Problem A : Carpenters' Language

Time Limit : 1 sec, Memory Limit : 65536 KB

International Carpenters Professionals Company (ICPC) is a top construction company with a lot of expert carpenters. What makes ICPC a top company is their original language.

The syntax of the language is simply given in CFG as follows:

```
S -> SS | (S) | )S( | ε
```

In other words, a right parenthesis can be closed by a left parenthesis and a left parenthesis can be closed by a right parenthesis in this language.

Alex, a grad student mastering linguistics, decided to study ICPC's language. As a first step of the study, he needs to judge whether a text is well-formed in the language or not. Then, he asked you, a great programmer, to write a program for the judgement.

Alex's request is as follows: You have an empty string S in the beginning, and construct longer string by inserting a sequence of '(' or ')' into the string. You will receive q queries, each of which consists of three elements (p, c, n), where p is the position to insert, n is the number of characters to insert and c is either '(' or ')', the character to insert. For each query, your program should insert c repeated by n times into the p-th position of S from the beginning. Also it should output, after performing each insert operation, "Yes" if S is in the language and "No" if S is not in the language.

Please help Alex to support his study, otherwise he will fail to graduate the college.

Input

The first line contains one integer q $(1 \le q \le 10^5)$ indicating the number of queries, follows q lines of three elements, p_i, c_i, n_i, separated by a single space $(1 \le i \le q$, $c_i = '('or')'$, $0 \le p_i \le$ length of S before i-th query, $1 \le n \le 2^{20}$). It is guaranteed that all the queries in the input are valid.

Output

For each query, output "Yes" if S is in the language and "No" if S is not in the language.

Sample Input 1

```
3
0 ( 10
10 ) 5
10 ) 5
```

Output for the Sample Input 1

```
No
No
Yes
```

Sample Input 2

```
3
0 ) 10
10 ( 5
10 ( 5
```

Output for the Sample Input 2

```
No
No
Yes
```

Sample Input 3

```
3
0 ( 10
10 ) 20
0 ( 10
```

Output for the Sample Input 3

```
No
No
Yes
```

Problem B : Kth Sentence

Time Limit : 3 sec, Memory Limit : 65536 KB

A student, Kita_masa, is taking an English examination. In this examination, he has to write a sentence of length m.

Since he completely forgot the English grammar, he decided to consider all sentences of length m constructed by concatenating the words he knows and write the K-th sentence among the candidates sorted in lexicographic order. He believes that it must be the correct sentence because K is today's lucky number for him.

Each word may be used multiple times (or it's also fine not to use it at all) and the sentence does not contain any extra character between words. Two sentences are considered different if the order of the words are different even if the concatenation resulted in the same string.

Input

The first line contains three integers n $(1 \leq n \leq 100)$, m $(1 \leq m \leq 2000)$ and K $(1 \leq K \leq 10^{18})$, separated by a single space. Each of the following n lines contains a word that Kita_masa knows. The length of each word is between 1 and 200, inclusive, and the words contain only lowercase letters. You may assume all the words given are distinct.

Output

Print the K-th (1-based) sentence of length m in lexicographic order. If there is no such a sentence, print "-".

