Problem A
Colourful Graph
Time limit: 5 seconds

Consider an undirected graph on n vertices. A k-colouring of the graph is simply an assignment to each vertex one of the k colours. There are no other restrictions – two vertices can get the same colour even if they are connected by an edge.

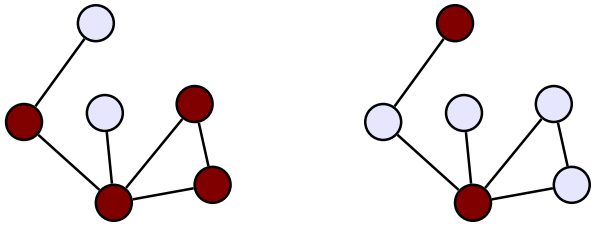
You are given two k-colourings s and t. You want to transform from the initial colouring s to the final colouring t step by step. In each step, each vertex may change its colour to one of its neighbours' colour, or keep its current colour.

Formally, you are looking for a sequence of k-colourings $C_0, C_1, C_2, \ldots, C_\ell$ such that $C_0 = s$, $C_\ell = t$, and every $C_i(x)$ equals either

- $C_{i-1}(x)$; or
- $C_{i-1}(y)$ where (x, y) is an edge in the graph

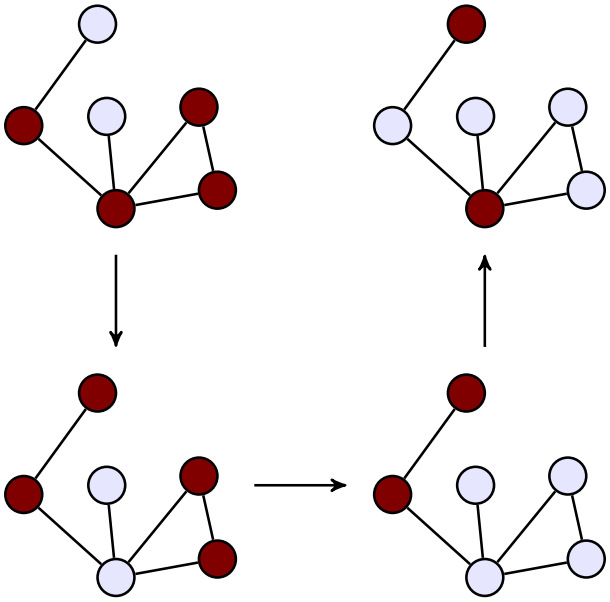
The input graph is guaranteed to be connected and has no self loops. Determine whether it is possible to construct such a sequence, and output one if there is any. The sequence need not be the shortest as long as it uses at most $20\,000$ steps.

For example, suppose you need to start with the initial colouring on the left and end up with the final colouring on the right:



initial final

One solution is as follows:



Input

The first line of input contains three integers n, m, and k. Here n is the number of vertices, m is the number of edges and k is the number of colours ($1 \leq n, k \leq 100$, $0 \leq m \leq \binom{n}{2}$). The second line of input contains n integers, each in the range $[0, k-1]$. They indicate the initial colouring. The third line of input contains n integers, each in the range $[0, k-1]$. They indicate the final colouring. Each of the following m lines contains two integers x_i and y_i, representing an edge in the graph ($1 \leq x_i, y_i \leq n$).

The input graph is connected. There are no self loops or multiple edges.

Output

Output `Impossible` if there is no such sequence. Otherwise, output one or more lines describing a sequence of colouring $C_0, C_1, C_2, \ldots, C_\ell$ ($\ell \leq 20\,000$). Each line contains n integers separated by a single space, describing the colouring. The first line indicates the initial colouring and the last line indicates the final colouring.

You can use at most $20\,000$ steps, therefore the output contains at most $20\,001$ lines.

