Problem 1. Casino

Input file:	`casino.in`
Output file:	`casino.out`
Time limit:	4 seconds
Memory limit:	256 megabytes

The new Greek casino "Las Figas" is opening in Las Vegas. Andrew is going to a celebration of the opening of the casino. Of course he is going to play roulette.

The brand new roulette game is going to be held at "Las Figas". The roulette has s sectors, and there are m possible types of bets possible. The i-th type of bet is the following. The player bets one chip for some p_i sectors of the roulette and if one of the selected sectors wins the player gets w_i chips in addition to her chip. If none of them wins, the player loses the chip he bets. Each sector of roulette has the same probability of winning equal to $1/s$.

Andrew uses the following strategy. He starts with n chips. He chooses one of the types of bets and bets a chip. If at some moment he has strictly greater than n chips he considers himself winning and walks away from the roulette. If he ends up with no chips he loses. He uses such strategy of choosing which bet type to use that maximizes the probability of winning.

Given the possible types of bets and n find the maximal possible probability of winning that Andrew can achieve.

Input

The first line of the input file contains three integer numbers: n, m and s ($1 \le n \le 50$, $1 \le m \le 20$, $1 \le s \le 100$). The following m lines contain two integer numbers each: p_i and w_i ($1 \le p_i \le s$, $1 \le w_i \le 50$).

Output

Output one real number — the maximal possible probability of winning. Your answer must be accurate up to 10^{-7}.

Examples

casino.in	casino.out
2 1 2 1 1	0.6666666666666667
10 8 38 18 1 12 2 9 3 6 5 4 8 3 11 2 17 1 35	0.8919856914626223

Problem 2. Chip Reconstruction

Input file:	`chip.in`
Output file:	`chip.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

The team of professor Chi Hua Hua is working for Nanoprocessor Factory in Zhozhozho province. During the daytime the factory produces nanoprocessors for HZCh company, but during the night they make pirate copies of the processor to sell it under their own trademark. Unfortunately, the main production line is well guarded by HZCh representatives, so the additional line is running in the factory basement.

Recently the production of a new nanoprocessor has started at the factory. The team of the professor was given an order to reconstruct the processor structure in order to start the production of its duplicate on a pirate line.

The chip of the nanoprocessor is positioned on an $m \times n$ grid. There are two nanoelements used in the processor: botvizator and unquantizer. Botvizator has a size of 1×1 and unquantizer has size of $h \times 1$. Unquantizers are always positioned on a grid in a way that their long side is parallel to vertical side of the grid (which has length m), so they are completely located in grid columns. Each grid cell is occupied by some element.

Unfortunately the exact structure of nanoprocessors is not known. All that is known is the amount of botvizators in each grid row and the number of unquantizers in each grid column. In order to start production of nanoprocessors the team of professor Chi must reconstruct its internal structure. You were assigned by the team as a programmer and your task is to write the program that would do so.

Input

The first line of the input file contains m, n and h ($1 \le m, n \le 500$, $1 \le h \le m$). The second line contains m integer numbers a_i ($0 \le a_i \le n$) — the number of botvizators in each row (from top to bottom). The third line contains n integer numbers b_i ($0 \le b_i \le m/h$) — the number of unquantizers in each column (from left to right).

Output

Output the reconstructed chip. Output m lines of n characters. If the corresponding cell is occupied by a botvizator, output '`*`', if it is occupied by an unquantizer, output '`+`' if it's top or bottom cell of the unquantizer, or '`|`' if it is its internal cell. If there are several solutions, output any one.

If it is impossible to reconstruct chip from the given data because it is inconsistent, output "`inconsistent`" at the first line of the output file.

