一些等大的正方体积木块，搭成一个玩具。玩具的形状可以用一个“高度矩阵”来描述。比如下面的矩阵表示棋盘中心格子的上方的高度为 4，而其他位置的高度为 1。

1	1	1
1	4	1
1	1	1

你有足够多的 1*1 和 1*2 积木块，所以可以搭出很多种不同的玩具。比如下图展示了一种可能的方案，其中字母表示单位正方体，相同字母代表同一个木块。

(a) 顶视图　　(b) 正视图

如果至少用了一个 1*1 木块，这个积木成为普通玩具，否则称为高级玩具。

输入一个高度矩阵，你的任务是统计出有多少种不同的普通玩具和高级玩具。

输入格式

输入包含不超过 20 组数据。每组数据第一行为两个正整数 R, C (1<=R*C<=16)，即棋盘的行数和列数。以下 R 行每行 C 个整数，表示各个格子的高度 h(i,j). (0<=h(i,j)<=20)

输出格式

对于每组数据，输出测试点编号，普通玩具的个数和高级玩具的个数。因为答案可能很大，只需输出这两个值除以 10^9+7 的余数。

样例输入	样例输出
3 3 1 1 1 1 4 1 1 1 1 1 5 1 1 1 1 1 2 2 2 3 4 5	Case 1: 485 2 Case 2: 8 0 Case 3: 2794 12

题目 C. 酷酷的单词

输入一些仅由小写字母组成的单词。你的任务是统计有多少个单词是“酷”的，即每种字母出现的次数都不同。

比如 ada 是酷的，因为 a 出现 2 次，d 出现 1 次，而 1 和 2 不同。再比如，banana 也是酷的，因为 a 出现 3 次，n 出现 2 次，b 出现 1 次。但是，bbacccd 不是酷的，因为 a 和 d 出现的次数相同（均为 1 次）。

输入格式

输入包含不超过 30 组数据。每组数据第一行为单词个数 n (1<=n<=10000)。以下 n 行各包含一个单词，字母个数为 1~30。

输出格式

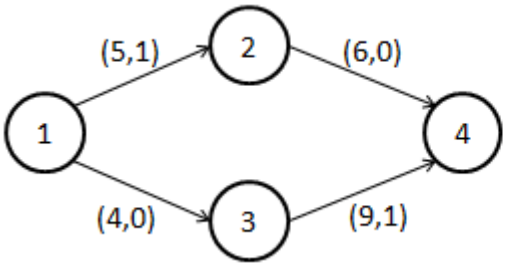
对于每组数据，输出测试点编号和酷单词的个数。

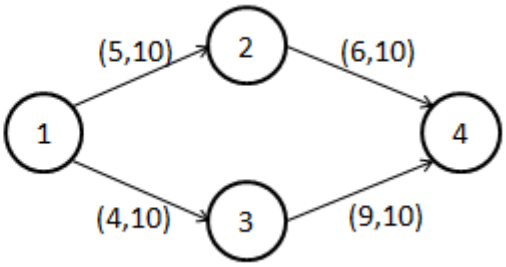
样例输入	样例输出
2 ada bbacccd 2 illness a	Case 1: 1 Case 2: 0

Problem D. Double Shortest Paths

Alice and Bob are walking in an ancient maze with a lot of caves and one-way passages connecting them. They want to go from cave 1 to cave n. All the passages are difficult to pass. Passages are too small for two people to walk through simultaneously, and crossing a passage can make it even more difficult to pass for the next person. We define d_i as the difficulty of crossing passage i for the first time, and a_i as the additional difficulty for the second time (e.g. the second person's difficulty is d_i+a_i).

Your task is to find two (possibly identical) routes for Alice and Bob, so that their total difficulty is minimized.





For example, in figure 1, the best solution is 1->2->4 for both Alice and Bob, but in figure 2, it's better to use 1->2->4 for Alice and 1->3->4 for Bob.

Input

There will be at most 200 test cases. Each case begins with two integers n, m (1<=n<=500, 1<=m<=2000), the number of caves and passages. Each of the following m lines contains four integers u, v, d_i and a_i (1<=u,v<=n, 1<=d_i<=1000, 0<=a_i<=1000). Note that there can be multiple passages connecting the same pair of caves, and even passages connecting a cave and itself.

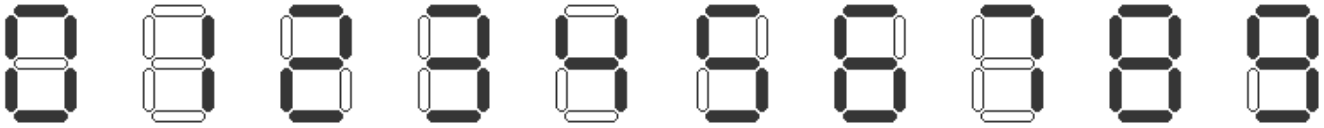
Output

For each test case, print the case number and the minimal total difficulty.