Sample Input 1

```
6 6 2
0 1 0 1 1 1
1 0 0 0 1 0
1 2
2 3
3 5
5 4
4 6
6 5
```

Sample Output 1

```
0 1 0 1 1 1
1 1 0 1 0 1
1 1 0 0 1 0
1 0 0 0 1 0
```

Sample Input 2

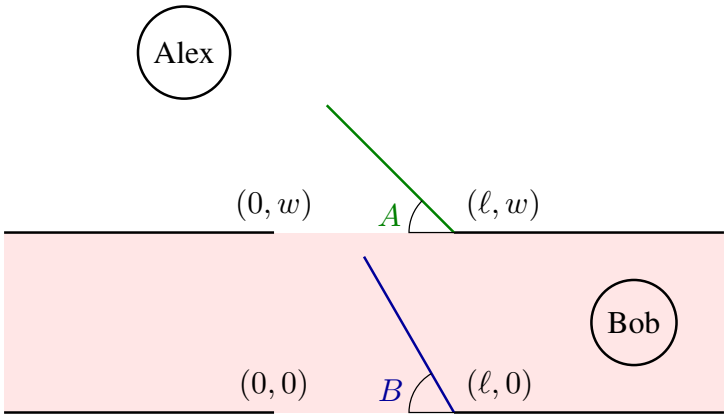
```
1 0 2
0
1
```

Sample Output 2

```
Impossible
```

Problem B
Doors
Time limit: 3 seconds

Alex is a circle of radius R. Well, life as a circle is not easy. If he were a point, moving around and passing through doors would be effortless. But now he has to carefully inspect the surroundings before making every move.



Alex's is initially at position $(0, y_\infty)$, where y_∞ is much bigger than the width of the corridor w. Alex wants to meet Bob, who happens to be living to the far right of the corridor (may as well be somewhere at $(x_\infty, w/2)$, where x_∞ is much bigger than ℓ). The lengths of the two doors are ℓ. It is guaranteed that $\ell \leq w$, so that door B will never hit the opposite side of the wall.

You are given T scenarios. In each scenario, the angles A and B are given (both are radians in the range $[0, \pi]$). Find the largest $r \leq R$ such that when Alex's radius is shrunk to r, he can reach Bob while avoiding the obstacles (walls and doors).

Formally, Alex when shrunk to radius r can reach Bob if and only if there exists a (continuous) curve from $(0, y_\infty)$ to $(x_\infty, w/2)$ such that the minimal distance between a point on the curve and a point on an obstacle (a wall or a door) is at least r. In particular, if $r = 0$ then Alex will be able to reach Bob.

Input

The first line of input consists of three integers, R, ℓ, and w ($1 \leq \ell, w, R \leq 100$ and $\ell \leq w$). The second line of input consists of an integer T ($1 \leq T \leq 10\,000$), the number of scenarios to follow. Each of the next T lines consists of a pair of real numbers, representing angles A and B (in radians). The numbers are given with exactly 4 decimal places.

Output

For each scenario, output the required answer on a separate line. Your answer will be accepted if its absolute or relative error (compared to the judge's answer) is at most 10^{-5}.

Sample Input 1	Sample Output 1
10 6 8 4 0.0000 0.0000 3.1415 0.0000 1.0472 0.0000 1.0472 1.5708	0.000000000 3.000000000 2.598079885 1.000000000

Problem C
Playing with Numbers
Time limit: 1 second

You have a list of N numbers, each of the form $2^a 3^b$ for some non-negative integers a and b. You want to perform $N-1$ operations on these numbers. Each operation acts on two numbers X and Y of your choice from the list, replacing them with a new number $\mathrm{op}(X, Y)$. After each operation, your list has one fewer number.

In this task, an operation op can be $\gcd$ or lcm (they stand for greatest common divisor and least common multiple, respectively). There are a total of N scenarios: You may apply "$\gcd$" operations k times and "lcm" operations $N-1-k$ times. For each of the N scenarios, what is the largest possible outcome after these $N-1$ operations? What about the smallest possible outcome?