Examples

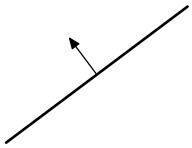
chip.in	chip.out
4 4 3 2 1 1 3 1 1 0 1	+**+ |+*| +|*+ *+**
4 4 3 1 1 1 1 1 1 1 1	inconsistent

Problem 3. Convex Hull

Input file:	`convex.in`
Output file:	`convex.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

The convex hull of a set of points is the minimal convex polygon that contains all the points from the set.

There are several lines on a plane. Each line is moving with its own constant speed of v_i units per second perpendicular to itself.



Consider all their points of intersections (if two lines are parallel they are considered to have no points of intersection even at the moment they coincide). Their convex hull changes as lines move. You have to find the average area of their convex hull on the period from the starting moment till T seconds.

Namely, let $S(t)$ be the area of the convex hull at the moment t. You have to find

$$\frac{1}{T}\int_0^T S(t)dt.$$

Input

The first line of the input file contains n — the number of lines ($3 \le n \le 10$). The following n lines of the input file describe the given lines. Each line is described by four integer numbers x_1, y_1, x_2 and y_2 and one floating point number v ($-20 \le x_1, y_1, x_2, y_2, v \le 20$, $x_1 \ne x_2$ or $y_1 \ne y_2$, $|v| \ge 0.01$). The line described by the given numbers passes through points (x_1, y_1) and (x_2, y_2) at moment 0 and moves with constant speed v in the direction perpendicular to itself. If you stand at point (x_1, y_1) and look at point (x_2, y_2), the line moves towards your right if v is positive and towards your left if v is negative.

We consider the coordinate system where Oy axis is directed to your left if you stand at the origin and look in the direction of Ox axis.

The last line of the input file contains a real number T ($0.01 \le T \le 100$).

There are three lines that are pairwise not parallel. No two lines coincide for a non-zero period of time.

Output

Output one real number — the average area of the convex hull between moments 0 and T. Your answer must be accurate to within 10^{-4} absolute or 10^{-9} relative.

Examples

convex.in	convex.out
4 0 0 0 1 1.0 1 0 1 1 -1.0 0 0 1 0 -1.0 0 1 1 1 1.0 0.2	0.65333333333333333
4 0 0 1 0 0.1 0 0 0 1 0.1 1 0 1 1 0.1 2 0 0 2 -0.3 3.0	1.0237828009

In the first example the convex hull is always a square with the side length $1-2t$. So its area is $(1-2t)^2$, and the average area is

$$\frac{1}{0.2}\int_0^{0.2}(1-2t)^2dt = 5\left(\frac{4}{3}t^3-2t^2+t\right)\Big|_0^{0.2} = \frac{49}{75}.$$

Problem 4. Discussions

Input file:	`discuss.in`
Output file:	`discuss.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

— Parliament is not a place for discussions!

Boris Gryzlov, Russian State Duma Speaker

The Ministry of Bureaucracy is preparing the new law proposal to be presented to the parliament. However, they want the parliament to have a possibility for the discussion, and they don't want this discussion be whether to approve the law, or no. So they are planning to prepare n possible variants of the proposal (actually they will only differ by a couple of phrases), and suggest that parliamentarians choose k of them for a final voting.

The Minister of Bureaucracy knows that the favorite number of the leader of the major faction in the parliament is z. Therefore he would like the number of ways to choose k variants of the law proposal out of n be z. Now he wonders what is the minimal number of proposals n the Ministry needs to prepare so that there was such k.

Input

Input file contains one integer number z ($1 \le z \le 10^{100}$).

Output

Output the minimal number of proposals the Ministry must prepare.

Examples

discuss.in	discuss.out
6	4
7	7

Problem 5. Yet Another Game with Words

Input file:	`game.in`
Output file:	`game.out`
Time limit:	3 seconds
Memory limit:	256 megabytes

A funny game with words is gaining popularity among computer geeks. The game is played by two geeks. The geeks have a common word which is initially empty. The players make moves in turn. A move consists of adding a letter to the current word, either to its end, or to the beginning.

