Problem A. Bipartite Matching

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

You are given a bipartite graph. Find the maximum matching in it.

Input

The first line of input contains two integers: n and m — the number of vertices in parts ($1 \le n, m \le 250$). Let vertices in each part be independently numbered starting from 1.

The following n lines describe edges, the i-th line contains one or more integers: the numbers of vertices in the second part connected to the i-th vertex in the first part, followed by 0.

Output

The first line of output must contain one integer l — the number of edges in the maximum matching. The following l lines must describe edges of the matching, the i-th line must contain two integers u_i and v_i — the numbers of vertices in the first and the second part, correspondingly, connected by the i-th edge of the matching.

Example

standard input	standard output
2 2 1 2 0 2 0	2 1 1 2 2

Problem B. Minimal Vertex Cover

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

You are given a bipartite graph and the maximum matching in it.

Recall that a set C of vertices is called a *vertex cover* if each of the edges has at least one of its end vertices in C.

Find the minimum vertex cover of the given graph.

Input

The first line of input contains two integers n and m — the number of vertices in the parts ($1 \le n, m \le 4\,000$). The vertices in each part are numbered independently starting from 1.

Each of the following n lines describe edges of the graph. The i-th of these lines starts with an integer k_i — the number of vertices adjacent to the i-th vertex of the first part, k_i integers follow — these vertices. The sum of k_i doesn't exceed $500\,000$.

The last line of input contains n integers that describe the maximum matching in the given graph. The i-th number is the number of the vertex of the second part that is matched with the i-th vertex of the first part, or 0 if the i-th vertex of the first part is not covered by the matching.

Output

The first line of the output file must contain an integer s — the number of vertices in the minimum vertex cover.

The following two lines must describe the minimum vertex cover C itself. The second line must start with an integer s_1 — the number of vertices of the first part in C, followed by s_1 integers — the numbers of these vertices. The third line must start with an integer s_2 — the number of vertices of the second part in C, followed by s_2 integers — the numbers of these vertices.

Example

standard input	standard output
3 2 2 1 2 1 2 1 2 1 2 0	2 1 1 1 2

Problem C. Path Cover

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

You are given a directed acyclic graph G.

Find the minimum set of paths in G such that each vertex belongs to exactly one path.

Input

The first line of input contains two integers n and m — the number of vertices and the number of edges of the graph ($2 \le n \le 1000$, $0 \le m \le 10^5$).

The following m lines describe edges, the i-th line contains two integers u_i and v_i — the beginning and the end of the i-th edge. It is guaranteed that the described graph has no cycles.

Output

Output one integer — the minimum number of paths needed to cover all vertices.

Example

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

Problem D. Cubes

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Parents have bought little Senya a set of letter cubes as a birthday present. Each cube has a letter at each of its six faces.

Now Senya wants to show off to his elder sister that he can read. So he wants to compose his name out of his cubes. But is turned out quite difficult — two letters of her name might be at different faces of the same cube, so Senya cannot use them both. Fortunately, some letters can be at faces of several cubes.

Help Senya to compose his sister's name using his cubes, or find out that it is impossible.

Input

The first line of input contains n — the number of cubes Senya has ($1 \le n \le 100$).

The second line contains a string of uppercase English letters — the name of Senya's sister. Its length doesn't exceed 100.

The following n lines contains 6 uppercase English letters each — the letters at Senya's cubes.

Output

The first line of output must contain "`YES`" if Senya can compose the name of his sister, or "`NO`" if he cannot.

If the answer is "`YES`", the second line must contain m distinct integers ranging from 1 to n, where m is the length of the sister's name. The i-th integer must be equal to the number of the cube that must be at the i-th position when composing the name.

Examples

standard input	standard output
4 ANN ANNNNN BCDEFG HIJKLM NOPQRS	NO
5 HELEN ABCDEF GHIJKL MNOPQL STUVWN EIUOZK	YES 2 5 3 1 4

Problem E. Domino Tiling

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

You are given a grid of size $n \times m$, some grid cells are already tiled by a square.

