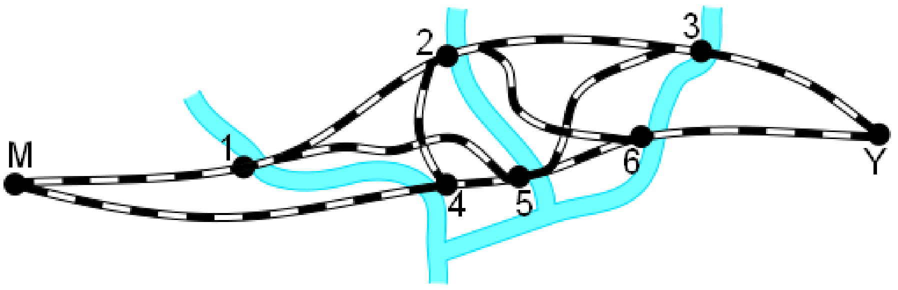
Problem A. Transsib

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	64 mebibytes

The route of the Moscow–Vladivostok "Rossiya" passenger train is considered to be the main route of the Trans-Siberian Railway. Its itinerary includes Nizhny Novgorod, Kirov, Perm, and Yekaterinburg. The train goes from Moscow to Yekaterinburg in 25 h 41 min. The "Ural" train follows the southern line of the Trans-Siberian Railway through Kazan and completes the journey in 25 h 25 min. It is impossible to go by train from Moscow to Yekaterinburg in a shorter time.



This is the scheme of the railroads between Moscow (`M`) and Yekaterinburg (`Y`). It is seen that the routes of the trains "Rossiya" (the upper line in the scheme) and "Ural" (the lower line) are crossed by major rivers: Volga, Vyatka, and Kama. The former train crosses them at Nizhny Novgorod (1), Kotelnich (2), and Perm (3), respectively. The latter train crosses the rivers in 35 km west of Kazan (4), in Vyatskie Polyany (5), and in Sarapul (6). The scheme also shows direct lines connecting some of these cities.

In addition to passenger trains, there are also goods trains following these railroads. They can take one of the four routes:

1. Moscow – Nizhny Novgorod – Kotelnich – Sarapul – Yekaterinburg
2. Moscow – Kazan – Vyatskie Polyany – Sarapul – Yekaterinburg
3. Moscow – Kazan – Kotelnich – Perm – Yekaterinburg
4. Moscow – Nizhny Novgorod – Vyatskie Polyany – Perm – Yekaterinburg

Minister of Railway Transport wants to organize the goods train service in such a way that the freight flow from Moscow to Yekaterinburg be as much as possible. He knows that the bridges across the rivers shown in the scheme are the "bottlenecks" for the trains. For each bridge, the carrying capacity is known, i.e., the amount of freight that can be taken through the bridge in a day. Help the minister solve the problem.

Input

The only input line contains the carrying capacities of the bridges in Nizhny Novgorod, in Kotelnich, in Perm, near Kazan, in Vyatskie Polyany, and in Sarapul. These are integers in the range from 1 to 10^9.

Output

Find the daily amount of freight sent from Moscow along each of the routes specified above so that the total freight flow from Moscow to Yekaterinburg be maximal. Output these four numbers accurate to 10^{-3}, separating them with a space. If the problem has several solutions, output any of them.

Example

standard input	standard output
`70 30 60 100 20 50`	`20.000 10.000 10.000 10.000`

Problem B. Death Star 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 mebibytes

A long time ago in a galaxy far, far away…

The battle space station "Death star" was designed even before the Clone wars. Many years later, it was given to the Empire to control the Outer Rim Territories. The "Death star" was about 100 miles in diameter, was equipped with a graviton gun, capable of destroying whole planets, and could carry a few thousands of space fighters on board. The "Death star" was supposed to terrify the population and to absolutely exclude any possibility of resisting the power of the Empire.

After the first "Death star" had been destroyed by the rebels, the construction of a new, even more deadly model started. The new model, as the first one, has a ball shape and can translationally move in N-dimensional space. It is equipped with M firmly anchored krypton engines. If the i-th engine is provided with X units of energy, its contribution to the j-th coordinate of the jet thrust vector will be equal to $A_{ij} \cdot X$. Note that the engines are bidirectional, so supplying a negative X just means using it to thrust in the opposite direction with $|X|$ units of energy. The resulting jet thrust vector is equal to the sum of contributions of each of M engines.