If after the player's move the word is the correct word of the geek language, or after the player's move the word is not a subword of any correct word of the geek language, the player who made such move loses the game.

For example, the game can proceed as follows:

First player: "m"

Second player: "mp"

First player: "0mp"

Second player: "c0mp"

"c0mp" is the correct word of the geek language, so the second player loses.

Given the set of correct words of the geek language, find out who wins if both players play optimally, and if it is the first player, which letter she should say first.

Input

The first line of the input file contains n — the number of correct words of the geek language ($1 \le n \le 200$). The following n lines contain one word each, word consists of characters with ASCII codes ranging from 33 to 126 and have length not exceeding 50. All words are case sensitive.

Output

Output "`First`" if the first player wins the game, or "`Second`" if the second player does. If the first player wins the game, the second line must contain all possible characters that the first player can start the game with in order to win it, ordered by ASCII codes.

Example

game.in	game.out
10 one two three four five six seven eight nine ten	First fu

Problem 6. Merge

Input file:	`merge.in`
Output file:	`merge.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Suppose that you have two non-decreasing sequences of integer numbers $x_1 \le x_2 \le \ldots \le x_m$ and $y_1 \le y_2 \le \ldots \le y_n$. We can merge these sequences into one non-decreasing sequence $z_1 \le z_2 \le \ldots \le z_{m+n}$. If we do so, there is a way to express z_i in terms of x-s and y-s. For example, clearly $z_1 = \min(x_1, y_1)$ and $z_{m+n} = \max(x_m, y_n)$.

For i other than 1 and $m + n$ formulas can be more complicated. For example, $z_2 = \min(\max(x_1, y_1), \min(x_2, y_2))$. It turns out that it is possible to express z_i using min and max for all i.

Given m n, you have to find such expression.

Input

The first line of the input file contains m and n ($1 \le m, n \le 20$).

Output

Output the expression of z_i for all i from 1 to $n+m$, one on a line. Use brackets and binary infix notation with '`&`' for min and '`|`' for max. The length of each expression must not exceed 2 000.

Example

merge.in	merge.out
2 2	x1&y1 (x1|y1)&(x2&y2) (x1|y1)|(x2&y2) x2|y2

Problem 7. The Most Relevant Pattern

Input file:	`pattern.in`
Output file:	`pattern.out`
Time limit:	4 seconds
Memory limit:	256 megabytes

Let us call two strings s and t over the same alphabet Σ *isomorphic* if they have the same length n and there exists a bijection $\varphi : \Sigma \to \Sigma$ such that $s = \varphi(t_1)\varphi(t_2)...\varphi(t_n)$.

Consider a string s, a string p is said to have a isomorphic occurrences in s if there are a positions i of s such that $s[i..i + |p| - 1]$ is isomorphic to p. The *relevance* of p that has a isomorphic occurrences in s is the value $a \cdot |p|$. Given s you have to find the string p that is most relevant to s.

Input

Input file contains the string s. It contains only lowercase letters ('`a`'–'`z`'). Its length doesn't exceed 4000.

Output

Output one string containing only lowercase letters — the most relevant string to s. If there are several solutions, output any one.

Example

pattern.in	pattern.out
abacabadabacaba	evere

Problem 8. Straight or Flush

Input file:	`poker.in`
Output file:	`poker.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

In classic poker players are dealt five cards each. After some negotiations and card exchanges the players are comparing their hands, whoever has the best hand wins. Unlike classic poker, in Texas Holdem the players are dealt two cards each and five other cards are dealt for all players (community cards). Peter started playing Texas Holdem recently and noticed that players are more often winning with flush than with straight, although the flush is considered to be more valuable.

For those unfamiliar with poker rules, we give some information needed for this problem. Each card has a *rank* (ranging from 2 to Ace, 13 possible values) and a *suit* (one of four variants: spades, clubs, diamonds or hearts). A *hand* consists of five cards. *Straight* is five cards with consecutive ranks, for example, six diamonds, seven spades, eight spades, nine hearts and ten spades form a straight (for those *familiar* with poker — we will ignore A2345 straight for the purpose of this problem). *Flush* is five cards of the same suit, for example, two, three, nine, queen and ace of hearts form a flush.