You must tile the free cells by domino tiles 1×2 or 2×1 that cost a dollars each, or 1×1 squares that cost b dollars each.

Find the minimum cost of tiling all the free cells.

Input

The first line of input contains four integers n, m, a, b ($1 \leqslant n, m \leqslant 100$, $|a| \leqslant 1\,000$, $|b| \leqslant 1\,000$).

Each of the following n lines contains m characters: "`.`" denotes an already tiled cell, "`*`" denotes a free one.

Output

Output one integer — the minimum cost of tiling all the free cells.

Example

standard input	standard output
2 3 3 2 .** .*.	5

Problem F. Birthday

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

There are m boys among Mitya's friends and n girls. He is planning to celebrate his birthday and going to invite some of his friends to the celebration.

For each of the boys among his friends he knows which girls he is a friend with. If a boy is the friend of a girl, he is her friend as well.

He would like to invite as many friends as possible, so that every boy at the celebration was a friend all of the invited girls.

Help Mitya to choose guests for the celebration!

Input

Each input contains one or more test sets. The first line of the input contains k — the number of test sets ($1 \leq k \leq 20$). The description of the test sets follows.

The first line of each description contains two integers m and n ($0 \leq m \leq 150$, $0 \leq n \leq 150$). The following m lines describe friendships, the i-th of these lines contains several integers — the numbers of the girls, friends of the i-th boy, followed by 0.

Output

For each test set output four lines.

The first one must contain the maximum number of friends Mitya can invite to his birthday. The second line must contain two integers: the number of boys to invite and the number of girls to invite.

The third line must contain the boys to invite. The fourth line must contain the girls to invite.

If there are several possible solutions, output any of them.

Example

standard input	standard output
2 2 2 1 2 0 1 2 0 3 2 1 2 0 2 0 1 2 0	4 2 2 1 2 1 2 4 2 2 1 3 1 2

Problem G. Relocation

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

There are n cities in Flatland connected by $n-1$ bidirectional roads. For each road its length is known. It is possible to get from any city to any other city by the roads.

There are m Edgeland spies in Flatland, the i-th spy is currently located in a city s_i, all s_i are distinct.

The spies want to be able to communicate. Two spies can communicate directly, if they are located in the same city, or if they are located in two cities connected by a road. Two spies u and v can communicate if they can communicate directly, or if there is a chain of spies $x_0 = u, s_1, x_2, \ldots, x_k = v$ such that for all i from 1 to k spies x_{i-1} and x_i can communicate directly.

You must relocate some spies, so that any two spies could communicate. After relocation there can be more than one spy in a city. But spies want to keep their movements secret. So you must come with such relocation plan that the maximal distance traveled by a spy is as small as possible.

Input

The first line of input contains an integer n — the number of cities ($1 \le n \le 50$).

The following $n-1$ lines describe roads. The i-th of these line contains three integers a_i, b_i, l_i — the cities connected by the i-th road and its length ($1 \le l_i \le 100\,000$).

The following line contains an integer m — the number of spies ($1 \le m \le n$). The following line contains m distinct integers $s_1, s_2, \ldots, s_m$ — the cities that the spies are initially located at.

Output

Output one integer: the minimum possible distance d such that each spy can move to some city at a distance of at most d from his initial location, and after the movement any two spies can communicate.

Example

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

Problem H. Postfix RLE

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Postfix notation is the way to represent arithmetic expression, in this notation operands are positioned before the operation signs. It allows to completely avoid using brackets.

Postfix notation for an expression that is a single variable or a number is just this variable or number. Postfix notation for $A \circ B$ (where A and B are some expressions and $\circ$ is the final operation to be performed) is represented as $AB\circ$. For example, postfix notation for $((a + f) \times b) \times (c \times d)$ is $af + b \times cd \times \times$.

You are given an expression in postfix notation that only uses variables and operations. All operations are binary, associative and commutative. That is, for any operation $\circ$ and any values A, B, C the following conditions are satisfied:

- $A \circ (B \circ C) = (A \circ B) \circ C$ *(associative)*.
- $A \circ B = B \circ A$ *(commutative)*.

