Codeforces Round #467 (Div. 1)

A. Save Energy!

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Julia is going to cook a chicken in the kitchen of her dormitory. To save energy, the stove in the kitchen automatically turns off after k minutes after turning on.

During cooking, Julia goes to the kitchen every d minutes and turns on the stove if it is turned off. While the cooker is turned off, it stays warm. The stove switches on and off instantly.

It is known that the chicken needs t minutes to be cooked on the stove, if it is turned on, and $2t$ minutes, if it is turned off. You need to find out, how much time will Julia have to cook the chicken, if it is considered that the chicken is cooked evenly, with constant speed when the stove is turned on and at a constant speed when it is turned off.

Input
The single line contains three integers k, d and t ($1 \le k, d, t \le 10^{18}$).

Output
Print a single number, the total time of cooking in minutes. The relative or absolute error must not exceed 10^{-9}.

Namely, let's assume that your answer is x and the answer of the jury is y. The checker program will consider your answer correct if $\frac{|x-y|}{max(1,y)} \le 10^{-9}$.

Examples

input
3 2 6
output
6.5

input
4 2 20
output
20.0

Note
In the first example, the chicken will be cooked for 3 minutes on the turned on stove, after this it will be cooked for $\frac{3}{6}$. Then the chicken will be cooked for one minute on a turned off stove, it will be cooked for $\frac{1}{12}$. Thus, after four minutes the chicken will be cooked for $\frac{3}{6} + \frac{1}{12} = \frac{7}{12}$. Before the fifth minute Julia will turn on the stove and after 2.5 minutes the chicken will be ready $\frac{7}{12} + \frac{2.5}{6} = 1$.

In the second example, when the stove is turned off, Julia will immediately turn it on, so the stove will always be turned on and the chicken will be cooked in 20 minutes.

B. Sleepy Game

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Petya and Vasya arranged a game. The game runs by the following rules. Players have a directed graph consisting of n vertices and m edges. One of the vertices contains a chip. Initially the chip is located at vertex s. Players take turns moving the chip along some edge of the graph. Petya goes first. Player who can't move the chip loses. If the game lasts for 10^6 turns the draw is announced.

Vasya was performing big laboratory work in "Spelling and parts of speech" at night before the game, so he fell asleep at the very beginning of the game. Petya decided to take the advantage of this situation and make both Petya's and Vasya's moves.

Your task is to help Petya find out if he can win the game or at least draw a tie.

Input
The first line of input contain two integers n and m — the number of vertices and the number of edges in the graph ($2 \le n \le 10^5$, $0 \le m \le 2 \cdot 10^5$).

The next n lines contain the information about edges of the graph. i-th line ($1 \le i \le n$) contains nonnegative integer c_i — number of vertices such that there is an edge from i to these vertices and c_i distinct integers $a_{i,j}$ — indices of these vertices ($1 \le a_{i,j} \le n$, $a_{i,j} \ne i$).

It is guaranteed that the total sum of c_i equals to m.

The next line contains index of vertex s — the initial position of the chip ($1 \le s \le n$).

Output

If Petya can win print «`Win`» in the first line. In the next line print numbers $v_1, v_2, \ldots, v_k$ ($1 \le k \le 10^6$) — the sequence of vertices Petya should visit for the winning. Vertex v_1 should coincide with s. For $i = 1 \ldots k - 1$ there should be an edge from v_i to v_{i+1} in the graph. There must be no possible move from vertex v_k. The sequence should be such that Petya wins the game.

If Petya can't win but can draw a tie, print «`Draw`» in the only line. Otherwise print «`Lose`».

Examples

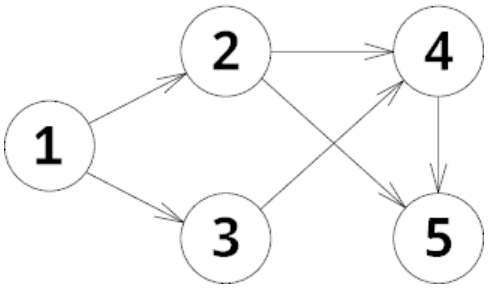
input
5 6 2 2 3 2 4 5 1 4 1 5 0 1
output
Win 1 2 4 5

input
3 2 1 3 1 1 0 2
output
Lose

input
2 2 1 2 1 1 1
output
Draw

Note

In the first example the graph is the following:

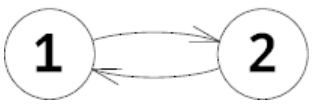


Initially the chip is located at vertex 1. In the first move Petya moves the chip to vertex 2, after that he moves it to vertex 4 for Vasya. After that he moves to vertex 5. Now it is Vasya's turn and there is no possible move, so Petya wins.

