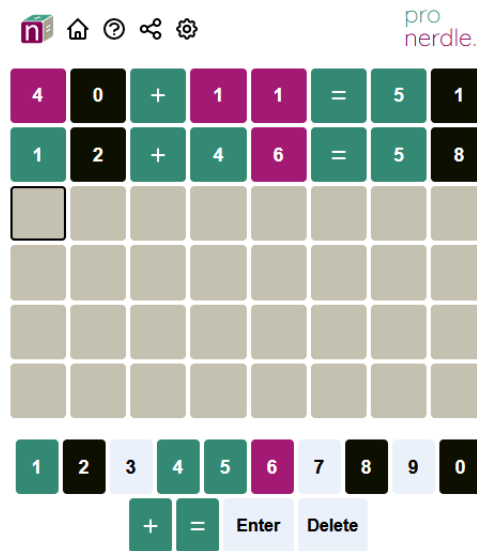


Problem A. Another A+B Problem

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 1024 megabytes

Just before the 2022 Shanghai Collegiate Programming Contest, you decide to play some games to chill out. Today, your friend Tommy introduces a game called Nerdle to you. This game is about to guess some equations.



A game in Nerdle.

You can present an equation to the jury each time. The jury will send back results by the colors:

- If it's green, as the +, = and 5 in the first line of the figure, the digit or operator are in the solution and in the correct spot.
- If it's purple, as the 4 and the first two 1s in the first line of the figure, the digit is in the solution but in the wrong spot.
- If it's black, the digit is not in the solution in any spot, or
- If your guess includes a digit that appears more times than it is in the answer, you will get color tiles exactly as the number of appearances in the answer, and the remaining tiles are black. Therefore, the first equation in the figure contains exactly two 1s, but none of them are in the two middle spots and the last spot. Purple and black tiles can be chosen arbitrarily if both exist for a digit.

After playing for a while, you found that you spent a lot of time just trying to make the left-hand side equal to the right-hand side. As a strong competitive programmer, it's so much a waste of time for you. You decide to use your computer to analyze the result of one equation so that the remaining possible equation can be obtained. The equation and answer have the following restrictions:

- The third and the sixth character are fixed with '+' and '=' respectively, and they are all green.
- The only allowed operator is the binary plus sign. Here, 'binary' is for involving two operands. So, the answer is always in the form "??+??=??" where '?' can be replaced by a digit (0...9).
- Leading zeros are allowed here, so "01+02=03" is a valid equation.

Write a program to find out all possible remaining equations by a given result.

Input

The input contains two strings E, C of length 8 in two lines, denoting the equation asked and the color returned by the jury, respectively. C consists of 'GPB', denoting green, purple, and black of the corresponding spot.

It's guaranteed that:

- E is a valid equation.
- The third and the sixth character of E are fixed with '+' and '=' respectively, and the other positions are digits (0...9).
- The third and the sixth character of C are 'G'.

Output

Print an integer R in the first line, denoting the number of different remaining possible equations. Two equations are considered different if they are different on at least one character.

For the next R lines, print a string in each line denoting a valid equation corresponding to the input.

You can print your solution in any order. It's guaranteed that at least one valid equation exists.

Examples

standard input	standard output
40+11=51 PBGPPGGB	7 11+42=53 11+43=54 11+44=55 11+45=56 11+46=57 11+47=58 11+48=59
12+46=58 GBGGPGGB	6 11+45=56 13+43=56 15+41=56 16+40=56 16+41=57 16+43=59
11+22=33 PBGPPGGP	1 22+13=35
11+22=33 BPGPPGGP	1 22+13=35
01+02=03 PPGPPGPP	2 10+20=30 20+10=30

Note

From the first and the second example, one can conclude that the only solution for the game in the figure is 11+45=56.

The third and the fourth sample are the same, yet the black and purple tiles are chosen differently.

Problem B. Bracket Query

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Bracket sequences are perfect examples of graceful balancing. Here we talk about bracket sequence consisting of ‘(’ and ‘)’. For example, “(())”, “()()” are valid balanced bracket sequence.

More formally, a bracket sequence is valid if and only if it can be constructed by the following rules:

- “()” is a valid bracket sequence.
- If A is a valid bracket sequence, then (A) , i.e., adding a left bracket before and a right bracket after, forms a valid bracket sequence.
- If A, B are valid bracket sequences, then AB , i.e., concatenating them, forms a valid bracket sequence.

Today Sayu is playing a game with you. She has a valid bracket sequence S with length n , and you can ask her questions in the form “What is the difference between the numbers of left and right brackets in the substring of S from the l -th character to the r -th character?”, and then Sayu will tell you the answer.

You want to determine the bracket sequence using the above method. After playing for a while, however, Sayu escaped and went to grab some sleep. “Mission accomplished! Can I go back and sleep now?”

You want to retrieve a possible valid bracket sequence from already known queries. You may also find contradictions in the queries since Sayu was sleepy, in which case you should report no solution.

Input

The first line contains two integers n, q ($2 \leq n \leq 3000, 0 \leq q \leq \min(\binom{n+1}{2}, 5 \times 10^5)$), denoting the length of the bracket sequence and the number of known queries.

In the following q lines, the i -th line contains three integers l_i, r_i, c_i ($1 \leq l_i \leq r_i \leq n, -n \leq c_i \leq n$), denoting the i -th query ranges $q_i = [l_i, r_i]$, and the answer to the query is c_i , which is the number of the left brackets subtracts the number of right brackets in the interval.

