Problem A. Anniversary

Input file:	`a.in`
Output file:	`a.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

In the year 2134, when the Earth population reached 10^{18}, the people decided to celebrate this event. N diminutive bulbs (the linear size of each bulb was less than 0.1 millimeters) were placed on a huge area specially allotted for the celebration. The bulbs were numbered with consecutive integers from 1 to N in some order.

In the beginning, all the bulbs were off. Afterwards, exactly 10^{18} steps were performed — one per each citizen of Earth. At the i-th step, the states of all bulbs with number X such that i divides X toggled at the same time. If a bulb is on, toggling its state would switch it off, and vice versa, toggling the state of an off bulb would switch it on. The interval between the consecutive steps was only 1 picosecond, so the whole celebration took around a week and a half.

Actually, all this stuff looked as pointless flickering, but the spectators were delighted by the colossal scale of the wonderful event. Finally, it was over. There was nothing more to look at, and — if you think soberly — nothing worthy ever happened.

In the meanwhile, after step 10^{18} some of the bulbs are still on. While the people are recovering from the shock, pondering why did they need such a celebration and who will cover its costs, we suggest that you count the number of bulbs which are still on and consume the precious power.

Input

The only line of input contains the integer N ($1 \le N \le 2^{63} - 1$).

Output

Print a single integer — the number of bulbs which are still on after step 10^{18}.

Example

a.in	a.out
1	1
9	3
100	10

Problem B. Trade

Input file:	`b.in`
Output file:	`b.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

Trading is a subtle thing. A successful huckster not only should reasonably choose the time to sell the goods and master the art of touting, but also thoroughly examine the market. It's important to know which of the merchants trade with each other and which don't. Sometimes the merchants don't directly trade, but their goods still find a way to each other via other merchants. For example, if merchants A and B trade directly and merchants B and C trade directly, then goods from A and C will get to each other via merchant B. In general, the goods may get from a merchant to another via any number of intermediate merchants. Another important concept are the inseparable pairs — these are the pairs of merchants which trade directly, and no sequence of intermediate merchants exists through which the merchants in the pair could trade indirectly.

Manao wants to become a successful huckster. We don't know what necessary skills he does possess, but he surely lacks the knowledge about market situation. What he currently knows is that there are N merchants at the market, M pairs trade directly and K of them are inseparable. What he needs is the information of sort "A trades with B, C trades with D, X trades with A". In most cases, this is ambiguous, but at the moment any fitting scheme would suffice.

You're given T scenarios with the values of N, M and K. For each of them determine whether a corresponding trade market exists and if yes, output its description. Merchants are numbered in some order from 1 to N, the description of the trade market is the set of all pairs of merchants who trade directly.

Input

The first line contains the number of scenarios T. Each of the following T lines contains three numbers N, M, K ($2 \le N \le 100$ $0 \le K \le M \le N \cdot (N-1)/2$). The number of scenarios in a single input does not exceed 100. Sum of M's over all scenarios in a single input does not exceed $5 \cdot 10^4$.

Output

For each of the T scenarios output "`NO SOLUTION`" (without quotes) if the corresponding trade market does not exist. Otherwise, output "`TRADE MARKET FOUND`" followed by M lines. Each of the lines must contain a pair of numbers separated by a space — the numbers of the merchants trading directly.

Example

b.in	b.out
3 4 3 0 4 2 0 5 5 2	TRADE MARKET FOUND 1 2 2 3 3 1 NO SOLUTION TRADE MARKET FOUND 1 2 2 3 3 1 1 4 1 5

Problem C. Grid

Input file:	`c.in`
Output file:	`c.out`
Time limit:	2 seconds
Memory limit:	64 Mebibytes

There is an infinite grid on the plane consisting of vertical and horizontal lines. The lines are given by equations $X = i \cdot K$ and $Y = j \cdot K$ for any integers i and j. Also, a set of N points is given. We say that a point lies on the grid if it lies on at least one of the lines which form the grid.