Sample Input 1

```
2 10 2
hello
world
```

Output for the Sample Input 1

```
helloworld
```

Sample Input 2

```
3 3 6
a
aa
b
```

Output for the Sample Input 2

```
aba
```

Sample Input 3

```
2 59 1000000000000000000
a
b
```

Output for the Sample Input 3

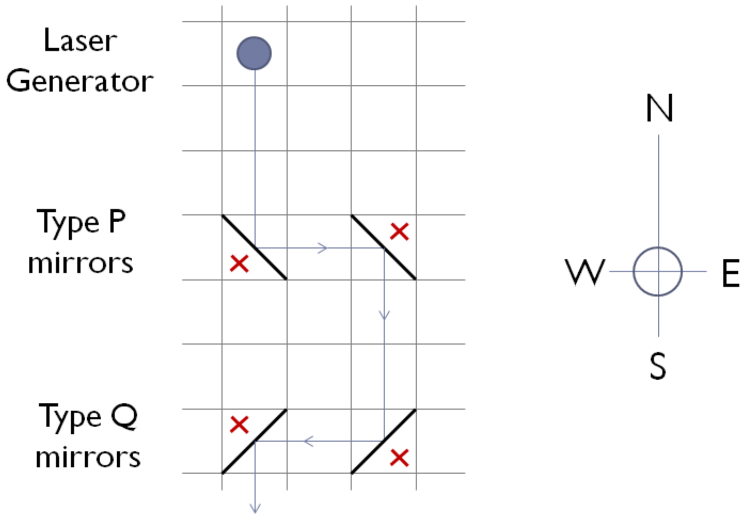
```
-
```

Problem C : A Light Road

Time Limit : 2 sec, Memory Limit : 65536 KB

There is an evil creature in a square on N-by-M grid ($2 \le N,M \le 100$), and you want to kill it using a laser generator located in a different square. Since the location and direction of the laser generator are fixed, you may need to use several mirrors to reflect laser beams. There are some obstacles on the grid and you have a limited number of mirrors. Please find out whether it is possible to kill the creature, and if possible, find the minimum number of mirrors.

There are two types of single-sided mirrors; type P mirrors can be placed at the angle of 45 or 225 degrees from east-west direction, and type Q mirrors can be placed with at the angle of 135 or 315 degrees. For example, four mirrors are located properly, laser go through like the following.



Note that mirrors are single-sided, and thus back side (the side with cross in the above picture) is not reflective. You have A type P mirrors, and also A type Q mirrors ($0 \le A \le 10$). Although you cannot put mirrors onto the squares with the creature or the laser generator, laser beam can pass through the square. Evil creature is killed if the laser reaches the square it is in.

Input

Each test case consists of several lines.

The first line contains three integers, N, M, and A. Each of the following N lines contains M characters, and represents the grid information. '#', '.', 'S', 'G' indicates obstacle, empty square, the location of the laser generator, and the location of the evil creature, respectively. The first line shows the information in northernmost squares and the last line shows the information in southernmost squares. You can assume that there is exactly one laser generator and exactly one creature, and the laser generator emits laser beam always toward the south.

Output

Output the minimum number of mirrors used if you can kill creature, or -1 otherwise.

Sample Input 1

```
3 3 2
S#.
...
.#G
```

Output for the Sample Input 1

```
2
```

Sample Input 2

```
3 3 1
S#.
...
.#G
```

Output for the Sample Input 2

```
-1
```

Sample Input 3

```
3 3 1
S#G
...
.#.
```

Output for the Sample Input 3

```
2
```

Sample Input 4

```
4 3 2
S..
...
..#
.#G
```

Output for the Sample Input 4

```
-1
```

Problem D : Matrix Operation

Time Limit : 2 sec, Memory Limit : 65536 KB

You are a student looking for a job. Today you had an employment examination for an IT company. They asked you to write an efficient program to perform several operations. First, they showed you an $N \times N$ square matrix and a list of operations. All operations but one modify the matrix, and the last operation outputs the character in a specified cell. Please remember that you need to output the final matrix after you finish all the operations.

Followings are the detail of the operations:

WR r c v
    (Write operation) write a integer v into the cell (r,c)
    $(1 \leq v \leq 1{,}000{,}000)$
CP r1 c1 r2 c2
    (Copy operation) copy a character in the cell (r1,c2) into the cell (r2,c2)
SR r1 r2
    (Swap Row operation) swap the r1-th row and r2-th row
SC c1 c2
    (Swap Column operation) swap the c1-th column and c2-th column
RL
    (Rotate Left operation) rotate the whole matrix in counter-clockwise direction by 90 degrees
RR
    (Rotate Right operation) rotate the whole matrix in clockwise direction by 90 degrees
RH
    (Reflect Horizontal operation) reverse the order of the rows
RV
    (Reflect Vertical operation) reverse the order of the columns

Input

First line of each testcase contains nine integers. First two integers in the line, N and Q, indicate the size of matrix and the number of queries, respectively $(1 \leq N,Q \leq 40{,}000)$. Next three integers, A B, and C, are coefficients to calculate values in initial matrix

$(1 \le A,B,C \le 1{,}000{,}000)$, and they are used as follows:
$A_{r,c} = (r * A + c * B) mod C$ where r and c are row and column indices, respectively $(1 \le r,c \le N)$. Last four integers, D, E, F, and G, are coefficients to compute the final hash value mentioned in the next section $(1 \le D \le E \le N, 1 \le F \le G \le N, E - D \le 1{,}000, G - F \le 1{,}000)$. Each of next Q lines contains one operation in the format as described above.

Output

Output a hash value h computed from the final matrix B by using following pseudo source code.

```
h <- 314159265
for r = D...E
  for c = F...G
    h <- (31 * h + B_{r,c}) mod 100,000,007
```

where "<-" is a destructive assignment operator, "for i = S...T" indicates a loop for i from S to T (both inclusive), and "mod" is a remainder operation.