It’s guaranteed that n is even, and the queried intervals are pairwise different.

Output

If there’s no valid solution, print “?”.

Otherwise, print “! S ” in a single line where S is a valid bracket sequence consistent with input.

Examples

standard input	standard output
4 1 1 2 0	! () ()
4 1 1 2 2	! (())
2 2 1 1 1 2 2 -1	! ()
2 1 1 1 2	?
4 0	! (())

Note

For the first and the second test case, the only possible valid bracket sequences with length 4 are “(())” and “()()”. By asking the substring containing the first two characters, one can distinguish two different substrings since the answer will be 2 and 0, respectively.

Problem C. Coffee Overdose

Input file: `standard input`
Output file: `standard output`
Time limit: 3 seconds
Memory limit: 1024 megabytes

You're doing your giant project, which will due by the day after tomorrow. You still have some time, yet your body and mind are falling asleep and keeping your head knocking on the desktop.

Don't worry! You have a sufficient supply of coffee – everybody knows that coffee contains caffeine, and caffeine will keep you awake. One issue you might notice is that coffee will keep you awake for a period, but after that, you will be more exhausted. To be more productive, you must use your fatigue weapon wisely.

We use the *stamina* S to evaluate how energetic you are. At each second, you contribute S points of *completeness* to your project, and your stamina will decrease by 1 after that. When your stamina decreases to 0 or less, you will lose control and fall into dreamland.

You can drink a shot of coffee at the beginning of each second, and the effect will last for C seconds. During the effect of the coffee, you can not drink another shot and your stamina will be fixed at the beginning. After that, your stamina will reduce $C + 1$, i.e., 1 extra cost caused by coffee.

Your goal is to maximize the productivity before you are too sleepy. Given S and C , find the optimal schedule to maximize the total points of completeness.

Input

The first line contains one integer T ($1 \leq T \leq 10^5$), the number of test cases.

For each case, there is only one line containing two integers, S, C ($1 \leq S, C \leq 172\,800 = 2 \times 24 \times 60 \times 60$), denoting your initial stamina and the coffee period length.

Output

For each test case, output a single integer in a separate line denoting the maximum possible total points of completeness.

Example

standard input	standard output
4	2
1 2	3
2 1	63
10 4	29859840000
172800 172800	

Note

For the first sample case, you may drink the coffee at the very beginning, and the stamina sequence will be 1, 1, -2.

For the second sample case, since the coffee is literally not working for you, you should stop overdosing and keep a healthier life.

For the third sample case, one optimal schedule is to drink a shot at the beginning of 3-rd and 7-th second respectively, and the stamina sequence will be 10, 9, 8, 8, 8, 8, 3, 3, 3, 3, -2.

Problem D. Demonstrational sequences

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 1024 megabytes

Little Desprado2 learned how to calculate the Greatest Common Divisor (GCD) of two positive numbers a few days ago. One of the most famous algorithm in the history is the Euclidean Algorithm, which takes two positive integers x, y as input, calls itself recursively, and finally returns the GCD of them:

- If $y \neq 0$, return $\text{gcd}(y, x \bmod y)$;
- Otherwise, return x .

Little Desprado2 doesn't think it is interesting enough since everybody knows it. Now, he wants demonstrate some properties of the GCD by generating some infinite sequences. He has two positive integers P and Q , and $Q|P$ is satisfied. Here $a|b$ means that a is a factor of b , that is, $b \bmod a = 0$. And he has k candidate pairs of **positive** numbers $\{(a_1, b_1), (a_2, b_2), \dots, (a_k, b_k)\}$ to generate k infinite sequence, where the i -th sequence $\{x_{i,0}, x_{i,1}, x_{i,2}, \dots\}$ is generated by the following rules:

- $x_{i,0} = a_i$
- $x_{i,j} = x_{i,j-1}^2 + b_i \ (j > 0)$

He thinks that an infinite sequence $\{x_0, x_1, x_2, \dots\}$ is *demonstrational*, if and only if there exists two integers u and v ($0 \leq v < u$) such that $\text{gcd}(x_u - x_v, P) = Q$. Here $\text{gcd}(a, b)$ denotes the greatest common divisor of a and b .

For each infinite sequences, Little Desprado2 wants you to tell him whether it is *demonstrational*.

Input

The first line contains three integers P, Q, k ($1 \leq P \leq 2^{32} - 1$, $1 \leq Q \leq 2^{20}$, $1 \leq k \leq 200$). It's guaranteed that $Q|P$ is satisfied.

Then follows k lines. The i -th line contains two integers a_i and b_i ($1 \leq a_i, b_i \leq 2^{64} - 1$), denoting the i -th pair of numbers.

Output

Print a 0/1 string of length k . The i -th character is 1 if the i -th infinite sequence is *demonstrational*, 0 otherwise.

Examples

standard input	standard output
15 5 5 1 1 1 2 2 4 4 8 8 16	11010
998244352 1048576 3 2022 924 12345678 1234567 23333333 6666666	001

Note

In the first example,

- The first infinite sequence $\{x_{1,0}, x_{1,1}, x_{1,2}, \dots\}$ is $\{1, 2, 5, 26, \dots\}$, so there is $u = 3, v = 0$ satisfied $\gcd(x_u - x_v, P) = \gcd(26 - 1, 15) = 5 = Q$. Therefore, the first sequence is *demonstrational*.
- The second infinite sequence is *demonstrational*, and one of the solutions is $u = 2, v = 0$.
- The fourth infinite sequence is also *demonstrational*, and one of the solutions is $u = 2, v = 1$.
- It can be proved that the third and the fifth infinite sequence is not *demonstrational*.