Input

The first line consists of a single integer N ($1 \leq N \leq 50\,000$). The following N lines each contains a pair of integers a_i and b_i ($0 \leq a_i, b_i \leq 1\,000$), indicating that the ith number in your initial list is $2^{a_i} 3^{b_i}$.

Output

Output N lines in total. On line i ($i = 1, \ldots, N$), output four space-separated integers a, b, a' and b'. The first pair of integers a and b indicate that the largest possible outcome is $2^a 3^b$ with $i-1$ "$\gcd$" operations (and therefore $N-i$ "lcm" operations). The second pair of integers a' and b' indicate that the smallest possible outcome is $2^{a'} 3^{b'}$, again with $i-1$ "$\gcd$" operations.

Explanation for sample data

The three numbers are $2^0 3^0 = 1$, $2^1 3^2 = 18$, and $2^2 3^0 = 4$.

1. When $i = 1$, we can only take lcm. $\mathrm{lcm}(1, 18, 4) = 36 = 2^2 3^2$.
2. When $i = 2$, the largest outcome is $\mathrm{lcm}(18, \gcd(1, 4)) = 18 = 2^1 3^2$, and the smallest outcome is $\gcd(1, \mathrm{lcm}(18, 4)) = 1 = 2^0 3^0$.
3. When $i = 3$, we can only take $\gcd$. $\gcd(1, 18, 4) = 1 = 2^0 3^0$.

Sample Input 1	Sample Output 1
3 0 0 1 2 2 0	2 2 2 2 1 2 0 0 0 0 0 0

This page is intentionally left (almost) blank.

Problem D
Peak Tram
Time limit: 3 seconds

Inspired by the Peak Tram in Hong Kong, you are going to build your own peak tram on a hill, together with the buildings near it. The peak tram and the buildings are aligned on a straight line. The peak tram is at position 0 on the line. There are n buildings, where building i is at position i on the line ($i = 1, 2, \ldots, n$). Let the height of building i be h_i. The peak tram starts ascending from height 0, and the passengers would look at the building that first touches the horizontal ray from the peak tram at the current height. In other words, building i is visible some time during the ascent if and only if $h_i > h_j$ for all $j < i$ (i.e., no other buildings are blocking the sight). Assume the hill is sufficiently tall so the peak tram can always ascend to at least the height of the tallest building.

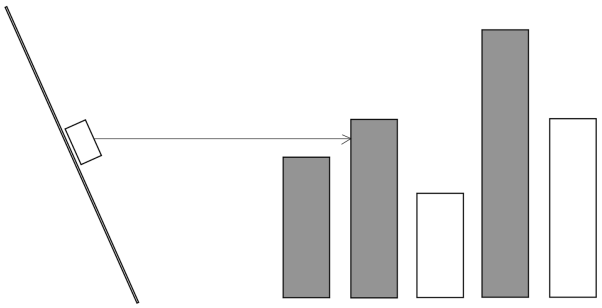


Figure D.1: The figure shows the peak tram (leftmost) and 5 buildings. The buildings in grey are visible during the ascent. It demonstrates an optimal answer to the sample input, with resultant heights of $5, 6, 4, 9, 6$.

You want the passengers to enjoy the view from the peak tram, so there should be at least k buildings that are visible some time during the ascent. However, the problem is that you do not have any building yet, and you now have to build those n buildings (i.e., you have to choose h_i). Each building i has a preferred height p_i and a cost per height difference c_i. The cost for making the height of building i to be h_i is given by $|h_i - p_i| \times c_i$. Also the heights h_i you choose must be positive integers. Your task is to determine the minimum total cost so that at least k buildings are visible during the ascent.

Input

The first line of the input contains two integers n and k ($1 \leq k \leq n \leq 70$). Each of the following n lines contains two integers p_i and c_i ($1 \leq p_i \leq 10^9$, $1 \leq c_i \leq 1\,000$).

Output

Output one integer, the minimum total cost so that at least k buildings are visible during the ascent.

