Problem A. Maximum Matching

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 seconds
Memory limit:	256 megabytes

Your task is to find a maximum matching in the given bipartite graph.

Input

The first line contains two integers n and m ($1 \le n, m \le 250$) — the number of the vertices in the part A and in the part B.

The following n lines contain a description of the edges. The i-th vertex from A is described in the $(i+1)$-th line. Each line contains the vertices from B connected to the i-th vertex from A. Vertices in both parts are numbered independently. Each line ends with 0.

Output

The first line must contain integer l — the number of edges in maximum matching. Each of the following l lines must contain two integers u_j, v_j — the edges of the maximum matching.

Examples

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

Problem B. Dominoes

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.5 seconds
Memory limit:	256 megabytes

There is a $N \times N$ square chessboard. P squares were removed from the chessboard. Find out if it is possible to cover the remaining part of the chessboard with domino pieces (each piece is 2×1).

Put each domino piece on two neighboring cells exactly. No two pieces can cover the same cell.

Your task is to find the required tiling, if it exists.

Input

The first line contains two integer numbers N ($1 \le N \le 40$) and P ($0 \le P < N^2$) separated by a space.

Each of the following P lines contains a pair of numbers separated by a space — coordinates of the removed cell ($1 \le X_i, Y_i \le N$). The bottom left square has the coordinates $(1, 1)$, the bottom right square — $(N, 1)$.

Output

If the required covering exists, output "`Yes`" in the first line and "`No`" in the opposite case.

If the first answer is positive, then output in the second line integer number N_h — the number of horizontally placed pieces. Each of the following N_h lines should contain two integers — the coordinates of the left cell covered by a corresponding piece.

Output in the next line N_v — the number of vertically placed pieces. And the following N_v lines should contain the coordinates of the bottom cell covered by a corresponding piece.

if there are many solutions, you can print any of them.

Example

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

Problem C. Checking Program

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Your task is to check if the given matching is maximum in the given graph. If it is not true you need to find the matching which is larger than the given.

Input

The first line contains two integers m and n ($1 \le m, n \le 4\,000$) — the sizes of parts.

Each of the following m lines contains the list of edges going from the corresponding vertex in the first part. This list starts with integer K_i ($0 \le K_i \le n$) — the number of the edges followed by the vertices from the second part, connected to the corresponding vertex from the first part in arbitrary order. The total sum of all values K_i does not exceed 500 000.

The last line contains some maximum matching in given graph — m integers $0 \le L_i \le n$, where the i-th value is the vertex from the second part matched to the i-th vertex from the first part, or `0`, if the i-th vertex from the first part is unmatched.

Output

The first line must contain word "`YES`" if the given matching is maximum and "`NO`" in the other case.

If the given matching is not maximum, the second line must contain integer l — the number of the edges in a greater matching. Each of the following l lines must contain two integers u_j, v_j — edges from the greater matching.

Examples

standard input	standard output
3 2 2 1 2 1 2 1 2 1 2 0	YES
3 2 2 1 2 1 2 1 2 1 0 0	NO 2 1 1 2 2

Problem D. Minimum Vertex Cover

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

You are given a bipartite graph and its maximum matching. Your task is to find minimum vertex cover in this graph.

Input

The first line contains two integers n and m ($1 \le n, m \le 4\,000$) — the sizes of the parts.

Each of the following n lines contains the list of the edges which begin with the corresponding vertex from the first part. This list begins with the integer K_i ($0 \le K_i \le m$) — the number of the edges. After that the vertices from the second part connected to the vertex from the first part are given in an arbitrary order. The total sum of all the values K_i does not exceed 500 000.

The last line contains some maximum matching in this graph — n integers $0 \le L_i \le m$, where the i-th integer denotes the vertex from the second part matched to the i-th vertex from the first part, or `0`, if the i-th vertex from the first part is unmatched.

Output

Print in the first line the size of minimum vertex cover.

Print in the second line the integer S — the number of the vertices from the first part and after it print S integers — the vertices from the first part which belong to minimum vertex cover **in ascending order**.

Print in the third line the description of the vertices from the second part in the similar format.

Example

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

Problem E. Minimum Edge Cover

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 seconds
Memory limit:	256 megabytes

Your task is to find minimum edge cover in the given bipartite graph.

The set of edges R is an edge cover if and only if each vertex of the given graph is incident to at least one edge from R.

Isolated vertices are allowed, but should not be covered. In this problem edge cover is such a subset of edges that each non-isolated vertex is covered.

Input

The first line contains two integers n and m ($1 \le n, m \le 250$) — the number of vertices in the part A and the number of vertices in the part B.

The following n lines contain edges description. The i-th vertex from A is described in $(i + 1)$-th line. Each line contains numbers of the vertices from B connected to the i-th vertex from A. The vertices from parts are numbered independently. Each line ends with 0.

Output

The first line must contain integer l — the number of edges in the minimum edge cover. Each of the following l lines must contain two integers u_j, v_j — edges of the minimum edge cover. If there are multiple solutions, print any of them.

Isolated vertices are allowed, but should not be covered. In this problem edge cover is such a subset of edges that each non-isolated vertex is covered.

Examples

standard input	standard output
2 2 1 2 0 2 0	2 1 1 2 2
5 5 1 2 3 0 1 4 5 0 1 2 0 3 4 5 0 1 0	5 5 1 3 2 1 3 4 4 2 5

Problem F. Minimum Path Cover

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 seconds
Memory limit:	256 megabytes

You are given a directed acyclic graph (DAG). Your task is to find minimum path cover of this graph, i.e. to find minimum set of paths such that each of the vertices belongs to exactly one path. Paths can have lengths than equal to zero (i.e. a path can consist of only one vertex).