Problem E. Expenditure Reduction

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 1024 megabytes

In order to make your money more abundant, rather than earning more it's much easier to reduce your expenditure.

Junpei is the manager of a membership restaurant. Due to the influences of the pandemic, the restaurant can not afford the excessively large menu of tasty dishes. It's then a problem to consider how to diminish the menu while keeping the specialties.

The menu can be viewed as a string S containing only lowercase English letters and digits, and Junpei believes that the core feature of the restaurant is a string F that is currently a *subsequence* of S . To reduce the menu, you can reduce S to one of its *substring* S' , while keeping F be the *subsequence* of S' . Junpei asks you to find the shortest *substring* S' from S that satisfies the requirement.

To those who may be curious about the definition of *subsequence* and *substring*, consider two non-empty strings A, B :

- If we say A is a *subsequence* of B , we can find a set of $|A|$ indices $\{i_k\}$ where $1 \leq i_1 < i_2 < \dots < i_{|A|} \leq |B|$, such that $A = B_{i_1}B_{i_2} \dots B_{i_{|A|}}$.
- If we say A is a *substring* of B , we can erase a (possibly empty) prefix and a (possibly empty) suffix from B to obtain A .

Input

The first line contains a single integer T ($1 \leq T \leq 10^4$), denoting the number of test cases.

For each test case, there's a single line containing two string S, F ($1 \leq |S| \leq 10^5, 1 \leq |F| \leq 100$) separated by a single space. It's guaranteed that F is a subsequence of S , and both strings containing only lowercase English letters ('a' to 'z') and digits ('0' to '9').

It's guaranteed that the $\sum |S|$ of T cases doesn't exceed 5×10^5 .

Output

For each test case, print one string in a single line denoting the shortest substring of S containing F .

If there are multiple answers, print any of them.

Example

standard input	standard output
4	145
114514 15	aic
shanghaicpc ac	abbbc
aaabbbbaabbbccc abc	owdeliciousandfreshiti
howdeliciousandfreshitis oishii	

Problem F. Forest of Magic

Input file: **standard input**
Output file: **standard output**
Time limit: 8 seconds
Memory limit: 1024 megabytes

Forest of Magic, as its name suggests, is a forest full of magic. In the Forest lives Kirisame Marisa, who is a young but skilled magician.

There are many places in the Forest, and they are connected with bidirectional roads, in such way that each pair of places are connected directly or indirectly by the roads, and there is only one simple path from a place to the other one. In other words, the places and the roads in the Forest forms a tree.

Initially, there are n places in the Forest, numbered from 1 to n . Marisa's house is in 1, so the Forest can be regarded as a tree rooted on 1 from Marisa's point of view.

As all lost things will go into Gensokyo, there will be new places now and then, and the new places will also have exactly one bidirectional road connecting with an existing place, so that the Forest is still a tree after the new place appears.

There are many naughty fairies in the Forest too. Sometimes a fairy will go from a specific place to another place, and since fairies are nature incarnate, on each place passed by the fairy, k flowers with a specific color will grow out. The colors can be denoted as numbers from 1 to 10^9 .

Marisa likes flowers, and sometimes she wants to invite her friends to watch the flowers together. Before that, she will count the total number of the flowers with color number no greater than c , in the *subtree* of a specific place. We say a place v is in the *subtree* of u if u lies on the simple path between v and 1. Note that Marisa will not pick the flowers, so the counting will not change anything.

However, the Forest is too big, so she asks you to count the flowers each time instead. Also, you must answer her immediately each time she makes a query.

Input

The first line of input contains two integers n and q ($1 \leq n \leq 3 \times 10^4$, $0 \leq q \leq 10^5$), indicating there are n places in the Forest initially, and Marisa will give you q operations.

In each of the following $n - 1$ lines, there are two positive integers u_i and v_i ($1 \leq u_i, v_i \leq n$, $u_i \neq v_i$) separated by a single space, indicating that there is a bidirectional road connecting the u_i -th and the v_i -th place directly.

Each of the following q lines represents an operation by order, which has one of the three formats below:

- 1 u_i : A new place appears, and it is numbered $n' + 1$, where n' denotes the count of places before this operation. In addition, there is a road connecting the new place and the u_i -th ($1 \leq u_i \leq n'$) place.
- 2 u_i v_i c_i k_i : A fairy goes from the u_i -th place to the v_i -th place, and there will grow up k_i ($1 \leq k_i \leq 10^7$) flowers of color c_i ($1 \leq c_i \leq 10^9$) on each place on the only simple path between them, including both endpoints. It is guaranteed that the u_i -th and the v_i -th place exist.
- 3 u_i c_i : Marisa wants to know that, how many flowers are in the subtree of the u_i -th place, whose color numbers are no greater than c_i ($1 \leq c_i \leq 10^9$). The definition of the subtree of a place is mentioned above in the statement.

In addition, to make sure that you answers Marisa immediately each time, you need to keep a variable last, which equals to the value of the last answer modulo 2^{31} (or 0 initially). Each time you read a line of operation, all numbers except the first one should *bitwise XOR* to last, and the result is the actual value of that number.