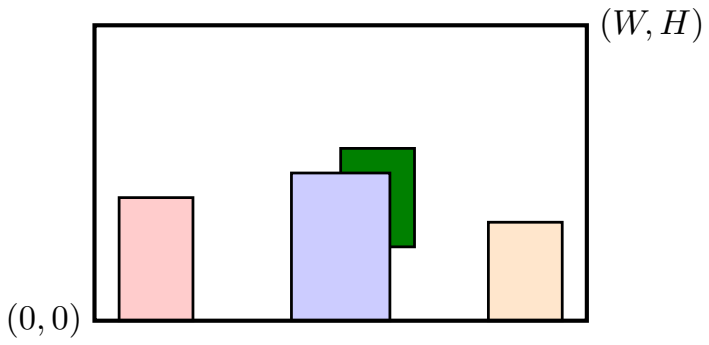
Sample Input 1	Sample Output 1
5 3 5 3 3 2 4 8 9 4 6 2	6

Problem E
Peak Tower
Time limit: 2 seconds

After taking the Peak Tram, you reach the *Peak Tower* — an ideal place to take a picture of the Victoria Harbour.

Being a must-see attraction, the Peak Tower is packed with tourists, and it's pretty hard to take a picture without many people blocking the beautiful scene. Occasionally birds are flying around as well. You are wondering: how to take the best picture to minimise the area blocked by tourists and birds?

You are taking a picture to capture a scene of width W and height H (both in metres). The bottem left corner of the scene are at coordinates $(0, 0)$. Each tourist or bird can be modelled as an axis-aligned rectangle that moves at a constant velocity.



In the above example there are three tourists (red, blue, and orange) and a bird (dark green). You want to minimise the area of the union of these rectangles.

Each tourist or bird can be represented by a rectangle of width w and height h (both in metres). Its motion is described by:

- s_x: The initial x-coordinate of its bottom left corner (at time $t = 0$).
- s_y: The initial y-coordinate of its bottom left corner (at time $t = 0$).
- v_x: The horizontal velocity in ms^{-1} at which it moves (positive if it's moving rightwards).
- v_y: The vertical velocity in ms^{-1} at which it moves (positive if it's moving upwards).

Obviously the best picture you can take is to wait until all objects are gone. However, you have a tight schedule and must take the picture within E seconds. Given the information about the motion of the N objects, when should you take the picture to minimise the area blocked by them?

Input

The first line consists of an integer N ($0 \le N \le 50$). The second line consists of 3 floating point numbers W, H, E ($0 < W, H \le 300$, $0 < E \le 100$). Each of the following N lines consists of 6 floating point numbers w, h, s_x, s_y, v_x, v_y, representing the size and motion of the ith object ($0 < w, h \le 100$, $-300 \le s_x, s_y, v_x, v_y \le 300$). Each floating point number has at most 6 decimal places.

Output

Output a floating point number to indicate the minimal possible total area blocked by tourists and birds. Any solution with an absolute or relative error within 10^{-6} will be accepted.

Sample Input 1	Sample Output 1
0 300.0 200.0 6.0	0.0

Sample Input 2	Sample Output 2
2 270.0 200.0 10.0 15.0 20.0 125.0 0.0 10.0 0.0 15.0 15.0 240.0 0.0 −10.0 0.0	300.0

Problem F
Perfect k-ary Tree
Time limit: 13 seconds

A graph G is given, which is a tree with N nodes. The nodes are labelled $1, 2, \ldots, N$. Count the number of subgraphs of G which is a perfect k-ary tree. An unrooted tree is called a perfect k-ary tree if it is possible to root the tree such that (1) every non-leaf node has exactly k children and (2) the distance between the root and a leaf is the same for all leaves.

For example, the graph in Sample Input contains 6 perfect binary (2-ary) trees, namely $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{1, 2, 3\}$, $\{2, 3, 4\}$.

Since the answer may be too large, output it modulo $1\,000\,000\,007$ ($10^9 + 7$).

