Problem A. Automorphism

Input file:	`auto.in`
Output file:	`auto.out`
Time limit:	6 seconds
Memory limit:	256 Mebibytes

You are given an undirected tree of n vertices. Vertices are numbered respectively from 1 to n.

An automorphism of a tree is a permutation s of the vertex set of this tree, such that for any edge $e = (u, v)$, $s(e) = (s(u), s(v))$ is also an edge.

Your task is to calculate number of automorphisms of a given tree modulo $10^9 + 7$.

Input

First line of the input file contains one integer n ($1 \leqslant n \leqslant 5 \cdot 10^5$) — number of vertices of a tree. Each of next $n - 1$ lines contains two integers u_i and v_i ($1 \leqslant u_i < v_i \leqslant n$) denoting an edge connecting vertices u_i and v_i.

Output

Print one integer — number of automorphisms of a given tree modulo $10^9 + 7$.

Example

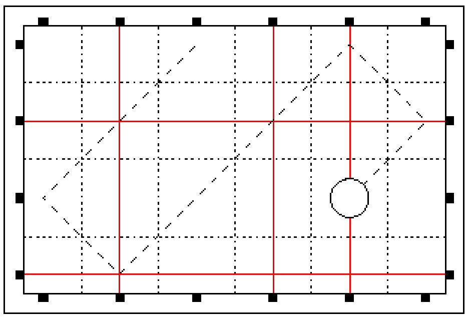
auto.in	auto.out

6
1 3
2 3
3 4
4 5
4 6
	8

Problem B. Laser Billiards

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

Recently, a new billiards variation was invented in Byteland, named “laser billiards”. The table for this game is n decimeters long and m decimeters wide. For the purposes of explanation, we will refer to length as horizontal direction and width as vertical direction, as if the table was drawn on a two-dimensional plane with sides parallel to coordinate axes and corners at points $(0, 0)$, $(n, 0)$, $(0, m)$ and (n, m). Along the edges of the table there is a barrier of width $1/4$ decimeters. There are also $n + m$ laser-receiver pairs mounted in the barriers in such a way that n vertical and m horizontal laser beams cover the table. The i-th $(1 \leqslant i \leqslant n)$ vertical beam crosses the j-th $(1 \leqslant j \leqslant m)$ horizontal beam at point $(i - 1/2, j - 1/2)$. Before the game, some laser beams are turned on, others are turned off. During the game, the state of laser beams does not change.



A ball for laser billiards has diameter $1/2$ decimeters. For each moment of time when a ball crosses one or more laser beams, the player is awarded one point. The next point can be awarded for the intersection of the same laser beam only if the laser beam and the ball weren’t overlapping for some time after the previous intersection.

At the start of the game, the ball is positioned at point $(x - 1/2, y - 1/2)$. After that, it starts moving uniformly with velocity (x_v, y_v) (in decimeters per second) for t seconds. The collisions of the ball with the barrier are elastic. The effect of friction is negligible.

Your task is to calculate the total number of points awarded to the player. The starting and ending moments of time are also counted, e. g. if the ball intersects one or more laser beams in the initial position, the player is awarded one point at time 0.

Input

The first line of input contains two integers n and m, the dimensions of the table $(3 \leqslant n, m \leqslant 10^5)$.

The second line contains n characters each of which is either ‘`0`’ or ‘`1`’. This line describes the state of vertical laser beams: if i-th character is ‘`0`’, i-th vertical laser is turned off; if it is ‘`1`’, the laser is turned on.

The third line contains m characters each of which is either ‘`0`’ or ‘`1`’. This line describes the state of horizontal laser beams in a similar fashion: if j-th character is ‘`0`’, j-th horizontal laser is turned off; if it is ‘`1`’, the laser is turned on.

The fourth line contains an integer k, the number of test cases ($1 \leqslant k \leqslant 10^4$). Each of the next k lines contains five integers x, y, x_v, y_v and t which are the coordinates of the initial position of the ball, the coordinates of velocity and the total time ($1 < x < m$, $1 < y < n$, x_v, $y_v \in \{-1, 1\}$, $1 \leqslant t \leqslant 10^9$).

Output

Output k lines. Line i should contain one integer: the total number of points awarded to the player in i-th test case.

Example

billiards.in	billiards.out

4 6
1010
010110
1
5 2 1 1 8
	6

3 3
011
100
2
2 2 -1 -1 68
2 2 1 1 76
	69
	77

Problem C. Smart Dog

Input file:	`dog.in`
Output file:	`dog.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

As an experiement, researchers blindfold one dog and put it in a maze laid out on a square grid. They put a cat somewhere else in the maze. Once every second, the cat takes a step one unit in a random direction (no diagonals), chosen uniformly at random among the directions that aren't blocked by walls.