The *bitwise XOR* of non-negative integers A and B , $A \oplus B$, is defined as follows:

- When $A \oplus B$ is written in base two, the digit in the 2^k 's place ($k \geq 0$) is 1 if exactly one of the digits in that place of A and B is 1, and 0 otherwise.

It is guaranteed that:

- Every number in input is a non-negative integer less than 2^{31} .
- After recovering using the above method, the actual values of the numbers meet the restrictions above.
- The total number of places will not exceed 5×10^4 .
- The total count of the second type of operation will not exceed 5×10^4 .

Output

For each operation of the third type, print a non-negative integer in a single line, denoting the count of flowers in the subtree of the u_i , which color is no greater than c_i .

Example

standard input	standard output
3 5	12
1 2	14
1 3	
2 2 3 1 4	
3 1 1	
1 13	
2 13 8 14 13	
3 13 14	

Note

The actual value of the numbers in the input should be:

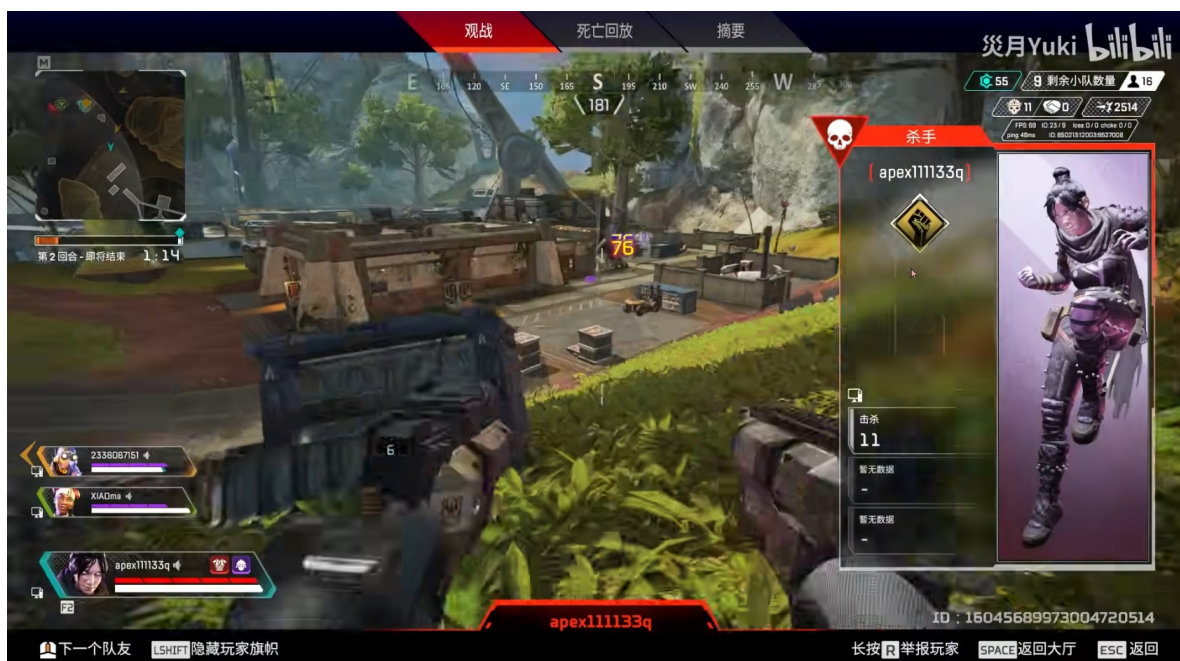
```
3 5
1 2
1 3
2 2 3 1 4
3 1 1
1 1
2 1 4 2 1
3 1 2
```

Problem G. Gua!

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 1024 megabytes

You are playing the game Apex Legends, which is a Battle Royale game famous for, uh, a large number of cheaters, a.k.a. Mr. Gua. ('Gua' means cheating in Chinese)

One type of cheating is changing the parameters of guns. For example, someone can even hold two guns parallelly like the following screenshot, which makes you wonder if she were the Lucian in the game League of Legends.



Bilibili Video: BV1V3411j7QW

After losing a huge amount of rating points in the ranking games, you are angry because of the arrogance of Mr. Gua and the ignorance of the game development company, Respawn. You decide to write a complaint letter to teach Respawn detecting cheaters.

Someone killed you with a gun that each bullet will deal **at maximum** B damages. This gun can shoot R Rounds Per Minute (RPM), meaning that after shooting a bullet the gun must rest for at least $1/R$ **minute**. There is a special case $R = 0$, which means that the player is without weapon or out of ammo, so no damage can be dealt.

In the replay, the player dealt D damages in S **seconds**, counting from the first bullet to the last bullet. If we say that this guy must be cheating by changing the parameters of guns, it means that he/she dealt more damage than he/she could. Here, we consider the maximum possible damage by only B, R, S , ignoring the other aspects in the games like magazine size, reload time and bugs.

We take the gun Wingman used in the above screenshot as an example. This gun can deal 38 damage when hit Fortified legends on the body, and the cheater dealt $152 = 38 \times 4$ damages to the enemy in just 1 second. Since the gun's rate of fire is only 156 RPM, it's only possible to wait $156/60 = 2.6$ cool-downs in a second, which means shoot 3 bullets at maximum. Because $4 > 3$, we can ensure that this guy is really cheating.