Sample Input 1

```
2 1 3 6 12 1 2 1 2
WR 1 1 1
```

Output for the Sample Input 1

```
676573821
```

Sample Input 2

```
2 1 3 6 12 1 2 1 2
RL
```

Output for the Sample Input 2

```
676636559
```

Sample Input 3

```
2 1 3 6 12 1 2 1 2
RH
```

Output for the Sample Input 3

```
676547189
```

Sample Input 4

```
39989 6 999983 999979 999961 1 1000 1 1000
SR 1 39989
SC 1 39989
RL
RH
RR
RV
```

Output for the Sample Input 4

```
458797120
```

Problem E : Multi-ending Story

Time Limit : 5 sec, Memory Limit : 65536 KB

You are a programmer who loves bishojo games (a sub-genre of dating simulation games). A game, which is titled "I * C * P * C!" and was released yesterday, has arrived to you just now. This game has multiple endings. When you complete all of those endings, you can get a special figure of the main heroine, Sakuya. So, you want to hurry and play the game! But, let's calm down a bit and think how to complete all of the endings in the shortest time first.

In fact, you have a special skill that allows you to know the structure of branching points of games. By using the skill, you have found out that all of the branching points in this game are to select two choices "Yes" or "No", and once a different choice is taken the branched stories flow to different endings; they won't converge any more, like a binary tree. You also noticed that it takes exactly one minute to proceed the game from a branching point to another branching point or to an ending. In addition, you can assume it only takes negligible time to return to the beginning of the game (``reset'') and to play from the beginning to the first branching point.

The game has an additional feature called "Quick Save", which can significantly reduce the playing time for completion. The feature allows you to record the point where you are currently playing and return there at any time later. You can record any number of times, but you can hold only the last recorded point. That is, when you use Quick Save, you overwrite the previous record. If you want to return to the overwritten point, you must play the game from the beginning once again.

Well, let's estimate how long it will take for completing all of the endings in the shortest time.

Input

A data set is given in the following format.

The first line of the data set contains one integer N ($2 \le N \le 500{,}000$), which denotes the number of the endings in this game. The following $N-1$ lines describe the branching points. The i-th line describes the branching point of ID number i and contains two integers Yes_i and No_i ($i+1 \le Yes_i, No_i \le N$), which denote the ID numbers of the next branching points when you select Yes or No respectively. $Yes_i = N$ means that you can reach an ending if you select Yes, and so for $No_i = N$. The branching point with ID 1 is the first branching point. The branching points with ID between 2 and $N-1$ (inclusive) appear exactly once in Yes_i's and No_i's.

Output

Print the shortest time in a line.

Sample Input 1

```
4
2 3
4 4
4 4
```

Output for the Sample Input 1

```
6
```

Sample Input 2

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

Output for the Sample Input 2

```
8
```

Problem F : Network Reliability

Time Limit : 3 sec, Memory Limit : 65536 KB

An undirected graph is given. Each edge of the graph disappears with a constant probability. Calculate the probability with which the remained graph is connected.

Input

The first line contains three integers N $(1 \le N \le 14)$, M $(0 \le M \le 100)$ and P $(0 \le P \le 100)$, separated by a single space. N is the number of the vertices and M is the number of the edges. P is the probability represented by a percentage.

The following M lines describe the edges. Each line contains two integers v_i and u_i $(1 \le u_i, v_i \le N)$. (u_i, v_i) indicates the edge that connects the two vertices u_i and v_i.

Output

Output a line containing the probability with which the remained graph is connected. Your program may output an arbitrary number of digits after the decimal point. However, the absolute error should be 10^{-9} or less.

Sample Input 1

```
3 3 50
1 2
2 3
3 1
```

Output for the Sample Input 1

```
0.500000000000
```

Sample Input 2

```
3 3 10
1 2
2 3
```

```
3 1
```

Output for the Sample Input 2

```
0.972000000000
```

Sample Input 3

```
4 5 50
1 2
2 3
3 4
4 1
1 3
```

Output for the Sample Input 3

```
0.437500000000
```

Problem G : Runaway Domino

Time Limit : 3 sec, Memory Limit : 65536 KB

‟Domino effect'' is a famous play using dominoes. A player sets up a chain of dominoes stood. After a chain is formed, the player topples one end of the dominoes. The first domino topples the second domino, the second topples the third and so on.