Sometimes, when the cat takes a step, it makes a little bit of noise at its new position. The dog hears the noise with a certain amount of uncertainty s — that is, if the cat is at position (x, y), then the dog thinks that it heard the cat at position (x_1, y_1) with probability $N \cdot e^{(((x-x_1)^2+(y-y_1)^2)/(2s^2))}$, where N is an appropriate normalization factor.

After every step the cat takes, the dog moves one unit in the direction (North, East, South, or West, or not moving at all) that minimizes the expected shortest-path distance to the cat. In case of a tie, the dog favors not moving, then North, then East, then South, then West. (That is, the dog first considers whether to stay or go West; then, if the expected distance if it goes South is better or within 10^{-5}, the dog considers going South; then East; then North; then staying still.)

Neither the dog nor the cat can walk into a wall or on a diagonal, and the dog knows this. The dog has a perfect memory and knows the exact layout of the maze. The dog assumes that the cat starts at a uniform random location within the maze. The dog assumes that it could occupy the same space as the cat without realizing it.

Write a program that determines what path the dog takes given a sequence of observations.

Input

The first line of input file contains two integers X and Y ($3 \leqslant X, Y \leqslant 20$); the maze has dimension X units West to East by Y units North to South.

This is followed by a map of the maze. The first line corresponds to the northernmost strip of the maze, and the first column is the western side. A '`W`' indicates a wall, a '`.`' indicates open space, and a '`d`' marks the starting position of the dog. The sides of the maze are always walls, and the interior of the maze is connected. The map of the maze is followed by integer s ($1 \leqslant s \leqslant 1000$) on one line.

The remainder of the input is a series of k observations, one per second ($1 \leqslant k \leqslant 2000$). Each observation is written at separate line. A line that contains "`silence`" indicates that the dog hears nothing, and a line that contains two integers x_i and y_i ($0 \leqslant x_i, y_i \leqslant 20$), gives the coordinates — column, row where northwest corner of the maze is $(0, 0)$ in which the dog thinks it heard the cat. Note that the coordinates may be outside the maze.

Output

For each observation, output on a separate line one the of symbols '`N`', '`S`', '`E`', '`W`', or '`.`' if the dog goes North, South, East, West, or stays still, respectively, after this observation.

Example

dog.in	dog.out
20 20 WWWWWWWWWWWWWWWWWWWW WWWWWWWWWWWWWWWWWWWW WWWW.....d.....WWWWW WWWW.WWWWWWWWW.WWWWW WWWW.WWWWWWWWW.WWWWW WWWW.WWWWWWWWW.WWWWW WWWW.WWWWWWWWW.WWWWW WWWW.WWWWWWWWW.WWWWW WWWW...........WWWWW WWWW.WWWWWWWWW.WWWWW WWWW.WWWWWWWWW.WWWWW WWWW.WWWWWWWWW.WWWWW WWWW...........WWWWW WWWWWWWWW.WWWWWWWWWW WWWWWWWWW.WWWWWWWWWW WWWWWWWWW.WWWWWWWWWW WWWWWWWWW.WWWWWWWWWW WWWWWWWWWWWWWWWWWWWW WWWWWWWWWWWWWWWWWWWW WWWWWWWWWWWWWWWWWWWW 5 silence silence silence 10 8 9 8 8 9	E E E E E S

Problem D. Expression

Input file:	`expression.in`
Output file:	`expression.out`
Time limit:	2 seconds
Memory limit:	256 mebibytes

The most important activity of mobile operator ACM is the GSM network. As the mobile phone operator, ACM must build its own transmitting stations. It is very important to compute the exact behaviour of electro-magnetic waves. Unfortunately, prediction of electro-magnetic fields is a very complex task and the formulas describing them are very long and hard-to-read. For example, Maxwell 's Equations describing the basic laws of electrical engineering are really tough.

ACM has designed its own computer system that can make some field computations and produce results in the form of mathematic expressions. Unfortunately, by generating the expression in several steps, there are always some unneeded parentheses inside the expression. Your task is to take these partial results and make them "nice" by removing all unnecessary parentheses.