Sample Input	Output for Sample Input
4 4 1 2 5 1 2 4 6 0 1 3 4 0 3 4 9 1 4 4 1 2 5 10 2 4 6 10 1 3 4 10 3 4 9 10	Case 1: 23 Case 2: 24

题目 E. 超大型 LED 显示屏

你是学生会体育部长，负责组织一年一度的校篮球比赛。马上就要决赛了，你希望吸引更多的人来看比赛，因此打算更新一下设备，用一个超大的 LED 屏幕来显示比分。当然，电也不是不要钱的，所以你决定先分析一下往年的比赛，估计一下大概要耗多少电。



如上图，每个数字由 7 条线段组成，每条亮着的线段每秒钟耗电量为 1 个单位。线段不亮的时候不耗电。为了省电，比分不显示前导 0（不过 0 分的时候要显示数字 0）。

你的 LED 显示屏共包含 6 个数字，即双方的比分各有 3 位数。

输入格式

输入包含不超过 100 组数据。每组数据第一行为"START hh:mm:ss"，表示比赛开始时刻为 hh:mm:ss。最后一行为"END hh:mm:ss"，即比赛结束时刻。二者之间至少会有一个 SCORE 信息，格式为"SCORE hh:mm:ss team score"，其中 team 要么是"home"（主场）要么是"guest"（客场）, score 表示得分，为 1,2 或者 3。这些信息保证按照时间从早到晚的顺序排列，且任意两条 SCORE 信息的时刻均不相同。比赛开始时间不会早于 9:00，结束时间不会晚于同一天的 21:00。注意，如果比赛开始时间为 09:00:00，结束时间为 09:00:01，比赛长度为 1 秒钟，而不是 2 秒钟。

输出格式

对于每组数据，输出测试点编号和总耗电量。

样例输入	样例输出
START 09:00:00 SCORE 09:01:05 home 2 SCORE 09:10:07 guest 3 END 09:15:00 START 09:00:00 SCORE 10:00:00 home 1 SCORE 11:00:00 home 1 SCORE 12:00:00 home 1 SCORE 13:00:00 home 1 SCORE 14:00:00 home 1 SCORE 15:00:00 home 1 SCORE 16:00:00 home 1 SCORE 17:00:00 home 1 SCORE 18:00:00 home 1 SCORE 19:00:00 home 1 SCORE 20:00:00 home 1 END 21:00:00	Case 1: 9672 Case 2: 478800

题目 F. 地图的四着色

有一个 R 行 C 列的网格地图，每个国家是一个四连通区域。你的任务是用红，绿，蓝，黄四种颜色给地图着色，使得相邻国家的颜色不同。

一个人着色比较无趣，所以你想请女朋友陪你一起涂——你涂红绿，她涂蓝黄。当然，绅士是不会让让女朋友受累的，所以她最多只需涂 5 个国家（恰好 5 个也行）。

你的任务是统计有多少种着色的方法。注意，每个颜色都至少要用一次。

输入格式

输入包含不超过 100 组数据。每组数据第一行为两个整数 R 和 C (1<=R,C<=20)，即网格的行数和列数。以下 R 行每行 C 个大写字母。相同字母所组成的四连通区域代表一个国家。输入保证国家数目不超过 30，并且**大多数测试点的国家数都比较小**。

输出格式

对于每组数据，输出测试点编号和着色方案数。

样例输入	样例输出
2 4 AABB BBAA 1 5 ABABA 4 7 AABAABB ABBCCCB BBAACBB CCABBAC	Case 1: 24 Case 2: 144 Case 3: 3776

Problem G. Giving directions to the tree

There is a rooted tree, some edges are undirected, while others are directed. We want to change maximum number of undirected edges to directed edges, but we don't want the length of the *longest* directed chain to be increased. Note that undirected edges are not allowed in a directed chain.

For example, if we have a "linear graph" 1->2->3-4, we cannot change 3-4 into 3->4 because the previous longest directed chain (1->2->3) would be extended to 1->2->3->4. However, we can change 3-4 into 3<-4 without extending the longest chain.