The grid may be moved parallel to the coordinate axes. Translation of the grid by vector (dx, dy) means that each line $X = i \cdot K$ is replaced by line $X = i \cdot K + dy$ and each line $Y = j \cdot K$ is replaced by line $Y = j \cdot K + dx$. Find the maximum possible number of points from the given set which can simultaneously lie on the grid after its translation by some vector.

Input

The first line of input contains two integers N and K ($1 \le N \le 10^5$, $2 \le K \le 10^9$). The i-th of the next N lines contains numbers X_i, Y_i — the coordinates of the i-th point. The points' coordinates are integers and do not exceed 10^9 by absolute value. No two points coincide.

Output

Output the maximal possible number of points from the given set which can simultaneously lie on the grid.

Example

c.in	c.out
6 2 2 0 0 1 2 2 3 3 3 2 4 1	5

Note

One of the optimal translation vectors in the given example is $(1, 0)$.

Problem D. Drawing

Input file:	`d.in`
Output file:	`d.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

Little Vasya is very fond of drawing.

Today he decided to paint the cells of a board consisting of N rows and M columns in different colors. To make the outcome less predictable, he will paint it according to the following rules:

- For each i from 1 to N, with probability $R_i\%$ he colors all the cells in row i red.
- For each j from 1 to M, with probability $C_j\%$ Vasya colors all the cells in column j blue.

If a cell is painted both red and blue then due to color mixing it becomes green.

Initially, neither cell on the board was colored at all. After the described drawing procedure ended, exactly G cells turned out green. Determine the expected number of cells which Vasya painted in some color (red, blue or green).

Input

The first line of input contains three integers N, M and G, separated by single spaces ($1 \le N, M \le 100$, $0 \le G \le N \cdot M$).

The second line contains integers $R_1, R_2, \ldots, R_N$, separated by single spaces, $0 < R_i < 100$. The third line contains integers $C_1, C_2, \ldots, C_M$, separated by single spaces, $0 < C_i < 100$. There exists at least one way to paint some rows red and some columns blue and obtain exactly G green cells in the end.

Output

Output the sought expected value. The absolute error of the output should not exceed 10^{-5}.

Example

d.in	d.out
2 2 1 50 50 50 50	3.0
2 2 0 50 50 50 50	2.2857142857142856
2 3 2 3 2 3 2 3	5.313107297058556

Problem E. Empire

Input file:	`e.in`
Output file:	`e.out`
Time limit:	3 seconds
Memory limit:	64 Mebibytes

The Empire spans across N planets. The planets are numbered from 1 to N. Planet 1 is the capital of the Empire, where the Emperor resides and the troops are trained. Uprisings are a common thing at the planets, and they must be suppressed immediately using the armed forces.

There are one-way teleports installed at some planets to ensure the troops' fast movement. There are a total of M teleports in the Empire. Using the i-th of them, one can instantly get from planet A_i to planet B_i (but not vice-versa). Therefore, an uprising on planet X can be suppressed in time if there is a sequence of planets $P_1, \ldots, P_k$ ($k \geq 2$) such that $P_1 = 1$, $P_k = X$ and for each $1 \leq i < k$ a teleport from planet P_i to planet P_{i+1} exists. Since the troops stay on the planet after suppressing an uprising to maintain the order, their comeback into the capital is not a matter of concern.

Verify whether an uprising at any planet of the Empire can be suppressed with the existing teleport system. If positive, output 0. Else determine the least number of teleports which must be added to ensure that an uprising at any planet can be suppressed. Each teleport may connect any pair of planets.

Input

The first line contains two integers N and M ($2 \leq N \leq 10^5,\ 0 \leq M \leq 2 \cdot 10^5$) . The i-th of the following M lines contains a pair of numbers A_i, B_i ($1 \leq A_i, B_i \leq N$ for each $1 \leq i \leq M$). No teleport connects a planet to itself. No two teleports have the same startpoint and endpoint.

Output

Print the least number of additional teleports which guarantees that any uprising can be suppressed.

Example

e.in	e.out
6 4 3 1 4 6 1 2 4 5	2

Note

One optimal way is to build teleports from planet 2 to planet 4 and from planet 5 to planet 3.

