Problem A. Algorith For Robot

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

Given the stripe of length $2n + 1$, the cells are enumerated by sequential integers from 1 to $2n + 1$ from left to right.

Each cell initially contains at most one token.

The robot starts at the central cell of the stripe (i.e. the cell $n+1$). The robot is controlled by the algorithm that consists exactly of n operations and is encoded in the memory as the string $w = w_0w_1...w_{n-1}$ consisting of the letters 'R', 'L', 'T':

- 'R' — move right;
- 'L' — move left;
- 'T' — collect the token if the current cell contains the token that was not collected before; otherwise do nothing.

You are given the program, represented by string w. You can replace the string with any of its cyclic shifts (or keep the string unchanged) and start the robot. Calculate the maximum number of tokens the robot can collect if you choose the cyclic shift optimally.

Input

The first line of the input contains one integer n ($1 \le n \le 3 \cdot 10^4$).

The second line contains the binary string s of length $2 \cdot n + 1$. $s_i = 1$ if i-th cell contains the token, and 0 otherwise.

The third line contains the program w — the string of length n, consisting of letters 'R', 'L', and 'T'

Output

Print one integer — the maximum number of tokens the robot can collect if you replace the program w with its cyclic shift.

Examples

standard input	standard output
3 0011000 TTL	2
4 000010000 TLRT	1

Note

In the sample 1, the string "TLT' allows to collect all the tokens.

Problem B. Build The Tree

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

Consider a tree on n vertices. Each pair of vertices is connected by a simple path. Your task is to construct a tree on n vertices such that the total length of paths between all unordered pairs of vertices is exactly m. The length of a path is the number of its edges.

Input

First line of the input contains two integers n and m ($2 \le n \le 20$, $1 \le m \le 10000$).

Output

If it is impossible to construct such a tree, printf "NO".

Otherwise, print "YES" on the first line. Each of next $n-1$ lines must describe one edge and contain two integers between 1 and n — numbers of vertices connected by the corresponding edge.

If more than one correct tree exists, you may print any of them.

Examples

standard input	standard output
3 4	YES 1 2 2 3
3 5	NO

Problem C. Create the Polygon

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

Construct a convex polygon with the following properties (ot determine that the construction is impossible):

- Coordinates of all vertices of the polygon are integers.
- All coordinates satisfy the inequalities $0 \le x_i \le n$, $0 \le y_i \le m$.
- Number of vertices should not exceed 10^5.
- No three vertices of the polygon lie on a common straight line.
- Area of the polygon is equal to S.

Input

The only line of input contains two positive integers n and m ($1 \le n, m \le 10^5$) — upper bounds on coordinates, and one real number S ($0 < S \le 10^9$) — desired area of the polygon given with at most 6 digits after the decimal point.

Output

If it is impossible to construct a suitable polygon, print -1.

Otherwise, on the first line print k — number of vertices in your polygon ($3 \le k \le 10^5$). On each of the next k lines print x_i and y_i — coordinates of the vertices of the polygon ($0 \le x_i \le n$, $0 \le y_i \le m$) listed in clockwise order.

Examples

standard input	standard output
10 10 81	5 0 0 0 7 9 9 10 9 10 0
1 2 0.5	3 0 0 0 1 1 1

Problem D. Dodge The Bullets

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.5 seconds
Memory limit:	1024 mebibytes

You are playing an oldschool arcade game. The point of the game is to dodge enemy's bullets.

The screen is n cells wide and 10^6 cells high. The cells are indexed by their column index x and row index y; indices are 1-based. You control a spaceship in shape of a rectangle 1 cell wide and H cell high. Your spaceship's bottom edge is aligned with the bottom edge of the screen, that is, your ship always occupies cells $(x, 1)$, $(x, 2)$, $\ldots$, (x, H) for your current x.

All l bullets that you have to dodge are already on the screen, i-th of them is located at the center of the cell (x_i, y_i). The game consists of m rounds, each round proceeds as follows:

- First, you can move your ship one cell to the left or to the right; you are also free to stay still. You cannot move outside the screen.
- Next, all bullets move down one cell, except for the bullets in the bottom row that instead disappear.

If at any point any cell of your ship contains a bullet, you immediately lose. If you stay alive after m-th round concludes, you win.

You have played k games. In i-th of these games the starting column of your ship was a_i. You are now wondering, how many of these games you could have won should you play optimally?