In the second example the graph is the following:

Initially the chip is located at vertex 2. The only possible Petya's move is to go to vertex 1. After that he has to go to 3 for Vasya. Now it's Petya's turn but he has no possible move, so Petya loses.

In the third example the graph is the following:



Petya can't win, but he can move along the cycle, so the players will draw a tie.

C. Lock Puzzle

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Welcome to another task about breaking the code lock! Explorers Whitfield and Martin came across an unusual safe, inside of which, according to rumors, there are untold riches, among which one can find the solution of the problem of discrete logarithm!

Of course, there is a code lock is installed on the safe. The lock has a screen that displays a string of n lowercase Latin letters. Initially, the screen displays string s. Whitfield and Martin found out that the safe will open when string t will be displayed on the screen.

The string on the screen can be changed using the operation «`shift` x». In order to apply this operation, explorers choose an integer x from 0 to n inclusive. After that, the current string $p = \alpha\beta$ changes to $\beta^R\alpha$, where the length of β is x, and the length of α is $n - x$. In other words, the suffix of the length x of string p is reversed and moved to the beginning of the string. For example, after the operation «`shift` 4» the string «`abcacb`» will be changed with string «`bcacab` », since $\alpha =$ `ab`, $\beta =$ `cacb`, $\beta^R =$ `bcac`.

Explorers are afraid that if they apply too many operations «`shift`», the lock will be locked forever. They ask you to find a way to get the string t on the screen, using no more than 6100 operations.

Input

The first line contains an integer n, the length of the strings s and t ($1 \le n \le 2\,000$).

After that, there are two strings s and t, consisting of n lowercase Latin letters each.

Output

If it is impossible to get string t from string s using no more than 6100 operations «`shift`», print a single number -1.

Otherwise, in the first line output the number of operations k ($0 \le k \le 6100$). In the next line output k numbers x_i corresponding to the operations «`shift` x_i» ($0 \le x_i \le n$) in the order in which they should be applied.

Examples

input
6 abacbb babcba

output
4 6 3 2 3

input
3 aba bba

output
-1

In the first example, after applying the operations, the string on the screen will change as follows:

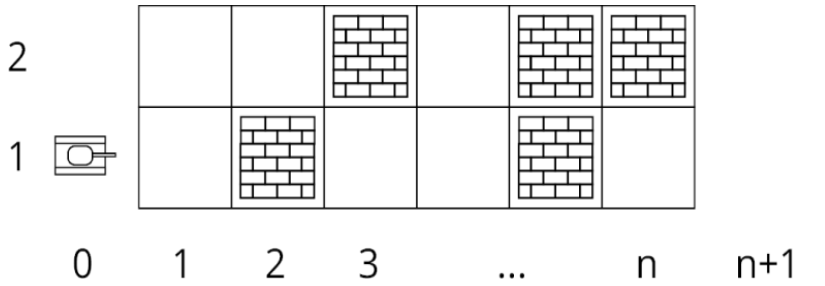
1. abacbb ⟶ bbcaba
2. bbcaba ⟶ ababbc
3. ababbc ⟶ cbabab
4. cbabab ⟶ babcba

D. World of Tank

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vitya loves programming and problem solving, but sometimes, to distract himself a little, he plays computer games. Once he found a new interesting game about tanks, and he liked it so much that he went through almost all levels in one day. Remained only the last level, which was too tricky. Then Vitya remembered that he is a programmer, and wrote a program that helped him to pass this difficult level. Try do the same.

The game is organized as follows. There is a long road, two cells wide and n cells long. Some cells have obstacles. You control a tank that occupies one cell. Initially, the tank is located before the start of the road, in a cell with coordinates $(0, 1)$. Your task is to move the tank to the end of the road, to the cell $(n + 1, 1)$ or $(n + 1, 2)$.