Input

There is a single positive integer T on the first line of input (equal to about 10000). It stands for the number of expressions to follow. Each expression consists of a single line containing only lowercase letters, operators ('`+`', '`-`;, '`*`, '`/`') and parentheses ('`(` and '`)`'). The letters are variables that can have any value, operators and parentheses have their usual meaning. Multiplication and division have higher priority then subtraction and addition. All operations with the same priority are computed from left to right (operators are left-associative). There are no spaces inside the expressions. No input line contains more than 250 characters.

Output

Print a single line for every expression. The line must contain the same expression with unneeded parentheses removed. You must remove as many parentheses as possible without changing the semantics of the expression. The semantics of the expression is considered the same if and only if any of the following conditions hold:

The ordering of operations remains the same. That means $(a+b)+c$ is the same as $a+b+c$, and $a+(b/c)$ is the same as $a+b/c$. The order of some operations is swapped but the result remains unchanged with respect to the addition and multiplication associativity. That means $a+(b+c)$ and $(a+b)+c$ are the same. We can also combine addition with subtraction and multiplication with division, if the subtraction or division is the second operation. For example, $a+(b-c)$ is the same as $a+b-c$. You cannot use any other laws, namely you cannot swap left and right operands and you cannot replace $a-(b-c)$ with $a-b+c$.

Examples

expression.in	expression.out

2
`x+(x+(x+x)-(x*x))/x`
`(x*x)/((x*x))+(x)`
	`x+(x+x+x-x*x)/x`
	`x*x/(x*x)+x`

Problem E. Paint

Input file:	`paint.in`
Output file:	`paint.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

You are given an undirected graph G without loops and multiple edges. You should paint the edges of G with white, red and blue colors, so that for each white edge there must be an adjacent red edge. Two edges are adjacent if they have a common vertex; it is guaranteed that any two edges have at most one common vertex.

Painting an edge in white is free, painting an edge in blue costs 1 rouble, painting an edge in red costs 2 roubles.

Find out the minimal amount of money you need to finish this work.

Input

The first line of the input file contains two integers n and m ($1 \leqslant n \leqslant 20$, $1 \leqslant m \leqslant 40$) — the number of vertices and edges respectively. Each of following m lines contains numbers of two endpoints of an edge — two integers a_i and b_i ($1 \leqslant a_i, b_i \leqslant n$).

Output

Print one integer — the minimal amount of money you need to paint all edges in appropriate way.

Examples

paint.in	paint.out

3 3
1 2
2 3
1 3
	2

18 27
1 2
1 4
1 6
3 2
3 4
3 6
2 5
4 5
7 8
7 10
7 12
9 8
9 10
9 12
8 11
10 11
13 14
13 16
13 18
15 14
15 16
15 18
14 17
16 17
5 11
12 17
18 6
	14

Problem F. Parliament

Input file:	`parliament.in`
Output file:	`parliament.out`
Time limit:	2 seconds
Memory limit:	256 Mebibytes

Byteland parliament speaker announced creation of a new faction of independent deputies. For each pair of independent deputies it is known whether they are friends, enemies or are neutral to each other (obviously, no two of these conditions can be true simultaneously).

Speaker's plan about faction's structure is the following:

- There must be no enemies among faction members;
- For each faction member, all his friends must be in the faction.

What is the maximal possible number of deputies that the speaker can gather into the new faction? What is the number of ways to gather a faction with that number of deputies?

Input

The first line of the input file contains three integers n, p and q ($2 \leqslant n \leqslant 250$, $\frac{n(n-1)}{3} \leqslant p \leqslant \frac{n(n-1)}{2}$, $0 \leqslant q \leqslant \frac{n(n-1)}{6}$) — the number of independent deputies, the number of friendly pairs and the number of enemy pairs. All the deputies are numbered from 1 to n. Each of the next p lines contains two integer numbers a_i and b_i ($1 \leqslant a_i, b_i \leqslant n$, $a_i \neq b_i$), meaning that deputies a_i and b_i are friends. Each of the next q lines contains pairs of enemies in the same format. It is guaranteed that no unordered pair appears more than once among the last $p + q$ lines.

Output

Output two integers — the maximal number of deputies that the speaker can gather into faction and also the number of ways to select those deputies.

Example

parliament.in	parliament.out

6 10 2
1 2
1 3
4 1
1 5
2 5
3 2
2 4
3 4
3 5
5 4
2 6
5 6
	5 1

Problem G. Sequences

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

You are given a sequence of integers A: $a_1, a_2, \ldots, a_n$.