Before the beginning of the movement a special navigational module calculates the required coordinates of the jet thrust vector $(b_1, b_2, \ldots, b_N)$. Your program should calculate how much units of energy should be provided to each of the engines in such a way that the length of the vector of difference between the resulting jet thrust and the required jet thrust will be minimal. If the answer is ambiguous, the sum of squares of the quantity of energy provided to the engines should also be minimized.

Input

The first line contains two integers N and M separated by a space ($1 \le N, M \le 100$). The following M lines with N numbers in each line represent the matrix A_{ij}. The last line contains N numbers b_j — the coordinates of required jet thrust vector. All A_{ij} and b_j are integers with absolute values not exceeding 100.

Output

Output M real numbers $X_1, \ldots, X_M$ precise up to 5 digits after the decimal point. X_i should be equal to the quantity of energy provided to i-th engine. If there is more than one answer, you can output any one.

standard input	standard output

4 3
2 3 -2 1
-1 2 1 3
4 2 3 -2
3 13 -9 13
	4.00000 2.00000 -1.00000

Problem C. Geometrical Dreams

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 mebibytes

There is a polygon $A_1A_2...A_N$ (the vertices A_i are numbered in clockwise order). On each side A_iA_{i+1} an isosceles triangle $A_iM_iA_{i+1}$ is built on the outer side of the polygon ($M_iA_i = M_iA_{i+1}$). The angle $A_iM_iA_{i+1}$ is equal to α_i. Here we assume that $A_{N+1} = A_1$. The set of angles α_i satisfies a condition that the sum of angles in any of its nonempty subsets is not aliquot to 360 degrees. You are given N, coordinates of vertices M_i and angles α_i (measured in degrees). Write a program, which restores coordinates of the polygon vertices.

Input

The first line contains an integer $N (3 \le N \le 50)$. The next N lines contain pairs of real numbers x_i, y_i which are coordinates of points $M_i (100 \le x_i, y_i \le 100)$. And the last N lines of the input consist of degree values of angles α_i. All real numbers in the input contain at most 2 digits after decimal point.

Output

Output N lines with points coordinates, i-th line should contain the coordinates of A_i. Coordinates must be accurate to 2 digits after decimal point. You may assume that solution always exists.

Example

standard input	standard output

3
0 2
3 3
2 0
90
90
90
	1 1
	1 3
	3 1

Problem D. Paul's Salads

Input file:	*standard input*
Output file:	*standard input*
Time limit:	1 second
Memory limit:	64 mebibytes

Paul the Chef loves recursive salads. For example, his favorite salad, "razduv" (*blow up*) consists of 30% cucumbers, 20% tomatoes and 50% salad "razgon" (*drive on*). The "razgon" salad consists of 40% bread, 20% cucumbers and 40% "razduv" salad.

For each of Paul's favorite salads, we know its formula as a percentage of all its components. A component of a salad could be a basic ingredient or another salad. Your task is to find the formulas of Paul's salads where all components are basic ingredients.

Input

Input data contains formulas of Paul's salads. The names of the salads and basic ingredients consist of at most 10 small English letters. If some component has a formula, then it is a salad, in the other case it is a basic ingredient. All formulas are listed in the lexicographical order of salad's names. The components in each formula are also listed in the lexicographical order. All the percentages are integers from 1 to 100, the sum of percentages in one formula is equal to 100. Overall, there are at most 10 distinct salads and at most 10 distinct basic ingredients. Follow the input data format from the sample tests.

Output

Output formulas of Paul's salads depending only on the basic ingredients, not on the other salads. Formulas and the components in each formula should be listed in the lexicographical order. Percentages should be output with an absolute or relative error no more than 10^{-3}. The sum of percentages in each formula should be equal to 100. If a salad does not contain some basic ingredient, the name of this ingredient should be output with a percentage 0. We guarantee that there exists exactly one solution for our input data.

