Технокубок 2017 - Финал (только для онсайт-финалистов)

A. Andryusha and Socks

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

Andryusha is an orderly boy and likes to keep things in their place.

Today he faced a problem to put his socks in the wardrobe. He has n distinct pairs of socks which are initially in a bag. The pairs are numbered from 1 to n. Andryusha wants to put paired socks together and put them in the wardrobe. He takes the socks one by one from the bag, and for each sock he looks whether the pair of this sock has been already took out of the bag, or not. If not (that means the pair of this sock is still in the bag), he puts the current socks on the table in front of him. Otherwise, he puts both socks from the pair to the wardrobe.

Andryusha remembers the order in which he took the socks from the bag. Can you tell him what is the maximum number of socks that were on the table at the same time?

Input

The first line contains the single integer n $(1 \le n \le 10^5)$ — the number of sock pairs.

The second line contains $2n$ integers $x_1, x_2, ..., x_{2n}$ $(1 \le x_i \le n)$, which describe the order in which Andryusha took the socks from the bag. More precisely, x_i means that the i-th sock Andryusha took out was from pair x_i.

It is guaranteed that Andryusha took exactly two socks of each pair.

Output

Print single integer — the maximum number of socks that were on the table at the same time.

Examples

input

```
1
1 1
```

output

```
1
```

input

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

output

```
2
```

Note

In the first example Andryusha took a sock from the first pair and put it on the table. Then he took the next sock which is from the first pair as well, so he immediately puts both socks to the wardrobe. Thus, at most one sock was on the table at the same time.

In the second example Andryusha behaved as follows:

- Initially the table was empty, he took out a sock from pair 2 and put it on the table.
- Sock (2) was on the table. Andryusha took out a sock from pair 1 and put it on the table.
- Socks $(1, 2)$ were on the table. Andryusha took out a sock from pair 1, and put this pair into the wardrobe.
- Sock (2) was on the table. Andryusha took out a sock from pair 3 and put it on the table.
- Socks $(2, 3)$ were on the table. Andryusha took out a sock from pair 2, and put this pair into the wardrobe.
- Sock (3) was on the table. Andryusha took out a sock from pair 3 and put this pair into the wardrobe.

Thus, at most two socks were on the table at the same time.

B. The Meeting Place Cannot Be Changed

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The main road in Bytecity is a straight line from south to north. Conveniently, there are coordinates measured in meters from the southernmost building in north direction.

At some points on the road there are n friends, and i-th of them is standing at the point x_i meters and can move with any speed no greater than v_i meters per second in any of the two directions along the road: south or north.

You are to compute the minimum time needed to gather all the n friends at some point on the road. Note that the point they meet at doesn't need to have integer coordinate.

Input

The first line contains single integer n $(2 \le n \le 60\,000)$ — the number of friends.

The second line contains n integers $x_1, x_2, ..., x_n$ $(1 \le x_i \le 10^9)$ — the current coordinates of the friends, in meters.

The third line contains n integers $v_1, v_2, ..., v_n$ $(1 \le v_i \le 10^9)$ — the maximum speeds of the friends, in meters per second.

Output

Print the minimum time (in seconds) needed for all the n friends to meet at some point on the road.

Your answer will be considered correct, if its absolute or relative error isn't greater than 10^{-6}. Formally, let your answer be a, while jury's answer be b. Your answer will be considered correct if holds.

Examples

input

```
3
7 1 3
1 2 1
```

output

```
2.000000000000
```

input

```
4
5 10 3 2
2 3 2 4
```

output

```
1.400000000000
```

Note

In the first sample, all friends can gather at the point 5 within 2 seconds. In order to achieve this, the first friend should go south all the time at his maximum speed, while the second and the third friends should go north at their maximum speeds.

C. Andryusha and Colored Balloons

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

Andryusha goes through a park each day. The squares and paths between them look boring to Andryusha, so he decided to decorate them.

The park consists of n squares connected with $(n - 1)$ bidirectional paths in such a way that any square is reachable from any other using these paths. Andryusha decided to hang a colored balloon at each of the squares. The baloons' colors are described by positive integers, starting from 1. In order to make the park varicolored, Andryusha wants to choose the colors in a special way. More precisely, he wants to use such colors that if a, b and c are distinct squares that a and b have a direct path between them, and b and c have a direct path between them, then balloon colors on these three squares are distinct.

Andryusha wants to use as little different colors as possible. Help him to choose the colors!

Input

The first line contains single integer n ($3 \le n \le 2 \cdot 10^5$) — the number of squares in the park.

Each of the next $(n - 1)$ lines contains two integers x and y ($1 \le x, y \le n$) — the indices of two squares directly connected by a path.

It is guaranteed that any square is reachable from any other using the paths.