Input

The first line of input contains five integers n, m, k, l, and h ($1 \le n, k, l, h \le 10^6$, $1 \le m \le 10^3$).

The second line contains k space-separated integers a_i — initial positions of the spaceships in each of your games.

Next l lines describe positions of bullets. Each of these lines contains two integers x_i and y_i — column and row index of the respective bullet ($1 \le x_i \le n$, $H + 1 \le y_i \le 10^6$).

Output

Print one integer — the answer to the problem.

Examples

standard input	standard output
3 3 2 2 2 1 2 1 3 2 3	1
6 10 7 7 3 1 2 2 3 4 5 6 6 10 2 4 2 6 3 5 4 4 5 6 6 10	6

Problem E. Encoded Permutations

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

Let's encode the permutation p of the integers $1, \ldots, n$ with property that there are no i such as $p_i = i$, with the sequence, consisting of '+' and '-' in the following way: $s_i = +$ if $p_i > i$, and $s_i = -$, if $p_i < i$,.

Given the sequence of pluses and minuses. Calculate the number of permutations that may be encoded of that sequence, modulo $10^9 + 7$.

Input

The first line of the input contains one string s ($1 \le |s| \le 5000$), consisting of the characters '+' and '-' — the given sequence.

Output

Print one integer — the number of the encoded permutations modulo $10^9 + 7$.

Examples

standard input	standard output
`+-+-`	2
`+-+-+`	0
`+-+-+---`	200

Problem F. Font Processor

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

Byteazar develops his own font processor. As any font processor, it supports work with the *spline curves*.

Each spline consists of several parts, linked one by one. Each part is given by a parametric equation of the following form:

$$X_i(t) = AX_i \cdot t^3 + BX_i \cdot t^2 + CX_i \cdot t + DX_i$$

$$Y_i(t) = AY_i \cdot t^3 + BY_i \cdot t^2 + CY_i \cdot t + DY_i,$$

where t – a real number, continuously changing from 0 to 1, inclusively.

One of active tasks is to check if the points lies inside the closed spline curve. Byteazar asked you to help him with that task.

Input

The first line of the input contains an integer N ($1 \le N \le 1000$) — the number of parts of the spline. Then N lines, each containg 8 real numbers AX_i, BX_i, CX_i, DX_i, AY_i, BY_i, CY_i, DY_i follow.

Then line with one integer M ($1 \le M \le 10^5$) follows — the number of points. Each of the followinfg two lines contain two real numbers X_j and Y_j — the coordinate of a point to be checked.

Within the specified accuracy, the spline is continuous and does not have self-intersections. The distance from each point to the boundary of the spline is not less than 10^{-5}. The spline lies entirely within the rectangular area $[-10, 10] \times [-10, 10]$.

Output

For each query print 1, if the point lies inside the contour and 0, if it lies outside.

Examples

standard input	standard output
6 3.53200 -4.87400 1.66600 -0.46000 0.67600 -0.70800 0.26600 -0.20200 1.74000 -2.86600 0.84800 -0.13600 1.29000 -1.88200 0.61200 0.03200 -1.40400 2.10600 -0.51200 -0.41400 -0.17200 0.25800 0.10600 0.05200 -1.40000 1.57000 0.00000 -0.22400 -0.22200 0.01600 -0.00000 0.24400 -3.42400 4.60600 -1.06000 -0.05400 0.11600 0.22800 -0.63400 0.03800 0.60200 -0.07000 -1.06000 0.06800 1.77400 -2.52800 0.80400 -0.25200 8 -0.23000 -0.27000 -0.37094 -0.10086 0.00000 0.15040 -0.39010 0.20360 -0.31067 0.04029 -0.20000 -0.08960 0.03000 -0.05071 -0.18603 0.17014	0 1 1 0 1 1 0 1

Problem G. Guess The Bit String

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

This is an interactive problem.

The jury program is generated a secret **positive** integer p and represented it as the bit string S of length N (possibly with the leading zeroes).

Your solution can do a queries of the form "? M X Y". Then the jury program take the bit string A of length M, starting with position X and the bit string B of length M, starting with position Y and compare strings A and B (from left to right) and strings A' and B' (same strings from right to left) lexicographically. As the answer, you will obtain two results of both comparsions. Bits are enumerated from **right to left**, starting from 0.

You must determine the string S by asking no more than $\lceil \frac{N+1}{2} \rceil$ queries.