It is a simple task of combinatorics to find out that there are 9216 possible straights and 5148 possible flushes, so straight is more probable (the probability of getting a straight is about 0.355%, the probability of getting a flush is about 0.198%).

However, things change in case of Texas Holdem. In Texas Holdem you can create your hand choosing five out of seven cards, so the probabilities of getting a flush and a straight change. The probability of getting a straight is now about 4.353%, and the probability of getting a flush is 3.057%. Straight is still more probable, but this time the ratio of probabilities is different (straight is "less more probable"). And since flush is easier to follow and trying to create a flush is more popular strategy, winning with flush becomes more frequent event than winning with straight.

In this problem you have to find the probabilities of getting straight and flush in generalized version of Texas Holdem. Let there be r possible card ranks (1 through r) and s possible suits, so the deck contains rs cards. Let d cards be dealt (together private and community) and the hand be formed by any c of them. The hand is a straight if the cards in the hand have consecutive ranks, the hand is a flush if the cards in the hand have the same suit. Find the probability that one can create a straight and the probability that one can create a flush out of d randomly dealt cards, and output them as irreducible fractions.

Input

The input file contains several test cases. Each test case consists of r, s, d and c on a line by itself $1 \le r \le 20$, $1 \le s \le 20$, $1 \le d \le 20$, $1 \le c \le d$, $d \le rs$).

The line containing four zeroes terminates input, this line must not be processed.

Output

For each test case output two irreducible fractions: the probability of getting a straight and the probability of getting a flush. Separate output for different test cases by a blank line.

Example

poker.in	poker.out
13 4 5 5 13 4 7 5 3 3 3 2 20 20 20 1 0 0 0 0	192/54145 33/16660 800/18377 78639/2572780 3/4 19/28 1/1 1/1

Let us give some comments on the third testcase. The total number of possible combinations of 3 cards is $\binom{9}{3} = 84$. The number of combinations when they all have different suits is $3^3 = 27$. So the probability of getting a flush with two of them is $(84-27)/84 = 19/28$. In order to not be able to create a straight there must be either three cards with rank 2 or no cards with rank 2. So there are $1 + \binom{6}{3} = 21$ such variants and the probability of having a straight is $(84-21)/84 = 3/4$.

In the last testcase one card is always both flush and straight, so both probabilities are $1/1$.

Problem 9. No Term Repetition Exactly One Satisfiability

Input file:	sat.in
Output file:	sat.out
Time limit:	2 seconds
Memory limit:	256 megabytes

Most problems related to satisfiability of boolean formulae turn out to be NP-complete. There are some notable exceptions, one quite surprising is the recent work by Valiant who shows that counting the number of satisfying assignments to monotone planar 3-CNF formulae where each variable occurs twice modulo 7 can be done in polynomial time. Well, probably we will give you this problem this day or that, but not today. Today your problem will be another SAT special case solvable in polynomial time — no term repetition exactly one satisfiablity.

A boolean formula in CNF is formula of a kind

$$\Psi(x_1, \ldots x_n) = (t_{11} \vee t_{12} \vee \ldots \vee t_{1s_1}) \wedge \ldots \wedge (t_{k1} \vee t_{k2} \vee \ldots \vee t_{ks_k})$$

where each term t_{ij} is either some variable x_l or its negation $\overline{x_l}$. A fomula in CNF is said to have no term repetitions if no term occurs more than once in it. For example, formula $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3 \vee x_4)$ has no term repetitions, but formula $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee \overline{x_3} \vee x_4)$ has one: the term $\overline{x_3}$ repeats twice.