Output

In the first line print single integer k — the minimum number of colors Andryusha has to use.

In the second line print n integers, the i-th of them should be equal to the balloon color on the i-th square. Each of these numbers should be within range from 1 to k.

Examples

input

```
3
2 3
1 3
```

output

```
3
1 3 2
```

input

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

output

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

input

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

output

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

Note

In the first sample the park consists of three squares: $1 \to 3 \to 2$. Thus, the balloon colors have to be distinct.

Illustration for the first sample.

In the second example there are following triples of consequently connected squares:

- $1 \to 3 \to 2$
- $1 \to 3 \to 4$
- $1 \to 3 \to 5$
- $2 \to 3 \to 4$
- $2 \to 3 \to 5$
- $4 \to 3 \to 5$

We can see that each pair of squares is encountered in some triple, so all colors have to be distinct.

Illustration for the second sample.

In the third example there are following triples:

- $1 \to 2 \to 3$
- $2 \to 3 \to 4$
- $3 \to 4 \to 5$

We can see that one or two colors is not enough, but there is an answer that uses three colors only.

Illustration for the third sample.

D. Innokenty and a Football League

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

Innokenty is a president of a new football league in Byteland. The first task he should do is to assign short names to all clubs to be shown on TV next to the score. Of course, the short names should be distinct, and Innokenty wants that all short names consist of **three letters**.

Each club's full name consist of two words: the team's name and the hometown's name, for example, "`DINAMO BYTECITY`". Innokenty doesn't want to assign strange short names, so he wants to choose such short names for each club that:

1. the short name is the same as three first letters of the team's name, for example, for the mentioned club it is "`DIN`",
2. or, the first two letters of the short name should be the same as the first two letters of the team's name, while the third letter is the same as the first letter in the hometown's name. For the mentioned club it is "`DIB`".

Apart from this, there is a rule that if for some club x the second option of short name is chosen, then there should be no club, for which the first option is chosen which is the same as the first option for the club x. For example, if the above mentioned club has short name "`DIB`", then no club for which the first option is chosen can have short name equal to "`DIN`". However, it is possible that some club have short name "`DIN`", where "`DI`" are the first two letters of the team's name, and "`N`" is the first letter of hometown's name. Of course, no two teams can have the same short name.

Help Innokenty to choose a short name for each of the teams. If this is impossible, report that. If there are multiple answer, any of them will suit Innokenty. If for some team the two options of short name are equal, then Innokenty will formally think that only one of these options is chosen.

Input

The first line contains a single integer n ($1 \le n \le 1000$) — the number of clubs in the league.

Each of the next n lines contains two words — the team's name and the hometown's name for some club. Both team's name and hometown's name consist of uppercase English letters and have length at least 3 and at most 20.

Output

It it is not possible to choose short names and satisfy all constraints, print a single line "`NO`".

Otherwise, in the first line print "`YES`". Then print n lines, in each line print the chosen short name for the corresponding club. Print the clubs in the same order as they appeared in input.

If there are multiple answers, print any of them.

Examples

input

```
2
DINAMO BYTECITY
FOOTBALL MOSCOW
```

output

```
YES
DIN
FOO
```

input

```
2
DINAMO BYTECITY
DINAMO BITECITY
```

output

```
NO
```

input

```
3
PLAYFOOTBALL MOSCOW
PLAYVOLLEYBALL SPB
GOGO TECHNOCUP
```

output

```
YES
PLM
PLS
GOG
```

input

```
3
ABC DEF
ABC EFG
ABD OOO
```

output

```
YES
ABD
ABE
ABO
```

Note

In the first sample Innokenty can choose first option for both clubs.

In the second example it is not possible to choose short names, because it is not possible that one club has first option, and the other has second option if the first options are equal for both clubs.

In the third example Innokenty can choose the second options for the first two clubs, and the first option for the third club.

In the fourth example note that it is possible that the chosen short name for some club x is the same as the first option of another club y if the first options of x and y are different.

E. Underground Lab

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

The evil Bumbershoot corporation produces clones for gruesome experiments in a vast underground lab. On one occasion, the corp cloned a boy Andryusha who was smarter than his comrades. Immediately Andryusha understood that something fishy was going on there. He rallied fellow clones to go on a feud against the evil corp, and they set out to find an exit from the lab. The corp had to reduce to destroy the lab complex.

The lab can be pictured as a connected graph with n vertices and m edges. k clones of Andryusha start looking for an exit in some of the vertices. Each clone can traverse any edge once per second. Any number of clones are allowed to be at any vertex simultaneously. Each clone is allowed to stop looking at any time moment, but he must look at his starting vertex at least. The exit can be located at any vertex of the lab, hence each vertex must be visited by at least one clone.

Each clone can visit at most vertices before the lab explodes.