Let us call a sequence of indices $c_1, c_2, \ldots, c_p$ where $1 \leqslant c_i \leqslant n$ *lowering* if for subsequence $a_{c_1}, a_{c_2}, \ldots, a_{c_p}$ the inequality $a_{c_1} > a_{c_2} > \ldots > a_{c_p}$ holds.

A «lowering» sequence of indices $c_1, c_2, \ldots, c_p$ is lexicographically smaller than another «lowering» sequence $d_1, d_2, \ldots, d_p$ if there exists $k \in [1, p]$ with the following properties: $c_i = d_i$ for all $i \in [1, k-1]$ and $c_k < d_k$. Using this rule one can sort lexicographically (in increasing order) all «lowering» sequences of fixed length p built upon given sequence A.

Your task is to find k-th «lowering» sequence in the sorted list. Numeration in list is 1-based.

Input

The first line of the input file contains three integer numbers n, p and q — the length of the sequence A, the length of the «lowering» sequences considered and the number of queries correspondingly ($1 \leqslant n, q \leqslant 10^5$, $1 \leqslant p \leqslant 10$). The second line contains n integers a_i ($-10^9 \leqslant a_i \leqslant 10^9$) — the elements of sequence A, listed in order of increasing their indices in sequence.

The next q lines contain queries: j-th line contains integer k_j ($1 \leqslant k_j \leqslant 10^{18}$) — the number of the «lowering» sequence you need to find.

Output

Output a lines. i-th line should contain the answer for the i-th query — k_i-th «lowering» sequence in the order of increasing indices (p integers from 1 to n) or -1 in case it doesn't exist.

Example

seq.in	seq.out

5 3 3
-1 6 5 2 1
1
5
3
	2 3 4
	-1
	2 4 5

Problem H. Squares

Input file:	`squares.in`
Output file:	`squares.out`
Time limit:	9 seconds
Memory limit:	**64 Mebibytes**

There are n points marked on the plane. How many different squares are there with sides parallel to the coordinate axes with all four vertices marked (two squares are considered different if the sets of their vertices are different)?

Input

The first line of the input file contains one integer n ($1 \leqslant n \leqslant 10^5$) — the number of marked points. Each of the next n lines contains one of the marked points. Each point is given by its coordinates — pair of integers x_i and y_i ($-10^6 \leqslant x_i, y_i \leqslant 10^6$). None of the points coincide.

Output

Output one integer — the number of squares.

Example

squares.in	squares.out

6
0 0
0 1
1 0
1 1
3 0
3 1
	1

Problem I. Tables

Input file:	`tables.in`
Output file:	`tables.out`
Time limit:	8 seconds
Memory limit:	256 Mebibytes

The Individual Programming Championship of Byteland consists of k rounds. Each of the n participants takes part in each round. After a round, participants are ranked 1 through n based on their relative performance in that round (ties are considered impossible). Thus the results table of each round can be viewed as a permutation of numbers 1, 2, . . ., n: first goes the winner's number, then the number of second place finisher, and so on.

Let the *distance* between results tables p_1, p_2, . . ., p_n and q_1, q_2, . . ., q_n be the sum

$$D(p,q) = \sum_{i=1}^{n} \min(|p'_i - q'_i|, 8)$$

where p' and q' are inverse permutations of p and q, respectively: p'_1 is the place of first participant in results table p, p'_2 is the place of second participant in p, and so on; the same goes for q'.

The Championship sponsors proposed to make the final results table r_1, r_2, . . ., r_n in such a way that the sum of k distances between r and the results of each round is the least possible. Your task is to calculate that sum.

Input

The first line of input contains two integers n and k, the number of participants and the number of rounds ($2 \leqslant n \leqslant 5000$, $2 \leqslant k \leqslant 3$). Each of the next k lines contains the results of that particular round and consists of n integers which form a permutation of numbers 1, 2, . . ., n. The first element of the permutation is the number of the first place, the second one is the number of the second place, and so on.

Output

Output one integer: the minimal possible sum of distances from the final results table to the results tables of each of the k rounds.

Example

tables.in	tables.out

`5 2`
`5 2 4 3 1`
`2 4 1 3 5`
	`8`

Problem J. Triangle

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

Statement of this problem is very short. Given positive integers A, B and C, find the number of points on the plane with non-negative integer coordinates x and y such that $Ax + By \leqslant C$.

Input

First line of input file contains three integers A, B and C ($1 \leqslant A, B \leqslant 10^9$, $1 \leqslant C \leqslant \min(A, B) \cdot 10^9$).

Output

Print the required number of points.

Example

triangle.in	triangle.out
3 4 13	12