Problem F. Reform (Division 1 Only!)

Input file:	`f.in`
Output file:	`f.out`
Time limit:	3 seconds
Memory limit:	64 Mebibytes

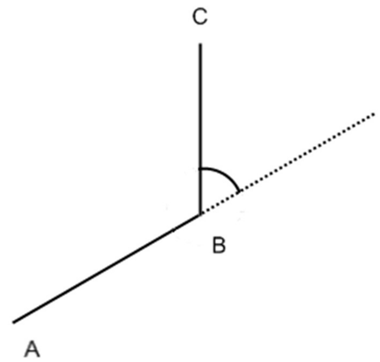
On one remarkable Cartesian plane there was an extremely conservative country. There were N cities in the country numbered from 1 to N. Compared to the infinite plane, the cities are so small that we shall consider them as points on the plane. The coordinates of the i-th city are (X_i, Y_i). No two cities reside in the same point. No three cities lie on a straight line.

Some pairs of cities are connected with bidirectional roads. Each road is a line segment connecting two cities. There are exactly three outgoing roads for each city. No road connects a city to itself. No pair of cities is connected with more than one road.

That's how it went in this country from the very old times, and nobody ever thought of changing anything. But the trouble crept quietly — a liberal king got to the throne. And immediately issued a decree concerning the roads system of the country. The decree stated that some of the roads will be removed in order to meet the following conditions:

- Each city must have 2 outgoing roads.
- The turning angle between two roads outgoing from the same city must be strictly less than 60 degrees.
- No two roads intersect anywhere except the cities.

The turning angle between two roads is calculated in the following way. Say there are roads from city B to cities A and C. The turning angle between them is equal to the exterior angle at point B in triangle ABC (see the picture).



Implementation of the reform was delegated to the minister of transportation. He has to decide which of the roads will be removed and which will be kept. To please the king, the minister wants to keep such set of roads (of all possible sets satisfying the decree) which minimizes the maximum turning angle between roads outgoing from the same city.

Input

The first line of input contains an even integer N ($4 \le N \le 200$). Each of the followng N lines contains 5 integers. The first two integers in the i-th of these lines are X_i and Y_i ($-10^5 \le X_i, Y_i \le 10^5$). The following three numbers are the numbers of cities with which city i is connected initially. No two cities lie in the same point. No three cities lie on a straight line. Each city is connected with exactly three other cities. No pair of cities is connected with more than one road. No road connects a city to itself. Any two turning angles (not necessarily at one city) in the country before the reform differ by at least 10^{-5} degrees. Any turning angle in the country before the reform differs from 60 degrees by at least 10^{-5} degrees.

Output

If there is no way to fulfill the decree conditions, print a single line with text "`Minister's life is short :(`" (without outer quotes). Characters "`'`", "`:`" and "`(`" have ASCII-codes 39, 58 и 40, correspondingly.

Otherwise output the solution of the problem which the minister should choose. Print N integers. If the i-th of them is equal to j, it means that the road between cities i and j should be removed.

Example

f.in	f.out
4 0 0 2 3 4 41 0 1 3 4 0 42 4 2 1 43 44 2 3 1	Minister's life is short :(
8 0 100 2 6 8 0 201 3 1 5 102 303 2 4 8 204 305 3 5 7 306 207 2 4 6 308 109 1 5 7 210 0 6 8 4 111 0 1 3 7	6 5 8 7 2 1 4 3

Problem G. Constellation (Division 1 Only!)

Input file:	`g.in`
Output file:	`g.out`
Time limit:	3 seconds
Memory limit:	256 Mebibytes

Recently explorers discovered an ancient scroll which describes some constellation. It suggests that the constellation consists of M stars and contains the distances between each pair of those stars. The distances are measured as if the sky was a plane and the stars were points on it.

There are N stars visible in that hemisphere of the present-day sky where the constellation is supposed to be. Usually, a constellation comprises of the brightest stars of some fragment on the sky, but the brightnesses of the stars could have changed over the past milleniums, so this won't help. Therefore, identifying the stars on present-day sky which were in the constellation should be performed on the basis of the given distances only.