Input

The first line of input consists of two integers N and k ($1 \leq N \leq 100\,000$, $2 \leq k \leq 5$). The following $N - 1$ lines each contain a pair of integers, u_i and v_i ($1 \leq u_i, v_i \leq N$), meaning that there is an edge directly connecting nodes u_i and v_i. It is guaranteed that the given graph is a tree.

Output

Output the required answer on one line.

Sample Input 1	Sample Output 1
4 2 1 2 2 3 3 4	6

This page is intentionally left (almost) blank.

Problem G
Scaffolding
Time limit: 1 second

by Greg Hume on Wikipedia CC BY-SA

Bamboo scaffolding is very popular in Hong Kong. It is widely used in construction and maintenance of buildings, when high temporary structures are built to hold construction workers. You will also find bamboo scaffolds in Cantonese opera theatres and at the Bun Scrambling Competition in Cheung Chau.

As a construction worker, you want to build a bamboo scaffold for an upcoming festival. There are specific requirements for the final scaffold, which will be part of a vertical, 2-dimensional grid. $N + 1$ vertical long bamboos are already established, so there are N columns in between. Initially all columns are empty. Our task is to place exactly H_i horizontal short bamboos in the ith column. All short bamboos in each column are to be equally spaced, literally from the ground up.

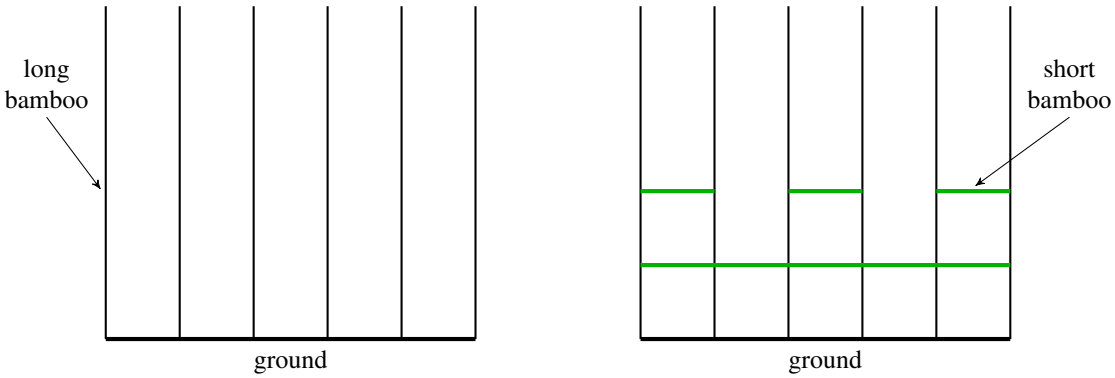


Figure G.1: The left diagram illustrates the initial scaffold that has N empty columns. The right diagram shows the final scaffold for Sample Input 2.

You can carry at most M short bamboos at a time. You will install the short bamboos in a number of rounds. In each round, you take at most M short bamboos with you, and start anywhere on ground level.

You can only stand on a short bamboo that was already built (or on ground level). Each round consists of a sequence of the following two kinds of actions:

- Climb left, right or up if a short bamboo is already in place there
- Place a short bamboo to your left, right or up if there is no short bamboo already placed, and then immediately climb to stand on the new bamboo

Note that you cannot go down while carrying short bamboos. You cannot place a short bamboo below you either.

Once you finish placing all the short bamboos in the current round, you go down to the ground level. You then take at most M short bamboos with you and start another round from the ground level. This goes on until you have built the desired scaffold, with exactly H_i short bamboos in column i for all i.

What is the minimum number of rounds you need?

Input

The first line contains two integers, N and M ($1 \leq N \leq 100\,000$, $1 \leq M \leq 10^9$). The second line contains N integers $H_1, \ldots, H_N$ ($0 \leq H_i \leq 100\,000$).

Output

Output an integer representing the minimum number of rounds.