Interaction Protocol

First your program reads one integer N — the number of the bits in X ($1 \le N \le 60$).

To make a query, print "? M L R", where M is the number of bits to compare, L and R – positions to compare ($1 \le M \le N$, $0 \le L, R \le N - M$). If you make a query outside those restrictions, the solution will be judged as wrong one.

The jury program answers each query with string of length 2 – results of left-to-right and right-to-left comparisons. Possible comparison results are: '<', '>' and '='.

To print the guessed X, use the format "! X".

Every query and final output must be a single line ending with single end-of-line. Output buffers must be flushed after every line.

Examples

standard input	standard output
5 >< << == 	 ? 3 0 2 ? 1 1 3 ? 2 3 0 ! 01101
3 == 	 ? 2 0 1 ! 111

Note

In the first sample secret number is 01101. First query compares 101 with 011 and 101 with 110, so the answer is "><.

Problem H. HR Codes

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

The startup, where Jean works, is planning to create the system to replace QR codes with the new improved system that they call HR (human-readable) -codes.

They decided to represent binary data as a string of digit patterns for 0 and 1, displayed on a picture below. Picture also contains a representation of a binary string "1001".

```
.#       ###            .#.###.###..#
##       #.#            ##.#.#.#.#.##
.#       ###            .#.###.###..#

 1        0                 1001
```

Digits in the code are separated by one column of '.' characters, and there are no empty columns at beginning nor at end of code (i.e. the code starts with the first column of the first letter and ends with the last column of the last letter).

You are given the image from camera. Let's define an *recognition error* as the number of different pixels in the image and in the code. Determine the minimal recognition error which can be obtained by choosing the HR-code most closely resembling the given image.

Input

The first line of the input contains one integer n ($5 \le n \le 10^5$) – the width of the image. Three following lines contain the image. Each character is either '#' or '.'.

Output

Output single integer – minimum possible recognition error.

Examples

standard input	standard output
5 .#..# ##.## .#..#	0
14 .#.#.#.##.#..# ##.#.#.##.#.## .#####.####..#	14

Problem I. Integer Function

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

Given three integers a, b, p, where p is prime and function $f(x, y)$ working in the following way:

- First, the integer $v = ab + xy$ is calculated.
- If v is divisible by p, then $f(x, y) = 1$.
- Otherwise, $u = ax + by$ is calculated; the resulting value of $f(x, y)$ will be equal to $u \cdot inv_p(v)$ modulo p.

Here $inv_p(a)$ denotes the integer $b \in [0, \ldots, p-1]$, such as $ab - 1$ is divisible by p.

Given integer k, calculate the value

$$f(f(\ldots f(f(k, k-1), k-2), \ldots, 2), 1)$$

Input

The first line of the input contains four integers a, b, p, k ($2 \le p \le 10^9 + 7$, $1 \le a, b \le min(p-1, 10^5)$, $2 \le k \le 10^{18}$, p is prime), the parameters of function and the number of the iterations, respectively.

Output

Print one integer — the calculated value.

Examples

standard input	standard output
2 4 5 3	4
1 1 2 10	1

Problem J. Jogging

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

n runners are running on the stadium, enumerated by the sequential integers from 1 to n, each runner runs with the speed 1.

The stadium can be represented as two parallel tracks of the length l. One track is used for running left, another — for the running right. When the runners reach the end of the stadium, he immediately changes the direction and the track.

One day, the coach Bytica arrived at the stadium and wrote for each runner the distance from the left end of the stadium and the running direction. Some times after, Byteasar, the Bytica's assistent, arrived at the stadium and noticed that the runners are in the distinct points and that they form the permutation p, counting from left to right.

Check if it is possible that both Bytica and Byteasar did not mistaken.

Input

The first line of the input contains two integers n and l ($1 \le n \le 10^5$, $1 \le l \le 10^9$) — the number of the runners and the length of the stadium, respectively.

Each of the following n lines contains two integers x_i and v_i ($0 \le x_i \le l$, $v_i \in \{-1, 1\}$) — the distance from i-th runner to left end of the stadium and the running direction, written by Bytica.

The last line contains the permutation p of length n, that was written by Byteasar.

You may assume that all pairs $\{x_i, v_i\}$ are pairwise distinct, and that if $x_i = 0$ then $v_i = 1$, and if $x_i = l$, then $v_i = -1$.