Your task is to choose starting vertices and searching routes for the clones. Each route can have at most vertices.

Input

The first line contains three integers n, m, and k ($1 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$, $1 \le k \le n$) — the number of vertices and edges in the lab, and the number of clones.

Each of the next m lines contains two integers x_i and y_i ($1 \le x_i, y_i \le n$) — indices of vertices connected by the respective edge. The graph is allowed to have self-loops and multiple edges.

The graph is guaranteed to be connected.

Output

You should print k lines. i-th of these lines must start with an integer c_i () — the number of vertices visited by i-th clone, followed by c_i integers — indices of vertices visited by this clone in the order of visiting. You have to print each vertex every time it is visited, regardless if it was visited earlier or not.

It is guaranteed that a valid answer exists.

Examples

input

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

output

```
3 2 1 3
```

input

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

output

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

Note

In the first sample case there is only one clone who may visit vertices in order (2, 1, 3), which fits the constraint of 6 vertices per clone.

In the second sample case the two clones can visited vertices in order (2, 1, 3) and (4, 1, 5), which fits the constraint of 5 vertices per clone.

F. Axel and Marston in Bitland

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A couple of friends, Axel and Marston are travelling across the country of Bitland. There are n towns in Bitland, with some pairs of towns connected by one-directional roads. Each road in Bitland is either a pedestrian road or a bike road. There can be multiple roads between any pair of towns, and may even be a road from a town to itself. However, no pair of roads shares the starting and the destination towns along with their types simultaneously.

The friends are now located in the town 1 and are planning the travel route. Axel enjoys walking, while Marston prefers biking. In order to choose a route diverse and equally interesting for both friends, they have agreed upon the following procedure for choosing the road types during the travel:

- The route starts with a pedestrian route.
- Suppose that a beginning of the route is written in a string s of letters P (pedestrain road) and B (biking road). Then, the string is appended to s, where stands for the string s with each character changed to opposite (that is, all pedestrian roads changed to bike roads, and vice versa).

In the first few steps the route will look as follows: P, PB, PBBP, PBBPBPPB, PBBPBPPBBPPBPBBP, and so on.

After that the friends start travelling from the town 1 via Bitlandian roads, choosing the next road according to the next character of their route type each time. If it is impossible to choose the next road, the friends terminate their travel and fly home instead.

Help the friends to find the longest possible route that can be travelled along roads of Bitland according to the road types choosing procedure described above. If there is such a route with more than 10^{18} roads in it, print -1 instead.

Input

The first line contains two integers n and m ($1 \le n \le 500$, $0 \le m \le 2n^2$) — the number of towns and roads in Bitland respectively.

Next m lines describe the roads. i-th of these lines contains three integers v_i, u_i and t_i ($1 \le v_i, u_i \le n$, $0 \le t_i \le 1$), where v_i and u_i denote start and destination towns indices of the i-th road, and t_i decribes the type of i-th road (0 for a pedestrian road, 1 for a bike road). It is guaranteed that for each pair of distinct indices i, j such that $1 \le i, j \le m$, either $v_i \ne v_j$, or $u_i \ne u_j$, or $t_i \ne t_j$ holds.

Output

If it is possible to find a route with length strictly greater than 10^{18}, print -1. Otherwise, print the maximum length of a suitable path.

Examples

input

```
2 2
1 2 0
2 2 1
```

output

```
3
```

input

```
2 3
1 2 0
2 2 1
2 2 0
```

output

```
-1
```

Note

In the first sample we can obtain a route of length 3 by travelling along the road 1 from town 1 to town 2, and then following the road 2 twice from town 2 to itself.

In the second sample we can obtain an arbitrarily long route by travelling the road 1 first, and then choosing road 2 or 3 depending on the necessary type.

G. Andryusha and Nervous Barriers

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Andryusha has found a perplexing arcade machine. The machine is a vertically adjusted board divided into square cells. The board has w columns numbered from 1 to w from left to right, and h rows numbered from 1 to h from the bottom to the top.

Further, there are barriers in some of board rows. There are n barriers in total, and i-th of them occupied the cells l_i through r_i of the row u_i. Andryusha recollects well that no two barriers share the same row. Furthermore, no row is completely occupied with a barrier, that is, at least one cell in each row is free.

The player can throw a marble to any column of the machine from above. A marble falls downwards until it encounters a barrier, or falls through the bottom of the board. A marble disappears once it encounters a barrier but is replaced by two more marbles immediately to the left and to the right of the same barrier. In a situation when the barrier is at an edge of the board, both marbles appear next to the barrier at the side opposite to the edge. More than one marble can occupy the same place of the board, without obstructing each other's movement. Ultimately, all marbles are bound to fall from the bottom of the machine.

Examples of marble-barrier interaction.