Sample Input 1	Sample Output 1
3 4 2 1 5	2

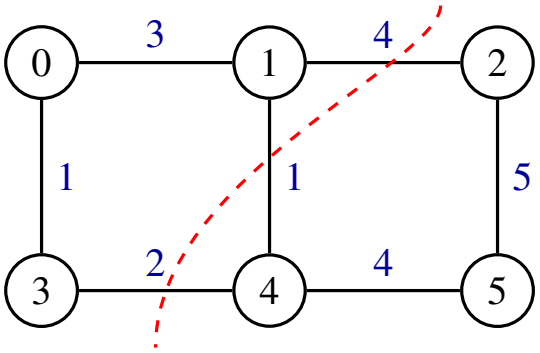
Sample Input 2	Sample Output 2
5 10 2 1 2 1 2	3

Problem H
Slim Cut
Time limit: 5 seconds

Consider a weighted, undirected graph $G(V, E; w)$. A cut is a partition of the vertices into two non-empty subsets S and $\overline{S}$. Let us define the *slimness* of the cut $\{S, \overline{S}\}$ to be

$$\frac{\max\limits_{(u,v)\in E, u\in S, v\in\overline{S}} w(u,v)}{\min(|S|, |\overline{S}|)}$$

In other words, the sliminess of the cut is the maximum weight among all edges crossng the cut, divided by size of the smaller part. For example, the cut below has slimness $\max(4,1,2)/3 = 4/3$, by choosing $S = \{0,1,3\}$ and $\overline{S} = \{2,4,5\}$ (or the other way round).



Among all possible cuts, which one is the slimmest (i.e. has the minimum slimness)? You are asked to find its slimness.

Note: Time limit is strict. Your implementation must be efficient.

Input

The first line contains two integers n and m, representing the number of vertices and the number of edges in the graph ($2 \le n \le 14\,000$, $1 \le m \le 30\,000$). The vertices are labelled from 0 to $n-1$. The next m lines each contains three space-separated integers x_i, y_i, w_i, representing an edge connecting vertices x_i and y_i of weight w_i ($0 \le x_i, y_i < n$, $1 \le w_i \le 10^8$).

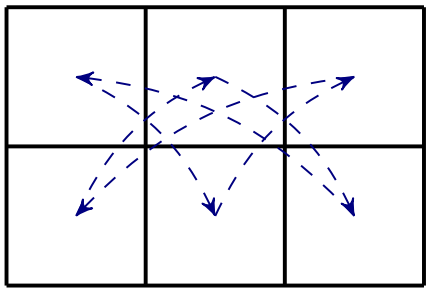
The graph is guaranteed to be connected. It has no self loops or multiple edges.

Output

Output a floating point number representing the slimness of the slimmest cut of the graph. Any answer with an absolute or relative error within 10^{-8} will be accepted.

Sample Input 1	Sample Output 1
6 7 0 1 3 1 2 4 0 3 1 1 4 1 2 5 5 3 4 2 4 5 4	1.33333333333

Problem I
Special Tour
Time limit: 4 seconds



Given an grid with N rows and M columns, your task is to find a tour such that:

1. The tour has length at least two
2. All squares in the grid are visited exactly once
3. Let $a_1, \ldots, a_{NM}$ be the sequence of visited squares. Then $d(a_k, a_{k+1}) = 2$ or 3 for every $1 \leq k < NM$. Also, $d(a_{NM}, a_1) = 2$ or 3.

   Here, $d(p, q)$ denotes the Manhattan distance between p and q. More precisely, if p is the square on row r_p and column c_p and q is the square on row r_q and column c_q, then we write $p = (r_p, c_p), q = (r_q, c_q)$ and $d(p, q) = |r_p - r_q| + |c_p - c_q|$.

Input

The first and only line of input consists of two integers N and M ($1 \leq N, M \leq 200$).

Output

If there is no such tour, output `-1`. Otherwise, output $N \times M$ lines. The ith line contains two space-separated integers, denoting the row and column numbers of a_i. If there is more than one such tour, any one will be accepted.