A possible location of the constellation is a list of stars $(I_1, I_2, ldots, I_M)$ such that for each i and j $(1 \le i, j \le M)$ the distance between stars I_i and I_j equals the distance between the i-th and j-th stars of the scroll. Two locations are different if there is at least a single position in the lists where stars differ.

You're given a set of stars on the present-day sky. Also, you're given an $M \times M$ matrix, where element (i, j) is the squared distance between star i and star j of the constellation. Count the number of different possible locations of the given constellation.

Input

The first line of the input contains a single number M. The next M lines contain M numbes each — the distance matrix (the given distance matrix is symmetric, its main diagonal contains zeroes only, and the rest of numbers are positive and do not exceed 10^8).

Next line contains number N. Each of the next N lines contains a couple of numbers X_i, Y_i — the coordinates of the i-th star on present-day sky ($2 \le N \le 3 \cdot 10^4$, $2 \le M \le min(N, 20)$). The coordinates of each star are integers not exceeding 10^4 by absolute value. No two stars are in the same point.

Output

Print the number of possible locations of the constellation from the scroll.

Example

g.in	g.out
3 0 1 2 1 0 1 2 1 0 4 0 0 1 0 0 1 1 1	8

Note

The possible locations of the constellation are (1,2,4), (1,3,4), (2,1,3), (2,4,3), (3,1,2), (3,4,2), (4,2,1), (4,3,1).

Problem H. Presents (Division 1 Only!)

Input file:	`h.in`
Output file:	`h.out`
Time limit:	4 seconds
Memory limit:	64 Mebibytes

Manao decided to arrange a party. He brightly decorated his hut, scrupulously selected the menu, invited his close friends... and suddenly realised that he forgot about the presents!

On his parties Manao always gives presents to his guests. Moreover, he always gives each guest something this guest will be glad to have. The guests themselves also bring gifts, each of them a single one. Since there is virtually no time left, Manao won't manage to buy any presents, so he has to find a way to slip out. And the brilliant idea strikes his mind: distribute the gifts which the guests brought between them! Of course, giving a person the gift he brought is unacceptable. Manao knows his friends quite well and can say in advance, who of them would like whose gift.

Unfortunately, it could be that distributing the gifts in a way to keep everyone satisfied is impossible. That's why Manao decided to additionaly invite some of his acquaintances. Manao knows them well, too, so he can exactly predict what each of them will bring and what will like. Manao doesn't want to hurt his acquaintances, so each of them should also receive a gift he'd be glad to have. Manao doesn't want his party to become a muddle, so he will invite the least number of acquaintances which guarantees that distributing the presents to make everyone happy is possible.

Manao has N friends and M acquaintances. We'll number the friends from 1 to N and acquaintances from $N+1$ to $N+M$. You're given a matrix which defines who would like whose gift. Determine the least number of acquaintances which must be called (all the friends are already invited). If it is impossible, print -1.

Input

The first line of the input contains two numbers N and M ($2 \le N \le 100$, $0 \le M \le 100$). Each of the following $N+M$ lines contains $N+M$ characters "`Y`" or "`N`". The j-th character at i-th line defines whether person i would like person j's gift: "`Y`" stands for "yes" and "`N`" stands for "no". There exists no guest that will like his own gift.

Output

Print -1, if no subset of the acquaintances will save Manao's party, and the least possible cardinality of such a subset otherwise.

Example

h.in	h.out
2 2 NYYY NNYN YNNN NYNN	1
3 2 NNNYN YNYNN NNNYN NYNNY NNYNN	-1
3 2 NYNYN YNYNN NNNYN NYNNY NNYNN	2

Note

In the first example, Manao may invite person 3 and give his gift to person 2, give person 2's gift to person 1 and give person 1's gift to person 3.

In the third example Manao may invite both acquaintances, exchange person 1 and person 2's gifts and distribute the three remaining gifts between remaining guests.

Problem I. ICPC (Division 1 Only!)