The problem of exactly one satisfiability is to assign such values to variables that exactly one term in each clause is satisfied. For example, the formula $(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_3 \vee x_4)$ can be exactly-one-satisfied by setting $x_1 = 1$, $x_2 = 0$, $x_3 = 1$, $x_4 = 0$.

Given no term repetition formula in CNF you have to decide whether it has an exactly-one-satisfying assignment and if it does, find this assignment.

Input

The first line of the input file contains two integer numbers: n — the number of variables that appear in the formula, and k — the number of clauses ($1 \le n \le 300$, $1 \le k \le 300$). The following k lines describe clauses. Each clause is described with the number of terms in it s_i followed by term descriptions. Each term is described by an integer number, a number $p > 0$ means x_p, a number $q < 0$ means $\overline{x_{-q}}$.

Output

If the formula is exactly-one-satisfiable, output "YES" at the first line of the output file. The second line must contain n numbers each of which is either 0 or 1 — values of $x_1, x_2, \ldots, x_n$.

If the formula is not exactly-one-satisfiable, output "NO" at the first line of the output file.

Examples

sat.in	sat.out
4 2 3 1 2 -3 3 -1 3 4	YES 1 0 1 0
5 4 3 1 2 3 3 -1 4 5 2 -2 -4 2 -3 -5	NO

Problem 10. Trading Game

Input file:	`trading.in`
Output file:	`trading.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are playing a trading game over internet. There are n types of resources in the game and m possible buildings that can be bought for resources.

The players make moves in turn. During your move you can trade with other players. When trading you can exchange some resource to other one if the other player agrees. Trade is always one for one.

It's your turn now and you want to build some building. However, you don't have enough resources to build it. You would like to trade with other players. However, each player also wants to build some building. For each player i there is a building v_i she wants to build. The player agrees to make exchange only if the number of resources needed for her building that she is missing decreases after the trade. For example, let there be five types of resources: sheep, brick, wood, ore and wheat. You would like to build a settlement that costs one sheep, one brick, one wood and one wheat. However you only have two sheep and two wood. Let one of your opponents have three bricks and two wheat, and let her want to build road which requires one brick and one wood. Then you can exchange a wood for a wheat or a wood for a brick with her, because now she is missing one resource (wood) for her building, but after exchange she would be missing zero resources. However, she wouldn't agree to exchange sheep for anything because she doesn't need it. You cannot perform both trades above because after the first exchange the player no longer needs wood.

Given a set of resources you and other players have, and the building everyone would like to build, find out whether it is possible to organize trade so that you could build the building you want.

Input

The first line of the input file contains p — the number of players, n — the number of resources, m — the number of buildings ($1 \le p, n, m \le 20$). The second line contains n words — the names of resources.

The following m lines contain descriptions of buildings. Each description starts with the name of the building followed by "`requires`" followed by resources needed for the building in format "`<number> <name>`". Each building requires at least 1 and at most 200 units of resources.

The following p lines contain descriptions of players, each descriptions starts with the name of the player, followed by "`wants`" followed by the name of the building she wants, followed by "`has`" followed by resources that the player has in format "`<number> <name>`". Each player has at most 200 units of resources. If the player has nothing, the "`has ...`" part is omitted. You are player number 1.

All names consist of lowercase and uppercase letters and are case sensitive. Items are separated by commas and/or spaces.

Output

If it is impossible to build the building, output "`No way`". In the other case output a sequence of trade operations, one on a line, followed by building operation. See example for further clarification.