You are playing domino effect. Before you finish to set up a chain of domino, a domino block started to topple, unfortunately. You have to stop the toppling as soon as possible.

The domino chain forms a polygonal line on a two-dimensional coordinate system without self intersections. The toppling starts from a certain point on the domino chain and continues toward the both end of the chain. If the toppling starts on an end of the chain, the toppling continue toward the other end. The toppling of a direction stops when you touch the toppling point or the toppling reaches an end of the domino chain.

You can assume that:

- You are a point without volume on the two-dimensional coordinate system.
- The toppling stops soon after touching the toppling point.
- You can step over the domino chain without toppling it.

You will given the form of the domino chain, the starting point of the toppling, your coordinates when the toppling started, the toppling velocity and the your velocity. You are task is to write a program that calculates your optimal move to stop the toppling at the earliest timing and calculates the minimum time to stop the toppling.

Input

The first line contains one integer N, which denotes the number of vertices in the polygonal line of the domino chain ($2 \leq N \leq 1000$).

Then N lines follow, each consists of two integers x_i and y_i, which denote the coordinates of the i-th vertex $(-10{,}000 \le x{,}y \le 10000)$.
The next line consists of three integers x_t, y_t and v_t, which denote the coordinates of the starting point and the velocity of the toppling.
The last line consists of three integers x_p, y_p and v_p, which denotes the coordinates of you when the toppling started and the velocity $(1 \le v_t < v_p \le 10)$. You may assume that the starting point of the toppling lies on the polygonal line.

Output

Print the minimum time to stop the toppling. The output must have a relative or absolute error less than 10^{-6}.

Sample Input 1

```
2
0 0
15 0
5 0 1
10 10 2
```

Output for the Sample Input 1

```
5.0
```

Sample Input 2

```
3
-10 0
0 0
0 10
-1 0 1
3 0 2
```

Output for the Sample Input 2

```
4.072027
```

Sample Input 3

```
2
0 0
10 0
5 0 1
9 0 3
```

Output for the Sample Input 3

2.0

Problem H : Sunny Graph

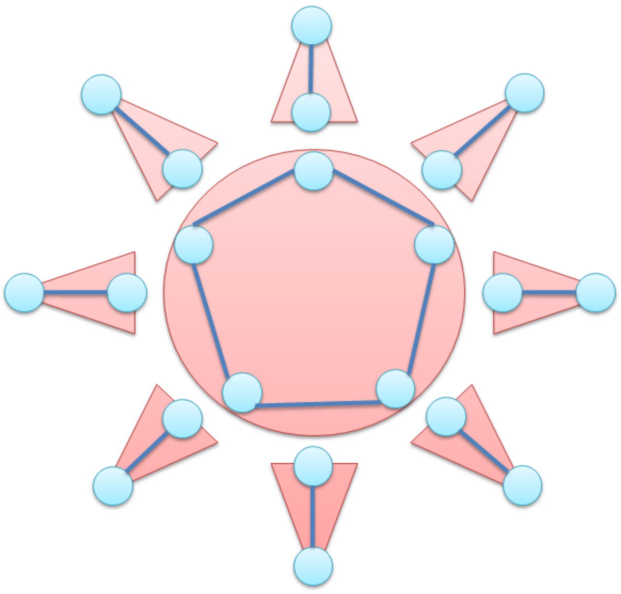
Time Limit : 3 sec, Memory Limit : 65536 KB

The Sun is a great heavenly body. The Sun is worshiped by various religions. Bob loves the Sun and loves any object that is similar to the Sun. He noticed that he can find the shape of the Sun in certain graphs. He calls such graphs "Sunny".

We define the property "Sunny" mathematically. A graph $G = (V,E)$ with a vertex $v \in V$ is called "Sunny" when there exists a subgraph $G' = (V,E'), E' \subseteq E$ that has the following two properties. (Be careful, the set of vertices must be the same.)

1. The connected component containing v is a cycle that consists of three or more vertices.
2. Every other component has exactly two vertices.

The following picture is an example of a subgraph $G' = (V,E')$ that has the above property.



Given a simple graph (In each edge, two end points are different. Every pair of vertices has one or no edge.) $G = (V,E)$, write a program that decides whether the given graph with the vertex 1 is "Sunny" or not.

Input

The first line contains two integers N (odd, $1 \le N \le 200$) and M ($0 \le M \le 20{,}000$), separated by a single space. N is the number of the vertices and M is the number of the edges.