Input file:	`i.in`
Output file:	`i.out`
Time limit:	4 seconds
Memory limit:	64 Mebibytes

The gloomy cult of ICPC is existing for quite a long time. The witnesses confess that the servants of the cult practice extremely strange things. They divide into groups of three, worship the mysteriously glowing display, frantically knock on some keys grouped on a weird panel. The communication seems to be bidirectional — sometimes the "deity" returns signs of favor, which cause the joyful shouting of the servants. Although actually the majority of slothful servants rarely achieves this. More often the "deity" is unsatisfied, which causes confusion and even despondency in the ranks of the servants.

The nature of the "deity" is widely rumoured. Some say that the "deity" exists in several guises and each of them is controlled by an initiated person (maybe he's not even human, but a creature of a higher order) — an Admin. One highly acclaimed Admin is the so-called Admin with Big Beard. But let's get back to the problem.

Once, N servants of the ICPC cult gathered and complained that the traditional means of worship are outworn and pestering, and they aren't grim enough. So it was decided to replace the frantic knocking on the keyboard with dark energy interchange. A plan of the interchange was developed — the set of M pairs (A, B), which means that servant A should transfer the dark energy to servant B (and correspondingly servant B should receive the dark energy from servant A).

The servants began acting according to the plan, but the energy wouldn't transfer. As you surely undestand, it can't possibly be that the ICPC servants possess no dark energy, so there had to be something else. In a while, the servants realized that they had to arrange in a special way. They drew a magic circle and stood at its border.

Of course, the existence of the magic circle is not enough. The process of energy transfer has to be associated with the circle. As all of us know, the process of energy transfer is associated with the circle if each servant transfers energy to the servants immediately succeeding him in the circle (in clockwise order) and receives energy from the servants immediately preceding him.

Let us define it more formally. Denote with $next(X)$ the servant immediately succeeding servant X in the circle (in clockwise order) and with $prev(X)$ the servant immediately preceding X. Also, let us define for $i > 1$ the values $next^i(X) = next(next^{i-1}(X))$ and $prev^i(X) = prev(prev^{i-1}(X))$. Put $next^1(X) = next(X)$ and $prev^1(X) = prev(X)$. The process of energy transfer is associated with the magic circle if and only if for each servant the following conditions are met:

- He transfers energy to servants $next^i(X)$, $1 \le i \le A$, where A is the total number of servants who receive energy from X;
- He receives energy from servants $prev^i(X)$, $1 \le i \le B$, where B is the total number of servants who transfer energy to X.

The goal has never been so close, but arranging on the circle in the proper way turned out to be a difficult task. Help the ICPC cult brethren!

Input

The input contains several scenarios. The first line of input contains their number T, afterwards their descriptions follow.

The description of each scenario begins with a line containing integers N and M ($N \ge 3$), which is followed by M lines. Each of them contains two integers A and B ($1 \le A, B \le N,\ A \ne B$), denoting that servant with number A should transfer the energy to servant with number B. Each pair (A, B) will be at most once in a single scenario. Both (A, B) and (B, A) will not be simultaneously in a single scenario. The servants are numbered from 1 to N. Sum of N's over all scenarios in a single input does not exceed 10^5. Sum of M's over all scenarios in a single input does not exceed $2 \cdot 10^5$.

Output

Print a single line for each scenario in the input. If the solution for the scenario does not exist, the line should contain text "`Epic fail`" (without quotes). Otherwise, the line should contain a permutation of numbers from 1 to N — the order in which the servants should stand to achieve the process of energy transfer associated with the magic circle. The order should be clockwise. If there are several solutions, output any.

Example

i.in	i.out
3 6 10 1 6 2 3 2 4 2 5 3 1 3 6 4 3 4 5 5 3 6 2 3 0 4 3 1 2 2 3 3 1	1 6 2 4 5 3 1 2 3 Epic fail

Problem J. Numbers (Division 2 Only!)

Input file:	`j.in`
Output file:	`j.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

A non-negative integer is K-digit, if it can be written using K digits, but $(K-1)$ digits are not enough for this. For example, 43 is a two-digit number, 2010 is four-digit, and 0 and 5 are one-digit numbers.