Input

There will be at most 1200 test cases. Each test case contains several lines. In each line, the first integer u is the node that is described, followed by its sons, terminated by a zero. The direction of an edge can be from father to son, and can also be from son to father. If the edge is from father to son, then we put a letter ``d'' after that son (meaning that it is a downward edge). If the edge is from son to father, then we put a letter ``u'' after that son (meaning that it is an upward edge). If the edge is undirected then we do not put any letter after the son. Nodes are numbered 1 to n (2<=n<=300) from top to down, left to right (so the first line is always root). Leaves are not given in the input. The test case ends with u=0. **Most test cases have very few nodes.**

Output

For each test case, print the case number, the number of changed edges, followed by the changed edges with directions. Each directed edge is formatted as (i,c), which means the i-th undirected edge is changed to direction c. The undirected edges are numbered from 1, in the same order they appear in the input.

If there are several optimal solutions, print the lexicographically smallest one. When comparing two changes (i1,c1) and (i2,c2) lexicographically, we first compare i1 and i2, if i1 and i2 are equal, we compare c1 and c2. For example (2,u) < (11,d), and (3,d) < (3,u).

Sample Input

```
1 2d 3 0
3 4 5 0
0
1 2d 0
2 3d 0
3 4 0
0
1 2d 0
2 3 0
3 4u 0
0
1 2u 3 0
3 4u 5 0
0
```

Output for Sample Input

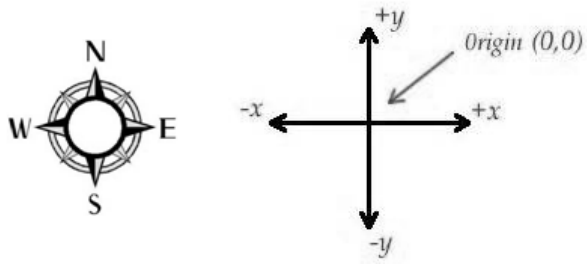
```
Case 1: 3 (1,d) (2,u) (3,u)
Case 2: 1 (1,u)
Case 3: 0
Case 4: 1 (2,u)
```

Problem H. Happy Robot

A robot is moving from (0,0) according to a command sequence. Each character in the sequence is command:

- L: turn left
- R: turn right
- F: go forward one step

Interestingly, the command sequence contains some wildcard character "?". The robot can treat it any one of L, R or F at its own wish, which makes it really happy.



Let (x,y) be the final position of the robot, your task is to find out the minimal/maximal possible value of x and y. Initially the robot is facing east (i.e. facing (1,0) in Cartesian coordinate system). After a left turn it will face north (i.e. facing (0,1)).

Input

There will be at most 1000 test cases. Each case contains a command sequence with no more than 1000 characters.

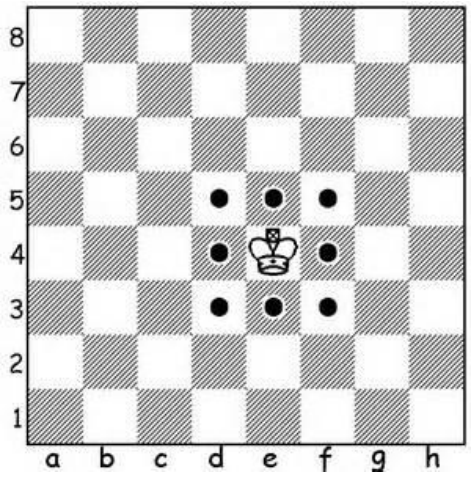
Output

For each test case, print the case number, followed by minimal/maximal possible x (in this order), then the minimal/maximal possible y.

Sample Input	Output for Sample Input
F?F L?? LFFFRF	Case 1: 1 3 -1 1 Case 2: -1 1 0 2 Case 3: 1 1 3 3

题目 I. 残缺的棋盘

在国际象棋里，王是最重要的一个棋子。每一步，王可以往上下左右或者对角线方向移动一步，如下图所示。



给定两个格子 A(r1,c1), B(r2,c2)，你的任务是计算出一个王从 A 到 B 至少需要走多少步。为了避免题目太简单，我们从棋盘里拿掉了一个格子 C(r3,c3)（ABC 保证互不相同），要求王从 A 走到 B 的过程中不能进入格子 C。在本题中，各行从上到下编号为 1~8，各列从左到右编号为 1~8。

输入格式

输入包含不超过 10000 组数据。每组数据包含 6 个整数 r1, c1, r2, c2, r3, c3 (1<=r1, c1, r2, c2, r3, c3<=8). 三个格子 A, B, C 保证各不相同。

输出格式

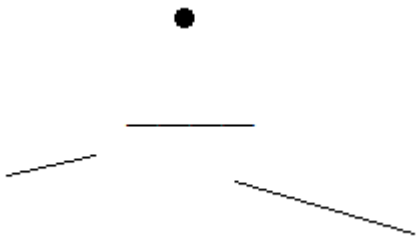
对于每组数据，输出测试点编号和最少步数。

样例输入	样例输出
1 1 8 7 5 6 1 1 3 3 2 2	Case 1: 7 Case 2: 3

Problem J. Just another pachinko-like machine

Like pachinko? Here is another one. It's not exactly a traditional pachinko, but it's also a let-the-ball-hit-things game.