Examples

trading.in

```
3 5 3
sheep brick wood ore wheat
settlement requires 1 sheep, 1 brick, 1 wood, 1 wheat
road requires 1 brick, 1 wood
city requires 3 ore, 2 wheat
Andrew wants settlement, has 2 sheep, 2 wood
Ann wants road, has 3 brick, 2 wheat
Jane wants settlement, has 3 brick, 1 wheat, 3 wood
```

trading.out

```
trade with Ann wood for wheat
trade with Jane sheep for brick
build settlement
```

trading.in

```
3 5 3
sheep brick wood ore wheat
settlement requires 1 sheep, 1 brick, 1 wood, 1 wheat
road requires 1 brick, 1 wood
city requires 3 ore, 2 wheat
Andrew wants settlement, has 2 sheep, 2 wood
Ann wants road, has 3 brick, 2 wheat
Jane wants settlement, has 1 sheep, 2 wheat, 3 wood
```

trading.out

```
No way
```

trading.in

```
5 5 3
sheep brick wood ore wheat
settlement requires 1 sheep, 1 brick, 1 wood, 1 wheat
road requires 1 brick, 1 wood
city requires 3 ore, 2 wheat
Andrew wants settlement, has 1 sheep, 3 wood
Ann wants road, has 3 brick
Alice wants road, has 3 brick
Jane wants settlement, has 1 sheep, 2 wheat, 1 wood
Joan wants road
```

trading.out

```
trade with Ann wood for brick
trade with Alice wood for brick
trade with Jane brick for wheat
build settlement
```

Problem 11. Parse Tree

Input file:	`tree.in`
Output file:	`tree.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Arithmetic expression with addition, subtraction, multiplication, division and power raising are defined using the following BNF:

$$\begin{aligned}
\langle \text{expression} \rangle &\longrightarrow \langle \text{term} \rangle \mid \langle \text{expression} \rangle + \langle \text{term} \rangle \mid \langle \text{expression} \rangle - \langle \text{term} \rangle \\
\langle \text{term} \rangle &\longrightarrow \langle \text{factor} \rangle \mid \langle \text{term} \rangle * \langle \text{factor} \rangle \mid \langle \text{term} \rangle \,/\, \langle \text{factor} \rangle \\
\langle \text{factor} \rangle &\longrightarrow \langle \text{item} \rangle \mid \langle \text{item} \rangle \,\hat{}\, \langle \text{factor} \rangle \\
\langle \text{item} \rangle &\longrightarrow \langle \text{variable} \rangle \mid (\langle \text{expression} \rangle) \\
\langle \text{variable} \rangle &\longrightarrow a \mid b \mid \ldots \mid z
\end{aligned}$$

Please note that addition, subtraction, multiplication and division are left-associative, and power raising is right-associative.

Given an expression, the parsing tree for it is defined in the following way: it is a binary tree with internal nodes corresponding to operations and leaves corresponding to variables. The tree is built recursively.

- The tree for a variable is just a one-node tree with this variable.
- The tree for an item that is a variable is the tree for this variable. The tree of an item that is an expression in parentheses is the tree for this expression.
- The tree for a factor that is an item is the tree for this item. The tree for a factor that is an item raised to the power of some factor is the tree with a root containing '^' operation, its left subtree is the tree for the corresponding item, its right subtree is the tree for the corresponding factor.
- Trees for term and expression are defined similarly, except that their operations are left-associative.

Given an arithmetic expression, build the parse tree for it and print it to the output file.

Input

The input file contains an arithmetic expression. Its length doesn't exceed 400 characters.

Output

Output the tree. The tree for a variable must have a size of 1×1 and contain the variable. The tree with the operation in the root with subtrees T_1 and T_2 of sizes $h_1 \times w_1$ and $h_2 \times w_2$, respectively, has size $(\max\{h_1, h_2\} + 2) \times (w_1 + w_2 + 5)$.

See sample output for exact formatting. The auxiliary characters used are minus '-' (ASCII 45), dot '.' (ASCII 46), vertical line '|' (ASCII 124), brackets '[' and ']' (ASCII 91 and 93).

Example

tree.in	tree.out

`(a+b+c)*(d-a)`	<pre>.----[*]----.
	| |
	.----[+]-. .-[-]-.
	| | | |
	.-[+]-. c d a
	| |
	a b</pre>