Your task is to create equivalent postfix expression that has the same evaluation result based on associativity and commutativity such that its *block size* is minimal possible.

Block size of a string is the number of blocks of equal characters in it, for example, the block size of a string "$xx + yy + zz + \times\times$" is 7.

Input

The first and the only line of input contains an expression in postfix notation. It uses lowercase English letters to denote variables, and characters "`+`", "`*`", "`#`", "`!`", "`@`", "`$`", "`%`", and "`^`" to denote operations.

It is guaranteed that the expression is correct, it's not empty and its length doesn't exceed 2500.

Output

Output minimal possible block size of the equivalent expression.

Examples

standard input	standard output
`af+b*cd**`	`7`
`xy*x*y*x*y*`	`3`

Problem I. Graph Game

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Nick and Peter like to play the following game when attending their complexity theory lectures. They draw an undirected bipartite graph G on a sheet of paper, and put a token to one of its vertices. After that they make moves in turn. Nick moves first.

A move consists of moving the token along the graph edge. After it the vertex where the token was before the move, together with all edges incident to it, are removed from the graph. The player who has no valid moves loses the game.

You are given the graph that Nick and Peter have drawn. For each vertex of the graph find out who wins if the token is initially placed in that vertex. Assume that both Nick and Peter play optimally.

Input

The first line of the input file contains three integer numbers n_1, n_2, and m — the number of vertices in each part, and the number of edges, respectively ($1 \le n_1, n_2 \le 500$, $0 \le m \le 50\,000$). The following m lines describe edges — each line contains the numbers of vertices connected by the corresponding edge. Vertices in each part are numbered independently, starting from 1.

Output

Output two lines. The first line must contain n_1 characters, the i-th character must be '`N`' in case Nick wins if the token is initially placed in the i-th vertex of the first part, and '`P`' if Peter does. The second line must contain n_2 characters and describe the second part in the same way.

Example

standard input	standard output
3 3 5 1 1 1 2 1 3 2 1 3 1	NPP NPP

Problem J. Drawing Windows

Input:	`standard input`
Output:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Andrew writes portable mailer *KittenMail* for FTN technology networks. This mailer uses text-mode windowed interface like well-known Norton-style shells do. Andrew wrote the trivial code for *Mycrowslowed Widows* displaying the contents of the window buffer on the screen. But he was so surprised when he ran the mailer! Clearing the screen took about a second! Oops... What's wrong?

Andrew started to investigate this problem. Few minutes later, he discovered that any *Mycrowslowed Widows* system call accessing the screen buffer works a huge amount of time.

After hours of thinking, Andrew decided to make some improvements to his code. Now he wants to use *Widows*-specific function that draws a rectangle of characters instead of subsequent calls that display only one character. They perfectly work under other operating systems but are very slow under *Mycrowslowed Widows*. The evident task has arisen. The procedure which redraws window must display its visible parts using the minimal possible number of non-overlapping operations.

It is your task to write the corresponding code for one visible part of the window. Given the part of the rectangular window, write a program which determines the minimal number of non-overlapping rectangles covering this part and finds the optimal way of covering.

Input

The visible part of the window is given by its edge containing only horizontal and vertical segments with integer vertex coordinates. The part does not contain holes and its edge does not intersect or touch itself. Each segment has length of at least one character.

The first line of the input contains the number n of vertices on the edge of the visible part. The next n lines contain coordinates of vertices in counter-clockwise order (y coordinates grow downwards on the screen).

Horizontal and vertical segments alternate in the input. The last segment is drawn between the last and the first vertices. The value of n does not exceed 400 and absolute values of coordinates are limited to 200.

Output

In the first line output m — the minimal number of non-overlapping rectangular regions covering the visible part of the window.

The next m lines must contain one rectangle description each. The rectangle is specified by four numbers: minimal x, minimal y, maximal x and maximal y.

If there are several optimal solutions, output any of them.

Example

standard input	standard output
8 0 0 0 1 1 1 1 2 2 2 2 -1 1 -1 1 0	2 0 0 1 1 1 -1 2 2