Input

The first line contains two integers n and m — the number of vertices and edges in the graph ($1 \le n \le 100$).

Each of the following m lines contains two integers — description of the edges.

It is guaranteed that graph is acyclic and it does not contain loops and multiple edges.

Output

Print in the first line the minimum number of paths. Then print each path one per line. If there are multiple solutions you are allowed to print any of them.

Examples

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

Problem G. Maximal flow

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given a directed graph G. Each edge has certain capacity. Find the maximal flow from vertex 1 to vertex n.

Input

The first line contains integers n and m ($2 \le n \le 500$, $1 \le m \le 10^4$) — the numbers of vertices and edges respectively. The next m lines describe the edges. Each edge is described by three numbers: the start vertex, the target vertex, and capacity. All capacities do not exceed 10^9.

Output

Print the value of the maximal flow between vertices 1 and n, followed by the amount of flow for each edge in the original order.

Examples

standard input	standard output
4 5 1 2 1 1 3 2 3 2 1 2 4 2 3 4 1	3 1 2 1 2 1
4 5 1 2 2 1 3 2 2 3 1 2 4 2 3 4 2	4 2 2 0 2 2
4 3 1 2 1 1 3 1 1 4 4	4 0 0 4

Problem H. Minimal Cut

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given an undirected graph. For the purpose of this problem, a $1-n$-*cut* is a set of edges S such that erasing all edges in S results in vertices 1 and n being disconnected. Find a $1-n$-cut with minimal total cost of edges.

Input

The first line contains integers n and m ($1 \le n \le 100, 0 \le m \le 400$) — the number of vertices and edges respectively. Next m lines describe the edges. An edge is described by indices of its endpoints, and its cost (a positive number not exceeding 10^8). It is guaranteed that each pair of vertices is directly connected by at most one edge.

Output

On the first line print the number of edges in a minimal cut and their total cost. On the second line print the indices of edges in the cut in ascending order. Edges are numbered in the order they are given in the input, starting from 1.

Example

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

Problem I. Chemistry

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Vasya has a rectangular $n \times m$ sheet of paper. In some of the cells Vasya has placed a character. Each character is '`H`', '`O`', '`N`', or '`C`'. Now Vasya wants to connect some adjacent pairs of characters with lines so that the picture shows a molecule. More specifically, the following conditions have to be satisfied:

- each line connects characters at cells sharing a side;
- each pair of characters has at most one line connecting them;
- each character is incident to the number of lines equal to the valency of the corresponding element (1 for H, 2 for O, 3 for N, and 4 for C).
- the empty cells are not connected with anything;
- at least one cell contains a character.

Note that the molecule is allowed to be disconnected. We also do not care if this molecule could actually exist in the real world, only the conditions above should be satisfied.

Input

The first line contains the dimensions of the sheet of paper n and m ($1 \le n, m \le 50$). The next n lines contain m characters each and describe the character layout; empty cells are denoted with '`.`'.

Output

Print '`Valid`' is it possible to draw lines to satisfy all conditions above, or '`Invalid`' otherwise.

Examples

standard input	standard output
3 4 HOH. NCOH OO..	Valid
3 4 HOH. NCOH OONH	Invalid

Problem J. Great Wall

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

King Ludovic has two sons who hate each other. He decided to separate his country into two pieces and give each to one of the sons. The king has given two cities A and B to his sons. Now he is going to raise a wall so that there is no way to get from city A to city B.

The country layout can be pictured as an $n \times m$ rectangle. Some cells of this rectangle contain impassable mountains, and all other cells are free to move on. Among the free cells, some of the cells are suitable to build a fragment of wall, and it is impossible to build on all others.

We assume that travelling through the country is only possible by going between free cells (that is, cells that do not contains mountains or fragments of walls) that share a side.

Input

The first line contains two integers n and m ($1 \le n, m \le 50$). The second line contains two integers k and l ($0 \le k, l, k + l \le mn - 2$) — the number of mountain cells, and the number of cells suitable for building. The next k lines describe the mountain cells; each of these lines contains two integers – the coordinate of a respective cell. The next l lines describe the cells suitable for building in the same format. The last two cells contain coordinates of the city A and the city B. It is guaranteed that all cells listed in the input are pairwise distinct. All coordinates satisfy $1 \le x \le n$, $1 \le y \le m$.

Output

On the first line print the smallest number F of wall fragments that the king needs to build so that the cities A and B are disconnected. On the next F lines print the coordinates of the wall fragments to build. If there are several possible solutions, you can output any of them.

If it is impossible to make the cities A and B disconnected, print -1.

Example

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

Problem K. Calculus

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A math professor is going to revise his calculus university course. There are n topics that the professor can cover. He assigned an integer value of usefulness to each topic (usefulness can be negative). Some of the topics require some other topics to be included in the course. If there is a cycle of dependencies between topics, it is both OK to include all topics in the cycle, or to include none of them.

The professor asked you to choose some of the topics to include the course so that their total usefulness is maximized.

Input

The first line contains the number of topics n ($1 \le n \le 200$). The second line contains n integer not exceeding 1 000 by absolute values — usefulness of each topic.

The next n lines describe dependencies between topics. i-th of these lines starts with an integer k_i — the number of dependencies for the topic i, followed by k_i integers — indices of topics that need to be included in the course together with i. The total number of dependencies is at most 1 800.

Output

Print a single integer — the maximum total usefulness of topics.

Example

standard input	standard output
4 -1 1 -2 2 0 1 1 2 4 2 1 1	2