Write a demo program to determine whether a player must be cheating by B, R, D, S .

Input

The first line contains a single integer $T(1 \leq T \leq 10^3)$, denoting the number of test cases.

For each test case, there are four integers $B, R, D, S(0 \leq B, R, D, S \leq 2000)$ in a single line, denoting maximum single bullet damage, rate of fire in RPM, damage dealt by the player, the time window from the first bullet to the last bullet in seconds.

Output

For each test print a single line,

- if the player must be cheating, print “gua!” (without quotes),
- otherwise print “ok” (without quotes).

Example

standard input	standard output
7	gua!
38 156 152 1	ok
280 25 280 0	ok
99 51 9 10	gua!
0 0 1 1	gua!
99 0 1 1	ok
11 1080 209 1	gua!
11 1080 210 1	

Note

The first sample case is explained above.

For the second sample case, it's possible to use Kraber to do a head shot, in which we count the time window as 0. We are not sure whether it is really cheating, although it's possible that the player use some aiming aid cheating.

The real Bronze player plays normally in the third case, and it's a reminder that the player can deal 'partial' damage using shotgun like Peacekeeper. As long as the total amount of damage does not exceed the maximum possible damage, we can't determine whether it is really cheating or not.

The fourth and fifth are the cases when one player has no weapon or has out of ammo. Therefore, no damage can be dealt without help of the 'high technique'.

Problem H. Heirloom Painting

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

Little Desprado2, the great artist, has a small painting robot to do some artistic creations.

Today, he draws a **ring** divided by n grids, and there are m kinds of colors. He wants to paint the ring as he wants. However, because of some technical issues – using a heirloom printer nozzle to save cost, for example – the robot will paint **exactly** k continuous grids with the same color each time. In addition, the strong organic pigment can overlay the previous paintings, which means that the color applied later will **replace** the previous color.

Little Desprado2 wants to know the minimum number of times that his robot should paint from the empty grids to a given pattern, or it's impossible to do so.

Input

The first line contains one integer T ($1 \leq T \leq 10^5$), denoting the number of test cases.

For each test case, the first line contains three integers n , m and k ($1 \leq n, m \leq 10^6, 1 \leq k \leq n$), denoting the number of grids, colors and grids the robot will paint each time, respectively. The second line contains n numbers c_1, c_2, \dots, c_n ($1 \leq c_i \leq m$), c_i denotes the color of the i -th grid that little Desprado2 wants.

There are no color at the beginning, and you can consider the uncolored grids with color -1 for simplicity.

It is guaranteed that sum of n over all test cases won't exceed 10^6 .

Output

For each test case, print a single integer in a separated line - the minimal times the robot should paint. If the mission is impossible, print -1 .

Example

standard input	standard output
3	6
11 4 2	5
1 1 1 2 2 3 3 3 4 4 1	-1
5 2 1	
1 2 1 2 1	
6 2 2	
1 2 1 2 1 2	

Note

For the first example, one optimal strategy is:

1. Paint grid 11 and grid 1 in color 1. Note that this is a ring so grid 11 and grid 1 is adjacent.
2. Paint grid 2 and grid 3 in color 1.
3. Paint grid 4 and grid 5 in color 2.
4. Paint grid 6 and grid 7 in color 3.
5. Paint grid 8 and grid 9 in color 3.
6. Paint grid 9 and grid 10 in color 4. Note that the color in 9 is now replaced with 4 from 3.

Problem I. It Takes Two of Two

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 1024 megabytes

Nice cooperation helps. Bad cooperation sucks.



A sittable chair and a sleepy Sayu.

As the leader of Mihomo, a company producing sittable chairs, Koji is encouraging cooperation to enlarge the scale of production pipeline. In order to guarantee the quality of cooperation, here are three requirements that the cooperation relations must satisfy:

- Each cooperation relation involves 2 different employees.
- No two cooperation relations involve the same pair of employees.
- Each employee must participate in no more than 2 different cooperation relations.

There are n employees in the company, including Koji himself. Koji considered it very important to make fair cooperation. The dice never lie, so he decided to roll some dice to decide the fair division. The procedure of the dice rolling is as follows:

- Initially, the set R of cooperation relations is empty set (\emptyset), and then starts to enter a loop.
- In each iteration, two integers u, v are independently chosen from $1, 2, \dots, n$ uniformly at random. If $R \cup (u, v)$ meets above requirements, add (u, v) to R , otherwise leave R unchanged.
- Before the beginning of each iteration, if no more cooperation could be added to R , the program terminates immediately.

Koji wrote a program of the above procedure in just 114 seconds, but he finds that it had taken 514 seconds until the program finished its execution. Since Koji is not good at counting numbers above nine, he wants you to help him calculate the expected number of iterations of the above procedure in order to decide if there is a potential bug in the program.

Input

Input contains a single integer n ($1 \leq n \leq 200$), the number of employees in Mihomo.

Output

Print a single real number in decimal, denoting your answer.

Your answer will be considered correct, if the absolute error or relative error to the reference answer is less than 10^{-6} .

To print a fixed decimal number for some digits after the decimal point, say, 9 digits, you can use

- `printf("%.9lf\n", ans)` or `printf("%.9Lf\n", ans)` in C-style output;
- `cout << fixed << setprecision(9) << ans << endl` in C++;
- `print("%.9f" % ans)` in Python;
- see documentation for other languages.