Sample Input 1	Sample Output 1
2 3	1 1 2 2 1 3 2 1 1 2 2 3

Sample Input 2	Sample Output 2
1 1	-1

This page is intentionally left (almost) blank.

Problem J
Taboo
Time limit: 1 second

Taboo is a popular party game. In this game one player, the Clue Giver, prompts his/her teammates to guess a keyword by giving clues. The Clue Giver is also given a list of taboo strings that must not appear in the clues. For example, if the keyword is "Bruce Lee", the famous kung-fu star, then the taboo strings may be "actor", "kung-fu", "fighting", "martial arts" and "*The Game of Death*" (Bruce Lee's final film). The Clue Giver may try such clues as "*Fist of Fury* star" and "Jeet Kune Do master" to avoid the taboo. Taboo strings bring challenges and fun to the guessing game.

Short clues are preferred, but now you are interested in the opposite: what is the longest clue? Given N taboo strings $s_1, \ldots, s_N$, what is the longest clue string s such that none of $s_1, \ldots, s_N$ appears as a substring of s? For simplicity, all taboo strings and your clue are represented as binary strings consisting only of `0`'s and `1`'s.

Input

The first line contains an integer, N, the number of taboo strings ($1 \leq N \leq 15\,000$). The following N lines each contains a non-empty binary string s_i, for $1 \leq i \leq N$. The sum of lengths of $s_1, \ldots, s_N$ will be at most $200\,000$.

Output

If your clue can be arbitrarily long, output `-1`. Otherwise, output a line containing the longest binary string that does not contain $s_1, \ldots, s_N$ as a substring. If there is more than one such longest string, output the one that is also smallest in lexicographic order.

Sample Input 1

```
5
00
01
10
110
111
```

Sample Output 1

```
11
```

Sample Input 2

```
3
00
01
10
```

Sample Output 2

```
-1
```

This page is intentionally left (almost) blank.

Problem K
Team Up
Time limit: 4 seconds

In online games, players form teams so they can cover each others' weakness with different skills (e.g. healing, buffing, teleporting). There are n types of skills in the game, labelled from 1 to n. To form a team, every skill must be covered by at least one player in the team.

The skills each player has depend on the character class (e.g. warrior, thief, wizard) of the player. There are m character classes in the game, and the ith class ($1 \leq i \leq m$) possesses skills $S_i \subseteq \{1, 2, \ldots, n\}$. For every $i \neq j$, we have $S_i \neq S_j$. Moreover, for every $i \neq j$, exactly one of the following properties holds:

1. $S_i \subset S_j$
2. $S_j \subset S_i$
3. S_i and S_j are disjoint ($S_i \cap S_j = \emptyset$)

There are p players, labelled form 1 to p. Each player can belong to at most one team. Given the character class of each player, compute the maximum number of teams they can form and output the teaming.

Input

- The first line contains three integers n, m, p ($1 \leq n \leq 100\,000$, $1 \leq m, p \leq 300\,000$).
- The next m lines each describes the skills that each character class possesses. The ith line of these m lines contains $|S_i| + 1$ integers separated by a space. The first integer will be $|S_i|$ ($1 \leq |S_i| \leq n$), and the following $|S_i|$ integers describe the elements in S_i. It is guaranteed that $|S_1| + \cdots + |S_m|$ is at most $500\,000$.
- The last line contains p space-separated integers, each ranging from 1 to m. The ith integer denotes the character class of the i-th player.

Output

- Output an integer on the first line representing the maximum number of teams that can be formed.
- For every team, output a line containing space-separated integers to describe the players in the team: the first integer represents the size of the team, the following integers represent the labels of the players. You can output the label in any order without duplication.

If there is more than one optimal solution, output any one of them.

Sample Input 1	Sample Output 1
3 4 7 1 1 1 2 2 1 2 1 3 1 2 2 3 4 4 2	2 3 1 3 5 2 4 6

This page is intentionally left (almost) blank.