Every second the tank moves one cell to the right: the coordinate x is increased by one. When you press the up or down arrow keys, the tank instantly changes the lane, that is, the y coordinate. When you press the spacebar, the tank shoots, and the nearest obstacle along the lane in which the tank rides is instantly destroyed. In order to load a gun, the tank needs t seconds. Initially, the gun is not loaded, that means, the first shot can be made only after t seconds after the tank starts to move.

If at some point the tank is in the same cell with an obstacle not yet destroyed, it burns out. If you press the arrow exactly at the moment when the tank moves forward, the tank will first move forward, and then change the lane, so it will not be possible to move diagonally.

Your task is to find out whether it is possible to pass the level, and if possible, to find the order of actions the player need to make.

Input

The first line contains four integers n, m_1, m_2 and t, the length of the field, the number of obstacles in the first lane, the number of obstacles in the second lane and the number of tank steps before reloading, respectively ($1 \le n \le 10^9$; $0 \le m_1, m_2 \le n$; $0 \le m_1 + m_2 \le 10^6$; $1 \le t \le n$).

The next two lines contain a description of the obstacles. The first of these lines contains m_1 numbers x_i — the obstacle coordinates in the first lane ($1 \le x_i \le n$; $x_i < x_{i+1}$). The y coordinate for all these obstacles will be 1.

The second line contains m_2 numbers describing the obstacles of the second lane in the same format. The y coordinate of all these obstacles will be 2.

Output

In the first line print «`Yes`», if it is possible to pass the level, or «`No`», otherwise.

If it is possible, then in the second line print the number of times the tank moves from one lane to another, and in the next line print the coordinates of the transitions, one number per transition: the coordinate x ($0 \le x \le n + 1$). All transition coordinates coordinates must be distinct and should be output in strictly increasing order.The number of transitions should not exceed $2 \cdot 10^6$. If the tank can pass the level, then it can do it using no more than $2 \cdot 10^6$ transitions.

In the fourth line print the number of shots that the tank makes during the movement, in the following lines print two numbers, x and y coordinates of the point ($1 \le x \le n$, $1 \le y \le 2$), from which the tank fired a shot, the number of shots must not exceed $m_1 + m_2$. Shots must be output in the order in which they are fired.

If there are several solutions, output any one.

Examples

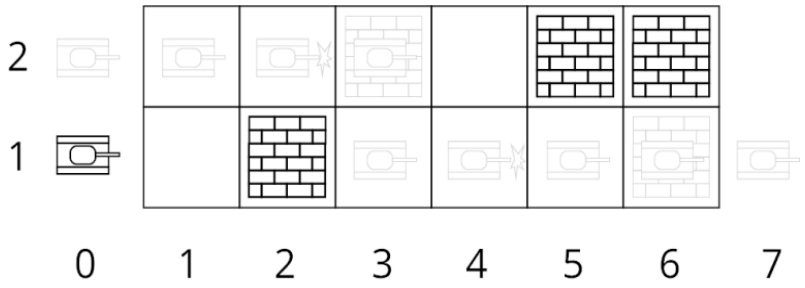
input
6 2 3 2 2 6 3 5 6
output
Yes 2 0 3 2 2 2 4 1

input
1 1 1 1 1 1
output
No

input
9 5 2 5 1 2 7 8 9 4 6
output
Yes 4 0 3 5 10 1

```
5 2
```

Note
Picture for the first sample test.



E. Iqea

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Gridland is placed on infinite grid and has a shape of figure consisting of cells. Every cell of Gridland is a city. Two cities that are placed in adjacent cells are connected by the road of length 1. It's possible to get from any city to any other city using roads. The distance between two cities is the minimum total road length on path from one city to another. It's possible to get from any cell that doesn't belong to Gridland to any other cell that doesn't belong to Gridland by using only cells which don't belong to Gridland. In other words, Gridland is connected and complement of Gridland is also connected.

At the moment no city in Gridland has Iqea famous shop. But Iqea has great plans for building shops in Gridland. For customers' convenience Iqea decided to develop an application. Using this application everyone can learn the distance to the nearest Iqea. You are to develop this application.