Examples

standard input	standard output
1	0.000000000
2	2.000000000
3	8.250000000

Note

For the first sample case, Koji can cooperate with nobody, so the program terminates immediately.

For the second sample case, either $\{(1, 2)\}$ or $\{(2, 1)\}$ will be the final R , and with probability $1/2$ invalid cooperation relations $(1, 1), (2, 2)$ can be obtained. Therefore, the expected number of iterations is

$$1 \times (1/2) + 2 \times (1/4) + 3 \times (1/8) + \dots = 2.$$

Problem J. Just Some Bad Memory

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 1024 megabytes

Relax. Let me tell you in the fastest pace what you need to do.

Give you a *simple* graph $G = (V, E)$ consisting of undirected edges. You need to tell me, what is the minimum number of edges should you add to the graph, resulting a simple graph containing at least one *odd cycle* and at least one *even cycle*.

A *simple* graph is a graph without multiple edges and self-loops, which means that each edge connects two different vertices and no two edges connect the same pair of vertices.

A *cycle* is a sequence of distinct vertices $\{v_1, v_2, \dots, v_k\}$, such that $(v_i, v_{i \bmod k + 1}) \in E$. The *odd* or *even* describes the parity of k . A smallest odd cycle is of length 3, and a smallest even cycle is of length 4.

Input

The first line contains two integers n, m ($1 \leq n \leq 10^5, 0 \leq m \leq \min\{2 \times 10^5, \binom{n}{2}\}$), denoting the number of vertices ($|V|$) and the number of edges ($|E|$).

In the next m lines, each line contains two integers u, v ($1 \leq u, v \leq n, u \neq v$), denoting that there are edges connecting vertices u and v .

It's guaranteed that the input graph is a simple graph.

Output

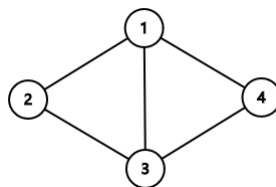
Print one integer in a single line, denoting your answer. If the mission is impossible, print '-1' instead.

Examples

standard input	standard output
3 3 1 2 2 3 1 3	-1
4 0	5
5 4 1 2 2 3 3 4 4 5	2
4 6 1 2 1 3 1 4 2 3 2 4 3 4	0
4 4 1 2 2 3 3 4 4 1	1
7 7 1 2 2 3 3 4 4 1 5 6 6 7 7 5	0

Note

Here is one possible solution of sample 2. The contained odd cycles are $\{1, 2, 3\}$ and $\{1, 3, 4\}$, and the only even cycle is $\{1, 2, 3, 4\}$.



Problem K. Known as the Fruit Brother

Input file: standard input
Output file: standard output
Time limit: 4 seconds
Memory limit: 1024 megabytes

In League of Legends, Blast Cone is a type of plant with explosive fruit. Their explosive properties are powerful enough to fling a humanoid several meters away.



Cryin used Blast Cone to speed up the movement of him and his teammates.

Suppose your character is in the Summoner's Rift now. For simplicity,

- The Summoner's Rift can be regarded as an infinite 2-dimensional plane, and your character is a point ignoring the radius;
- All the barriers are rectangles whose sides are parallel to coordinate axes, you can not go **strictly inside** the barriers, which means the borders are accessible;
- To use a blast cone, you can go to the location of it, destroy it, and then you can control yourself to jump to **any position except those in the barrier** within the distance of R . You can assume that the process is instantaneous, that is, it won't take any time.

There are n rectangle barriers and m Blast Cones. Now you start from the starting point (x_s, y_s) and you want to reach the end point (x_t, y_t) . Your moving speed is 1 unit length per second. Can you calculate the minimum time cost from starting point to end point?

Input

The first line contains three integers n, m, R ($0 \leq n, m \leq 40, 1 \leq R \leq 10^6$), denoting the number of rectangle barriers, the number of Blast Cones and the jump radius, respectively.

In the following n lines, the i -th line contains four integers $x_{i,1}^b, y_{i,1}^b, x_{i,2}^b, y_{i,2}^b$ ($-10^6 \leq x_{i,1}^b, y_{i,1}^b, x_{i,2}^b, y_{i,2}^b \leq 10^6, x_{i,1}^b < x_{i,2}^b, y_{i,1}^b < y_{i,2}^b$), which denotes the i -th barrier's bottom left corner is $(x_{i,1}^b, y_{i,1}^b)$ and its upper right corner is $(x_{i,2}^b, y_{i,2}^b)$.

In the following m lines, the i -th line contains two integers x_i^c, y_i^c ($-10^6 \leq x_i^c, y_i^c \leq 10^6$), which denotes the i -th Blast Cone is at (x_i^c, y_i^c) .

The last line contains four integers x_s, y_s, x_t, y_t ($-10^6 \leq x_s, y_s, x_t, y_t \leq 10^6$), denoting the starting point s and the end point t .

It is guaranteed that:

- Any two barriers don't share a point, including the borderlines and the corners.

- The positions of Blast Cones, the starting point s and the end point t are pairwise distinct, and are all strictly outside the barriers.

Output

Output a decimal real number in a single line, denoting the minimum time cost from starting point to end point. Your answer will be judged as correct, if the relative or absolute error between your answer and jury's answer is less than or equal to 10^{-6} .