Examples

standard input
```
razduv : cucumber 30 razgon 50 tomato 20
razgon : bread 40 cucumber 20 razduv 40
```
standard input
```
razduv : bread 25.0 cucumber 50.0 tomato 25.0
razgon : bread 50.0 cucumber 40.0 tomato 10.0
```
standard input
```
razduv : cucumber 100
razgon : tomato 100
```
standard input
```
razduv : cucumber 100.0 tomato 0
razgon : cucumber 0 tomato 100.0
```

Problem E. Space Poker 2

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	64 mebibytes

At the First Lunar Casino there are new rules for playing space poker. A pack contains N cards. A player pays the croupier 20 space rubles and chooses a card from his pack. The croupier chooses a card from his pack. If the selected cards are identical, then the player wins 1000 rubles. If the cards are different, then he/she may win a sum not exceeding 10 rubles. This sum depends on the cards that have been chosen and is specied in the prize-table. Your task is to write a program that tells the croupier which card to choose in order to maximize the mean profit of the casino. It is supposed that all players know the prize-table by heart and choose their strategies in the best possible way. If you ask any mathematician how to write such a program, you'll learn that this is a classical minimax problem, which was solved decades ago, and that everyone should know about matrix games with mixed strategies. The croupier should choose a card at random with such probabilities that the mathematical expectation of the gain is independent of the card chosen by the player. And if you ask about a mathematical expectation, this is a kind of a mean value: the sum of possible gains multiplied by probabilities of getting them.

Input

The first line contains the number of cards $N (1 < N \leq 100)$. Then follows the prize-table. Each row of the table shows possible gains of a player who has chosen the card corresponding to the number of the row. The number of the column corresponds to the card chosen by the croupier. The numbers in the main diagonal of the table are 1000, other numbers are in the range from 0 to 10.

Output

You should output with accuracy to the fifth digit the probabilities with which the croupier should choose a card.

Example

standard input	standard output

3
1000 10 10
0 1000 1
5 3 1000
	0.32986
	0.33623
	0.33391

Problem F. Kirchhoff's Law

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 mebibytes

Vasya's dad, as you know, is good in math but the son instead of following his father's steps, studies physics at school. Once Vasya asked his father to help him to solve a simple problem — to find out the resistance of the resistors system. Dad answered him: "Here is nothing to think about. You are to numerate the conductors nodes, to write the $I = U/R$ law for each conductor. T hen remember that the sum of currents at each of the nodes except the first and the last equals to zero, you may assume potential in the first node equal to one and in the last node — zero. Then you get a simple system of linear equations. Hence you find potentials in the intermediate nodes and currents between all the nodes. It's left only to divide the voltage by the total current from the first node and..."

But Vasya is not good in math, so his dad was to write the system of equations himself and to solve it. Vasya looks at the end of the book of problems and says that there is another answer. Dad tried to solve the problem again and got another answer. Vasya looked at the answer and said again: "Wrong". Dad resolves the problem for the third time and Vasya holds his own. Dad got tired to solve the problem manually and he decided to use a computer seeing that the students of mathematical department of the Ural State university are ready to write the required program.

Input

The first line contains integers N and M; N is a number of nodes in the circuit ($2 < N \leq 20$), M is the number of resistors ($0 \leq M < 1000$). Each of the next M lines consists of three integers A_i, B_i and R_i — description of a resistor that has resistance R_i connecting the nodes A_i and B_i ($1 \leq A_i < B_i \leq N; 1 \leq R_i \leq 1000$). There may be many resistors between two nodes.

Output

Your task is to output the total resistance between the nodes 1 and N rounded within two digits after a decimal points.

Example

standard input	standard output

4 5
1 2 15
2 4 5
1 3 10
3 4 10
2 3 1
	9.40

Problem G. Integer-valued Complex Determinant

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 mebibytes

Let $\mathbb{Z}[i]$ be a set of Gaussian integers, that is, a set of complex numbers with integer components. Let $a, b, q, r \in \mathbb{Z}[i]$, $a = bq + r$, $|r| < |b|$. Then r is a *remainder* of division of a by b.

Let $p \in \mathbb{Z}[i]$ and $X = (x_{ij})$ be a matrix of size $n \times n$ with elements from $\mathbb{Z}[i]$. Your goal is to calculate the remainder of division of determinant of X by p. Remember, that the determinant of X is equal to

$$\sum_{\pi} sign(\pi) \cdot x_{1\pi(1)} \cdot x_{2\pi(2)} \dots x_{n\pi(n)},$$

where the summation is taken over the set of all permutations π of n elements. Here addition and multiplication are the usual addition and multiplication of complex numbers.