In the machine, there are n non-overlapping non-vertical bars, shown below.



At the i-th step, the ball will be transferred to (x_i, y_i), then start to fall vertically, hopefully it'll hit a bar and earn some scores. A ball who hit the i-th bar will earn a score of s_i. If the ball directly drops on the floor (with y=0), it will not score.

The most interesting part of the machine is: **if the i-th bar is hit during this step, it will disappear at that moment and re-appear after d_i steps**. For example, if a bar with d_i=3 is hit in the 5-th step, then it'll be missing during step 6 and 7, and will re-appear in step 8.

Input

There will be at most 5 test cases. Each test case begins with one integer n ($1<=n<=10^5$), the number of bars. Each of the next lines contains 5 integers x1, y1, x2, y2, s, d ($0<=x1<x2<=10^9$, $1<=y1,y2<=200000$, $1<=s<=1000$, $1<=d<=5$), describing one bar. No two bars can have any common point (i.e. no intersection, can't touch each other etc).

The next line contains b ($1<=b<=10^5$), the number of balls. In the next b lines, the i-th line describes the ball appear in the i-th step. Each line contains two integers (x', y'), that means the ball will appear at (x_i,y_i)=(x' XOR a, y' XOR a), where a is the current score before the ball falls (which will be zero at the beginning of each test case). It is guaranteed that x_i and y_i are non-negative integers and will not be precisely on a bar.

Output

For each test case, print the case number in the first line and the scores after each step. There should be one empty line after each test case.

Sample Input

```
2
0 4 4 4 1 4
2 2 6 2 9 1
5
3 5
2 4
11 15
9 9
16 26
3
0 6 10 7 1 5
2 4 8 3 10 5
4 2 6 2 100 5
4
5 7
4 6
14 12
106 104
```

Output for Sample Input

```
Case 1:
1
10
10
19
20

Case 2:
1
11
111
111
```

Explanation for Sample 1

Step 1:
ball (3,5) will hit the first bar, score=1

Step 2:
ball (3,5) will hit the second bar, score=9

Step 3(bar 2 appear again):
ball (1,5) will hit the ground, score=0

Step 4:
ball (3,3) will hit the second bar, score=9

Step 5(bar 1&2 appear again):
ball (3,9) will hit the first bar again, score=1

Problem K. Kick the ball!

"A penalty shoot-out (officially kicks from the penalty mark) is a method of determining the winner of an association football (soccer) match that is drawn after the regulation playing time and any applicable extra time periods have been played. In a penalty shoot-out, each team takes turns attempting a specified number of shots from the penalty mark (usually 5) that are only defended by the opposing team's goalkeeper, with the team scoring the most goals being declared the winner."

-- wikipedia

The game finally comes to the shoot-out. What will the final result be? "1-3!" You took a wild guess. But what is the probability that your guess is correct?

In this problem, team A kicks first (which is determined by a coin toss, as usual), both teams will attempt at most 5 shots (after all the 10 shots, the game may end in draw again), but the game will end as soon as the winner is already determined. For example, after the first 8 kicks the score is 3-2 (left side is team A's score, right side is team B), then if the 9-th kick is a goal, the game will end immediately with score 4-2, because even team B got its last kick, it still loses for sure. Another example: if all the first 9 kicks are goals, the last kick (from team B) will still be performed, because although team B cannot win, the result might be a "draw", which is better than "lose".

Input

There will be at most 100 test cases. Each case contains two lines. The first line contains 10 floating numbers. The first 5 numbers are the goal probability of the players in team A (player 1 will shoot first, etc), the next 5 numbers are the goal probabilities of the players in team B. Each probability will have exactly one digit after the decimal point. The second line contains your guess, in the format of scoreA-scoreB. 0<=scoreA,scoreB<=5.

Output

For each test case, print the case number and the probability (in percentage) that your wild guess is correct, to 2 decimal places. An absolute error of 0.01% will be ignored.

Sample Input	Output for Sample Input
0.4 0.7 0.7 0.6 0.5 0.8 0.9 0.7 0.2 0.8 1-3 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 2-0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 2-0 0.4 0.7 0.7 0.6 0.5 0.8 0.9 0.7 0.2 0.8 5-5 0.4 0.7 0.7 0.6 0.5 0.8 0.9 0.7 0.2 0.8 4-2	Case 1: 6.98% Case 2: 100.00% Case 3: 0.00% Case 4: 0.47% Case 5: 9.73%