You are asked to process two types of queries:

- new Iqea shop has been opened in the city with coordinates (x, y);
- customer wants to know the distance to the nearest already opened Iqea shop from his city located in a cell with coordinates (x, y).

Pay attention that customer can move only by roads and can never leave Gridland on his way to the shop.

Input
The first line contains one integer n — number of cities in Gridland ($1 \le n \le 300\,000$). The following n lines contain two integers x_i and y_i — coordinates of cell that contains i-th city ($1 \le x_i, y_i \le 300\,000$). No two cells coincide. Cells form connected area, complement of this area is also connected.

The following line contains single integer q — number of queries ($0 \le q \le 300\,000$). Following q lines contain queries, i-th line consists of three integers t_i, x_i and y_i ($t_i \in \{1, 2\}$, $1 \le x_i, y_i \le 300\,000$). If $t_i = 1$, new Iqea shop has been opened in the city with coordinates (x_i, y_i). It's guaranteed that there was no Iqea shop in this city before. If $t_i = 2$, you are to find the distance to the nearest already opened Iqea shop from the city with coordinates (x_i, y_i). It's guaranteed that in queries of both types cell (x_i, y_i) belongs to Gridland.

Output
For every query of second type output single integer — minimum distance to the nearest Iqea shop. If no Iqea shop has been already opened, output -1.

Examples

input
7 1 2 1 3 2 3 3 1 3 2 3 3 4 2 5 2 3 2 1 4 2 2 1 2 1 3 3 2 2 3
output
-1 5 1

input

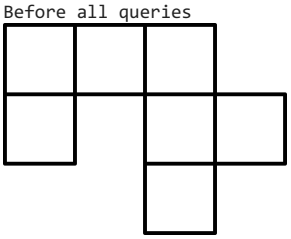
```
6
1 1
1 2
1 3
2 1
2 2
3 1
4
1 3 1
2 1 2
1 2 2
2 1 3
```

output

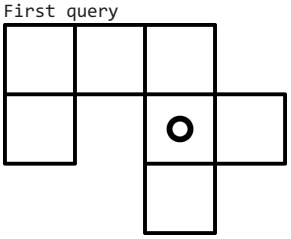
```
3
2
```

Explanation of the first example:

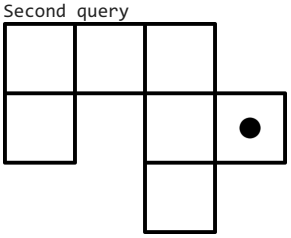
Before all queries



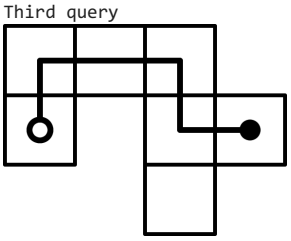
First query



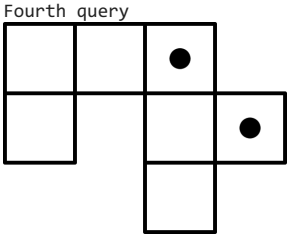
Second query



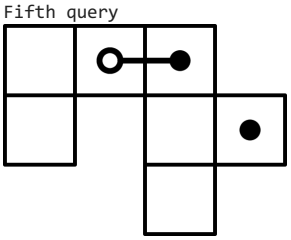
Third query



Fourth query

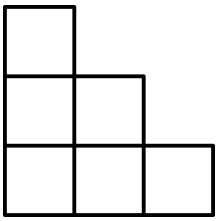


Fifth query

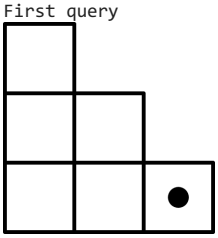


Explanation of the second example:

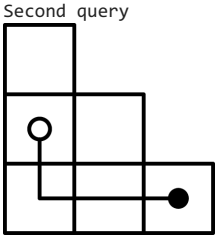
Before all queries



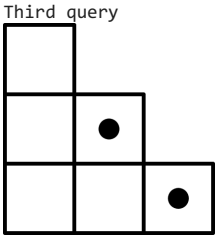
First query



Second query



Third query



Fourth query