Output

If it is possible that both Byteasar and Bytica did not mistaken, print "Yes". Otherwise print "No".

Examples

standard input	standard output
2 10 5 1 5 -1 1 2	Yes
4 4 0 1 1 -1 2 1 3 -1 2 1 4 3	Yes
4 8 8 -1 7 -1 0 1 2 1 1 3 4 2	No

Problem K. King And Tree

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

At the King's Garden the rooted tree is planted. The root vertex have the number 1. Some vertices of the tree contain the apples.

The King may take the set S (possibly empty) of the positive integers, does not exceeding the maximal distance from the root to any vertex. Then for all vertices such as the distance from root to that vertex is in the set, he may delete the edges connecting them with their direct parent. Then we have several trees. If all those trees have the equal number of vertices with the apples, the set S is called *nice*.

Count the number of the nice sets modulo $10^9 + 7$.

Input

The first line of the input contains two integers n and k ($1 \le k \le n \le 10^5$) — the number of the vertices in the tree and the number of the vertices with the apples, respectively.

The second line contains k pairwise distinct integers a_i ($1 \le a_i \le n$) — numbers of the vertices with the apples.

Then $n - 1$ line follow. i-th of those lines contains two integers u_i and v_i ($1 \le u_i, v_i, \le n$) — numbers of the vertices connected by i-th edge. It is guaranteed that the given graph is a tree.

Output

Print one integer — the number of the nice sets modulo $10^9 + 7$.

Examples

standard input	standard output
7 3 2 5 6 1 2 2 3 3 4 4 5 5 6 6 7	4
8 6 1 2 5 6 7 8 1 2 2 3 2 4 3 5 3 6 4 7 4 8	3

Problem L. List of Palindromes

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.5 seconds
Memory limit:	1024 mebibytes

Given a string S. Let's build the lexicographically ordered list of all different palindromes that can be obtained by deleting and/or rearranging characters in S.

Your task is to remove from this list the palindromes that are lexicographically less than the given string T. Additionally, if after removal the list is still longer than n, the extra palindromes from the end should also be removed.

Input

The first two lines of the input contain strings S and T, consisting of digits and lowercase English letters ($1 \le |T| \le |S| \le 100$). The last line contains one integer n — the length limit of the list ($1 \le n \le 2 \cdot 10^4$).

Output

Print all palindromes that are remaining in the list, in lexicographic order. If the list is empty, print an empty line.

Examples

standard input	standard output
abcbac aca 15	aca acbbca acbca acca b baab bab bacab baccab bb bcaacb bcacb bcb bccb c
123243 21a 50	22 23132 232 2332 23432 242 3 313 32123 3223 323 32423 33 343 4

Problem M. Matrix

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

Given the matrix consisting of N rows and M columns, the cell in i-th row and j-th column contains the integer $X_{i,j}$ such that $1 \le X[i][j] \le C$, where C is the given integer.

Consider the integer arrays A of size N and B of size M *parent arrays* for the given matrix, if the integers in those arrays are positive and do not exceed C, and if for any $1 \le i \le N$ and $1 \le j \le M$ the condition $X[i][j] = min(A[i], B[j])$ is held.

Your task is to calculate number of the distinct pairs of the arrays A and B that form the parent arrays for the given matrix modulo $10^9 + 7$.

Input

The first line of the input contains three integers N, M and C ($1 \le N \cdot M \le 10^5$, $1 \le C \le 10^9$) — the dimensions of the matrix and the maximal value of the integer in the matrix.

Then N lines follow, i-th of them containg M integers $X[i][j]$ ($1 \le X[i][j] \le C$).

Output

Print the answer of the number of the distinct pairs of the parent arrays A and B modulo $10^9 + 7$.

Examples

standard input	standard output
1 1 3 2	3
2 2 4 1 1 1 3	3

Note

Consider the first sample. In this sample both arrays shall have the length 1 and integers in arrays cannot be greater than 3. Because $min(A[1], B[1]) = 2$, the following pairs of the arrays form the pair of parent arrays

- $A[1] = 2$, $B[1] = 2$
- $A[1] = 2$, $B[1] = 3$
- $A[1] = 3$, $B[1] = 2$

It is easy to see that in all other arrays of length 1 consisting of positive integers not exceeding 3 the condition $min(A[1], B[1]) = 2$ is not held, so the answer is equal to 3.