Input

The first line contains an integer n ($1 \le n \le 50$). Each of the next n lines contains n space-separated complex numbers which are the elements of X. The last line contains a non-zero complex number p. Complex number is denoted by its real and imaginary parts, separated with space. All components of all complex numbers don't exceed 10 000 in their absolute value.

Output

Output real and imaginary parts of the remainder of division of determinant of X by p. If there are several possible answers, output any of them. It is guaranteed that the answer exists.

Example

standard input	standard output

`2`
`2 0 -7 0`
`1 0 0 -1`
`3 1`
	`3 0`

Problem H. Anti-equations

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 mebibytes

There is a system of linear anti-equations modulo 3:

$$\begin{array}{rcll} A_{11} \cdot x_1 + A_{12} \cdot x_2 \ + \ldots + & A_{1n} \cdot x_n & \neq & B_1 \mod 3 \\ A_{21} \cdot x_1 + A_{22} \cdot x_2 \ + \ldots + & A_{2n} \cdot x_n & \neq & B_2 \mod 3 \\ & \ldots & & \\ A_{k1} \cdot x_1 + A_{k2} \cdot x_2 \ + \ldots + & A_{kn} \cdot x_n & \neq & B_k \mod 3 \end{array}$$

Find the number of different solutions of this system assuming x_i are integers and $0 \leq x_i \leq 2$.

Input

First line contains two integers k and n ($1 \leq k, n \leq 30$): the amount of anti-equations and variables. Each of the next k lines contains $n+1$ integers — values A_{ij} and B_i ($0 \leq A_{ij}, B_i \leq 2$).

Output

Output a single integer — the number of solutions.

Examples

standard input	standard output

3 3
2 2 1 1
2 1 0 0
1 2 2 2
	8

4 3
2 2 1 1
2 1 0 0
1 2 2 2
1 0 1 2
	6

Problem I. Heroes 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 mebibytes

Not long ago the game ¡¡Heroes of Keyboard and Mouse 2¿¿ was released.

The 4D-Action essence be the following: A number of cities are placed in a four-dimensional space. Every city is located in a certain point. The player can build new cities. And rob ¡¡corovans¿¿.

At all times all cities are surrounded by a single connected wall of a finite size. The wall splits the game space into two parts: everything thats within it and all thats outside. The wall is built so that:

1. All the cities are within the wall.
2. There is a straight path between any two points within the wall, not crossing the wall.
3. The set of points within the wall is minimal as long as the conditions 1 and 2 are met.

When a new city is built, the wall has to be rebuilt if the city lies outside it. Your task is to define whether wall reconfiguration is necessary for every newly built town. It is guaranteed that a new city is always built either outside of the existing wall or within it. Moreover, the distance from the new city to the wall is always greater than 10^{-3}. No two cities occupy the same point.

Input

The game begins with five cities with the coordinates (x_1, y_1, z_1, w_1), (x_2, y_2, z_2, w_2), $\ldots$, (x_5, y_5, z_5, w_5), which are defined in the first five lines of the input file. The initial 4D volume within the wall is strictly positive. The second line contains an integer N — the number of newly built cities ($1 \le N \le 800$). Each of the following N lines contains four integers — the coordinates of a newly built city. The absolute value of each coordinate does not exceed 5000.

Output

The output file must contain N lines. The K-th line must contain the word “`Rebuild`, if wall reconfiguration is necessary after building K-th town, and the word “`Ignore`” otherwise ($1 \le K \le N$).

Example

standard input	standard output

0 0 0 0
8 0 0 0
0 8 0 0
0 0 8 0
0 0 0 8
5
1 2 2 2
2 2 3 2
8 8 8 8
3 5 3 4
-1 3 7 2
	Ignore
	Rebuild
	Rebuild
	Ignore
	Rebuild

Note

Initially five cities are placed at the vertices of a coordinate simplex with the length 8 along the axes. The wall is exactly the boundary of the simplex. The equation of the large hyperplane of the simplex is the following: $x + y + z + w = 8$. As it can be seen from the equation, the first newly built city belongs to the simplex and doesn't require wall reconfiguration, and the second city lies outside the simplex and the wall must be rebuilt. Once the wall has been rebuilt, the set of interior points consists of two adjacent simplexes. When the third city is built, the wall is reconfigured, and after that the set of interior points also consists of two simplexes. The fourth city belongs to this set, and the fifth apparently lies outside of it.