The following M lines describe the edges. Each line contains two integers v_i and u_i ($1 \le u_i, v_i \le N$). (u_i, v_i) indicates the edge that connects the two vertices u_i and v_i. u_i and v_i are different, and every pair (u_i, v_i) are different.

Output

Print a line containing "Yes" when the graph is "Sunny". Otherwise, print "No".

Sample Input 1

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

Output for the Sample Input 1

```
Yes
```

Sample Input 2

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

Output for the Sample Input 2

```
No
```

Problem I : Testing Circuits

Time Limit : 5 sec, Memory Limit : 65536 KB

A Boolean expression is given. In the expression, each variable appears exactly once. Calculate the number of variable assignments that make the expression evaluate to true.

Input

A data set consists of only one line. The Boolean expression is given by a string which consists of digits, x, (,), |, &, and ~. Other characters such as spaces are not contained. The expression never exceeds 1,000,000 characters. The grammar of the expressions is given by the following BNF.

```
<expression> ::= <term> | <expression> "|" <term>
<term> ::= <factor> | <term> "&" <factor>
<factor> ::= <variable> | "~" <factor> | "(" <expression> ")"
<variable> ::= "x" <number>
<number> ::= "1" | "2" |... | "999999" | "1000000"
```

The expression obeys this syntax and thus you do not have to care about grammatical errors. When the expression contains N variables, each variable in {x1, x2,..., xN} appears exactly once.

Output

Output a line containing the number of variable assignments that make the expression evaluate to true in modulo 1,000,000,007.

Sample Input 1

```
(x1&x2)
```

Output for the Sample Input 1

```
1
```

Sample Input 2

```
(x1&x2)|(x3&x4)|(~(x5|x6)&(x7&x8))
```

Output for the Sample Input 2

```
121
```

Problem J : World Trip

Time Limit : 5 sec, Memory Limit : 65536 KB

Kita_masa is planning a trip around the world. This world has N countries and the country i has M_i cities. Kita_masa wants to visit every city exactly once, and return back to the starting city.

In this world, people can travel only by airplane. There are two kinds of airlines: domestic and international lines. Since international airports require special facilities such as customs and passport control, only a few cities in each country have international airports.

You are given a list of all flight routes in this world and prices for each route. Now it's time to calculate the cheapest route for Kita_masa's world trip!

Input

The first line contains two integers N and K, which represent the number of countries and the number of flight routes, respectively. The second line contains N integers, and the i-th integer M_i represents the number of cities in the country i. The third line contains N integers too, and the i-th integer F_i represents the number of international airports in the country i. Each of the following K lines contains five integers [Country 1] [City 1] [Country 2] [City 2] [Price]. This means that, there is a bi-directional flight route between the [City 1] in [Country 1] and the [City 2] in [Country 2], and its price is [Price].

Note that cities in each country are numbered from 1, and the cities whose city number is smaller or equal to F_i have international airports. That is, if there is a flight route between the city c_1 in the country n_1 and the city c_2 in the country n_2 with $n_1 \neq n_2$, it must be $c_1 \leq F_{n_1}$ and $c_2 \leq F_{n_2}$. You can assume that there's no flight route which departs from one city and flies back to the same city, and that at most one flight route exists in this world for each pair of cities.

The following constraints hold for each dataset:

- $1 \leq N \leq 15$
- $1 \leq M_i \leq 15$
- $1 \leq F_i \leq 4$
- $sum(F_i) \leq 15$
- $1 \leq Price \leq 10{,}000$

Output

Print a line that contains a single integer representing the minimum price to make a world trip.

If such a trip is impossible, print -1 instead.

Sample Input 1

```
4 6
1 1 1 1
1 1 1 1
1 1 2 1 1
1 1 3 1 2
1 1 4 1 1
2 1 3 1 1
2 1 4 1 2
3 1 4 1 1
```

Output for the Sample Input 1

```
4
```

Sample Input 2

```
2 16
4 4
2 2
1 1 1 2 1
1 1 1 3 2
1 1 1 4 1
1 2 1 3 1
1 2 1 4 2
1 3 1 4 1
2 1 2 2 1
2 1 2 3 2
2 1 2 4 1
2 2 2 3 1
2 2 2 4 2
2 3 2 4 1
1 1 2 1 1
1 1 2 2 2
```

```
1 2 2 1 2
1 2 2 2 1
```

Output for the Sample Input 2

```
8
```

Sample Input 3

```
2 2
2 1
1 1
1 1 1 2 1
1 1 2 1 1
```

Output for the Sample Input 3

```
-1
```