It's guaranteed that there are valid solutions by the given condition.

To print a fixed decimal number for some digits after the decimal point, say, 9 digits, you can use

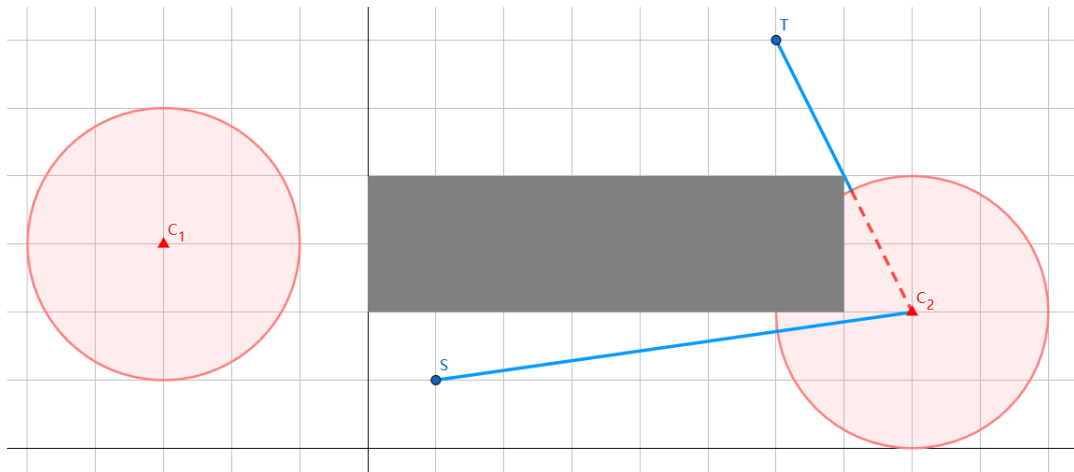
- `printf("%.9lf\n", ans)` or `printf("%.9Lf\n", ans)` in C-style output;
- `cout << fixed << setprecision(9) << ans << endl` in C++;
- `print("%.9f" % ans)` in Python.

Example

standard input	standard output
1 2 2 0 2 7 4 -3 3 8 2 1 1 6 6	9.543203767

Note

The figure of the sample input is as follows:



The answer is $\sqrt{7^2 + 1^2} + \sqrt{2^2 + 4^2} - 2 \approx 9.543203767$.

Problem L. Last Warning of the Competition Finance Officer

Input file: standard input
Output file: standard output
Time limit: 5 seconds
Memory limit: 1024 megabytes

As you might know, people from the ancient always stick to the old form, like the famous Kong Yiji. The Competition Finance Officer from the great Qin dynasty – who chose the Hot Spring Hotel without cell phone signal and clean tap water for Easter Chicken Finals – always speaks in an odd manner. For example, whenever you present some facts to him that he is unwilling to see, he would shout out ‘QFMYQ!!!!!! QFMYQ!!!!!! QFMYQ!!!!!!’ (QFMYQ means violation of the right of reputation, and its Chinese pronunciation is ‘qin fan ming yu quan’).

The competition participants, known as Easter Chicken predators, understand statistics and probabilities well and use them as weapons. As other predators are practicing how to use randomness and constructions to kill the intermediate boss Hash, AntiLeaf, the No. 1 seed, is already planning to kill the final boss Easter Chicken. To do so, he wants to analyze the frequency of words spoken by the Finance Officer in order to imitate him by some machine learning tricks – something not ancient at all. After that, AntiLeaf can easily sneak into the arena without triggering warnings by imitating the Finance Officer, who considers the Easter Chicken his own property and wants to make money from it.

A sample sentence said by the Finance Officer can be represented by a string s containing lowercase English letters. AntiLeaf has already selected n lowercase English words $\{t_i\}$ as the dictionary of Finance Officer’s classic quotes. For t_i , it has a corresponding value v_i , denoting how classic it is.

To predict the Finance Officer’s behavior, AntiLeaf want to calculate the *score* of each prefix of s , which is defined as follows:

- The *score* of a string u is defined as the sum of values of all possible *extractions*.
- An *extraction* of a string u , is a set of **disjoint** substrings in u , each of which is in the dictionary.
- The value of an extraction is the product of values of all substrings it has chosen. The value of empty set is defined as 1.
- Two substrings of u are considered different by their positions in u , so a string in the dictionary can appear more than once in an extraction, and they would all contribute to the product.
- Two extractions from u are different, if one contains a substring that the other doesn’t have.

Since the answers may be very huge, you just need to output the answers modulo 998 244 353.

Input

The first line of input contains a string s ($1 \leq |s| \leq 2 \times 10^5$), which contains only lowercase English letters.

The second line contains a positive integer n , denoting the number of classical quotes.

In the following n lines, the i -th line contains a lowercase English string t_i ($1 \leq |t_i| \leq 2 \times 10^5$) and a positive integer v_i ($0 < v_i < 998\,244\,353$), denoting the i -th classical word and the value of it.

It is guaranteed that all of t_i are pairwise distinct, and the sum of their lengths does not exceed 2×10^5 , and

Output

Output $|s|$ numbers separated with spaces in a single line, the i -th of which represents the answer of the prefix of s with length i , modulo 998 244 353.