Peculiarly, sometimes marbles can go through barriers as if they were free cells. That is so because the barriers are in fact alive, and frightened when a marble was coming at them from a very high altitude. More specifically, if a marble falls towards the barrier i from relative height more than s_i (that is, it started its fall strictly higher than $u_i + s_i$), then the barrier evades the marble. If a marble is thrown from the top of the board, it is considered to appear at height $(h+1)$.

Andryusha remembers to have thrown a marble once in each of the columns. Help him find the total number of marbles that came down at the bottom of the machine. Since the answer may be large, print it modulo 10^9+7.

Input

The first line contains three integers h, w, and n ($1 \le h \le 10^9$, $2 \le w \le 10^5$, $0 \le n \le 10^5$) — the number of rows, columns, and barriers in the machine respectively.

Next n lines describe barriers. i-th of these lines containts four integers u_i, l_i, r_i, and s_i ($1 \le u_i \le h$, $1 \le l_i \le r_i \le w$, $1 \le s_i \le 10^9$) — row index, leftmost and rightmost column index of i-th barrier, and largest relative fall height such that the barrier does not evade a falling marble. It is guaranteed that each row has at least one free cell, and that all u_i are distinct.

Output

Print one integer — the answer to the problem modulo 10^9+7.

Examples

input
```
10 5 1
3 2 3 10
```
output
```
7
```

input
```
10 5 2
3 1 3 10
5 3 5 10
```
output
```
16
```

input
```
10 5 2
3 1 3 7
5 3 5 10
```
output
```
14
```

input
```
10 15 4
7 3 9 5
6 4 10 1
1 1 4 10
4 11 11 20
```
output
```
53
```

Note

In the first sample case, there is a single barrier: if one throws a marble in the second or the third column, two marbles come out, otherwise there is only one. The total answer is 7.

In the second sample case, the numbers of resulting marbles are $2, 2, 4, 4, 4$ in order of indexing columns with the initial marble.

In the third sample case, the numbers of resulting marbles are $1, 1, 4, 4, 4$. Note that the first barrier evades the marbles falling from the top of the board, but does not evade the marbles falling from the second barrier.

In the fourth sample case, the numbers of resulting marbles are $2, 2, 6, 6, 6, 6, 6, 6, 6, 1, 2, 1, 1, 1, 1$. The picture below shows the case when a marble is thrown into the seventh column.

The result of throwing a marble into the seventh column.

H. Intranet of Buses

time limit per test: 10 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A new bus route is opened in the city . The route is a closed polygon line in the place, with all segments parallel to one of the axes. m buses will operate on the route. All buses move in a loop along the route in the same direction with equal constant velocities (stopping times are negligible in this problem).

Buses start their movement in the first vertex of the route with equal interval. Suppose that T is the total time for a single bus to travel the whole loop of the route. Then, the bus 1 starts moving at time 0, the bus 2 starts at time T/m, the bus 3 starts at time $2T/m$, and so on; finally, the bus m starts moving at time $(m-1)T/m$. Thus, all intervals between pairs of consecutive buses (including the interval between the last and the first bus) are equal.

Buses can communicate with each other via wireless transmitters of equal power. If the transmitters have power D, then only buses within distance D of each other can communicate.

The buses are also equipped with a distributed system of schedule tracking. For all buses to stick to the schedule, the system has to synchronize the necessary data between all buses from time to time. At the moment of synchronization, the bus 1 communicates with the bus 2, the bus 2 — with bus 3, and so on; also, the bus m communicates with the bus 1.

As a research employee, you are tasked with finding the smallest value of D such that it is possible to find a time moment to perform synchronization once all buses have started moving.

Input

The first line contains two integers n and m ($2 \leq n, m \leq 10^5$) — the number of vertices of the polygonal line, and the number of buses respectively.

Next n lines describe the vertices of the route in the traversing order. Each of these lines contains two integers x_i, y_i ($-1000 \leq x_i, y_i \leq 1000$) — coordinates of respective vertex.

It is guaranteed that each leg of the route (including the leg between the last and the first vertex) is paralles to one of the coordinate axes. Moreover, no two subsequent vertices of the route coincide. The route is allowed to have self-intersections, and travel along the same segment multiple times.

Output

Print one real number — the answer to the problem. Your answer will be accepted if the relative or the absolute error doesn't exceed 10^{-6}.

Examples

input

```
4 2
0 0
0 1
1 1
1 0
```

output

```
1.000000000
```

input

```
2 2
0 0
1 0
```

output

```
0.000000000
```

Note

Suppose that each bus travel 1 distance unit per second.

In the first sample case, in 0.5 seconds buses will be at distance 1, hence we can choose $D = 1$.

In the second sample case, in 0.5 seconds both buses will be at $(0.5, 0)$, hence we can choose $D = 0$.