For given A and B count, how many K-digit non-negative integers exist, where K is not less than A and not greater than B.

Input

The only line contains integers A and B ($1 \le A \le B \le 1000$).

Output

Output the answer to the problem without excessive leading zeros.

Example

j.in	j.out
1 1	10
3 5	99900

Problem K. English (Division 2 Only!)

Input file:	`k.in`
Output file:	`k.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

There are 26 letters in English alphabet, of which 5 ('a', 'e', 'i', 'o', 'u') are vowels, 20 are consonants and one is considered a semivowel, i.e. neither a vowel nor a consonant (letter 'y'). A word is sonorous if the number of vowels in it exceeds the number of consonants. If a word contains some letter more than once, then each its occurence counts — for example, "alabama" is a sonorous word.

For a given word determine the least number of letters in it which must be replaced to obtain a sonorous word.

Input

A single line of input contains the word given. The length of the given word does not exceed 1000 letters. The word consists of lowercase letters.

Output

Output the least number of letters replacement of which makes the given word sonorous.

Example

k.in	k.out
dog	1
alabama	0
y	1

Problem L. Diamond (Division 2 Only!)

Input file:	`l.in`
Output file:	`l.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

You are a big fancier of jewelry and gemstones. Recently, a remarkable diamond was put for sale in the jewelry store next to your house. Unfortunately, its price was also remarkable — X dollars. Regretful, you had to state that this is way too much.

Apparently, other potential buyers also came to the same conclusion, because soon the store started an advertising campaign. "Get a 5% discount on the diamond for each letter of your name!", read the streamer by the frontdoor. The formulation left some ambiguities, so you decided to question the sellers. As it turned out:

- If a letter occurs in the name several times, it's only counted once. For example, "alexandra" contains only 7 letters in terms of the campaign.
- 5% discount is multiplied by the number of distinct letters in the name. I.e., Alexandra would get a 35% discount.
- The discounts exceeding 100% are not provided, but a 100% discount is okay (yep, the sellers questioned your sanity when they heard this inquiry).

Of course, not making use of such a generous offer would be a crime, but maybe you can save more? By chance, you know a person who makes fake passports. His service is paid as follows: A fake passport costs A dollars. It may contain your real name without changes, or the name may be altered using any number of operations of the following three kinds.

- Inserting a letter at any position in the name costs B dollars.
- Erasing a letter at any position in the name costs C dollars. If a name contains a single letter, erasing it is not allowed.
- Replacing a letter at some position with any other letter of the English alphabet costs D dollars.

Given X, A, B, C, D and your real name $Name$, determine the least amount of money which you need to obtain the diamond (count both the money spent on buying the diamond and the money spent on the fake passport, if any). Taking a fake passport is not neccessary — you may just provide your real passport at the store.

Input

The first line of input contains integer X. The second line contains integers A, B, C and D ($1 \le X, A, B, C, D \le 10^6$), separated by single spaces. The third line contains $Name$. It contains from 1 to 15 characters and consists of lowercase English letters only.

Output

Print the least amount of money you need to get the diamond, measured in cents (a dollar is 100 cents).

Example

l.in	l.out
100 1 2 3 4 eldar	3100
100 100 100 100 100 ivan	8000
9876 1 567 890 1 nebuchadnezzar	296780

Problem M. Puzzle (Division 2 Only!)

Input file:	`m.in`
Output file:	`m.out`
Time limit:	1 second
Memory limit:	64 Mebibytes

You're given N^2 decimal digits from 1 to 9.

Consider some arrangement of these digits in the cells of a square board $N \times N$, one digit per cell. In each of the rows on the board, reading from left to right, we get a decimal representation of some N-digit number. In each of the columns, reading from top to bottom, we also get a decimal representation of an N-digit number. Let S be the sum of all N numbers in rows and all N numbers in columns.

Arrange the given digits on the board to make S maximally possible.

Input

The first line of input contains integer N ($1 \le N \le 8$). In the second line, there are N^2 decimal digits from 1 to 9 written with no delimiters.

Output

Print the maximal possible value of the sum S.

Example

m.in	m.out
2 9174	303