Examples

standard input	standard output
ababa 2 aba 2 ba 3	1 1 6 6 26
qfmyqqfmyqqfmyq 2 qfmyq 11111 myqq 404968002	1 1 1 1 11112 405079114 405079114 405079114 405079114 771912310 239058268 239058268 239058268 239058268 31169271
wwwsoupunetcom 2 money 999999 soup 998244352	1 1 1 1 1 1 0 0 0 0 0 0 0 0

Note

Here is the explanation of the last answer of the first example. Let us denote the unused letters with dots, and different substrings are denoted by underlines and overlines.

- $\underline{ab}\overline{aba} 6 = 2 \times 3$
- $\underline{aba} . . 2$
- $. . \underline{aba} 2$
- $. \underline{ba} . . 3$
- $. . . \underline{ba} 3$
- $. \underline{baba} 9 = 3 \times 3$
- $. 1$

The answer is

$$(6 + 2 + 2 + 3 + 3 + 9 + 1) \bmod 998\,244\,353 = 26.$$

The second sample's output is wrapped for the convenience of printing, in real test data it contains no extra line breaks.

Problem M. My University Is Better Than Yours

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 1024 megabytes

People rank for things. Yes, ranking is nothing for most of the time, except when you are doing year-end report to your boss.

Under the promotion of the construction of world-class universities, a lot of universities are struggling to improve ranking in every way. Publishing papers, applying for funds, improving diversity... They are too hard and your result may not be fairly judged by institutions like US News and Times! However, *some* universities are more *clever* – they publish their own rankings, which makes the ranking *indirectly better*. For example, using Shanghai Ranking's Academic Ranking of World Universities (ARWU) produced by Shanghai Jiao Tong University, Desprado2 can prove that his school is better than MIT.

My University Is Better Than Your University

My University

Shanghai Jiao Tong University

Your University

Massachusetts Institute of Technology - MIT

zizhengfang.com/applets/transitivity

QS	Shanghai Jiao Tong University (#46)	>	Washington University in St. Louis (#118)
ARWU	Washington University in St. Louis (#23)	>	Zhejiang University (#52)
QS	Zhejiang University (#42)	>	University of California - Los Angeles (#44)
ARWU	University of California - Los Angeles (#14)	>	University of Pennsylvania (#15)
QS	University of Pennsylvania (#13)	>	University of California - Berkeley (#27)
ARWU	University of California - Berkeley (#5)	>	California Institute of Technology - Caltech (#9)
Times	California Institute of Technology - Caltech (#2)	>	Massachusetts Institute of Technology - MIT (#5)

Therefore, Shanghai Jiao Tong University is better than Massachusetts Institute of Technology - MIT.

Flip

Share

<https://www.zizhengfang.com/applets/transitivity>

Anyway, that is a joke unless you are finding jobs and need to brag about your school. But at the same time, Desprado2 comes out a problem: assume there are n universities in total, and he has collected m university rankings. For simplicity, all the universities are denoted by a number from 1 to n . Here, Desprado2 defines that university x is **directly better** than university y , if and only if there exists a university ranking such that university x ranks higher than university y . Furthermore, Desprado2 defines that university x is **better** than university y , if and only if there exists a sequence $\{s_1, s_2, \dots, s_k\}$ ($k \geq 2$), such that:

- $s_1 = x, s_k = y$
- $\forall i \in \{1, 2, \dots, k-1\}$, university s_i is **directly better** than university s_{i+1}

For each university, Desprado2 want you to tell him it is **better** than how many of **other** universities.

Input

The first line contains two integers n and m ($1 \leq n \leq 5 \times 10^5$, $1 \leq m \leq 5 \times 10^5$, $1 \leq n \times m \leq 10^6$), denoting the number of universities considered, and the number of university rankings.

Then follows m lines. The i -th line contains n distinct integers $s_{i,1}, s_{i,2}, \dots, s_{i,n}$ ($1 \leq s_{i,j} \leq n$), denoting the order of the i -th university ranking (from high to low).

Output

Output one line with n integers a_1, a_2, \dots, a_n , separated by spaces. The i -th number a_i denotes the number of universities that university i is **better** than.

Examples

standard input	standard output
4 2 1 2 3 4 1 3 2 4	3 2 2 0
4 2 1 2 3 4 4 3 2 1	3 3 3 3
5 2 5 4 3 2 1 5 4 3 2 1	0 1 2 3 4

Problem N. Nine Is Greater Than Ten

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 1024 megabytes

Koji, the mathematician and philosopher, is hereby proving the statement that

$$9 > 10$$

by the following solid proof.

Proof. We write down two numbers in decimal representation.

9.

10

Look at the first digit that they are different at! Since $9 > 1$, we proved that $9 > 10$. *Q.E.D.*

Again, you can easily use the same argument to prove $114514 < 1919$:

114514

1919..

and $9 < 999$:

9.9.

999

Wow, that is math! How logical it is! Now you have fully understood how the Koji compares two numbers. Write a program to compare two input numbers!

Input

A single input line consists of two positive integers a, b ($1 \leq a, b \leq 20\,220\,924$) separated by a single space, both are guaranteed without leading zeros.

Output

Print a single line by the method of Koji Comparison,

- If $a > b$, print " $a>b$ "(without quotes).
- If $a < b$, print " $a<b$ "(without quotes).
- If $a = b$, print " $a=b$ "(without quotes).

Examples

standard input	standard output
9 10	9>10
114514 1919	114514<1919
9 999	9<999
99 99	99=99
2022 924	2022<924