Problem Set 7:
Network Flows
15-295 Fall 2018

A. Soldier and Traveling

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

In the country there are n cities and m bidirectional roads between them. Each city has an army. Army of the i-th city consists of a_i soldiers. Now soldiers roam. After roaming each soldier has to either stay in his city or to go to the one of neighboring cities by at **moving along at most one road**.

Check if is it possible that after roaming there will be exactly b_i soldiers in the i-th city.

Input

First line of input consists of two integers n and m ($1 \le n \le 100$, $0 \le m \le 200$).

Next line contains n integers $a_1, a_2, ..., a_n$ ($0 \le a_i \le 100$).

Next line contains n integers $b_1, b_2, ..., b_n$ ($0 \le b_i \le 100$).

Then m lines follow, each of them consists of two integers p and q ($1 \le p, q \le n, p \ne q$) denoting that there is an undirected road between cities p and q.

It is guaranteed that there is at most one road between each pair of cities.

Output

If the conditions can not be met output single word "NO".

Otherwise output word "YES" and then n lines, each of them consisting of n integers. Number in the i-th line in the j-th column should denote how many soldiers should road from city i to city j (if $i \ne j$) or how many soldiers should stay in city i (if $i = j$).

If there are several possible answers you may output any of them.

Examples

input Copy

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

output

```
YES
1 0 0 0
2 0 0 0
0 5 1 0
0 0 2 1
```

input Copy

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

output

```
NO
```

Problem B. Students Initiation

Time limit:	2 seconds
Memory limit:	256 megabytes

Soon the first year students will be initiated into students at the University of Berland. The organizers of the initiation come up with a program for this holiday. In their opinion, it would be good if the first-year students presented small souvenirs to each other. When they voiced this idea to the first-year students, they found out the following:

- some pairs of the new students already know each other;
- each new student agrees to give souvenirs only to those with whom they are already familiar;
- each new student does not want to present too many souvenirs.

The organizers have written down all the pairs of first-year friends who are familiar with each other and now want to determine for each new student, whom they should give souvenirs to. In their opinion, in each pair of familiar students *exactly one* student must present a souvenir to another student.

First year students already decided to call the unluckiest the one who will have to present the greatest number of souvenirs. The organizers in return promised that the unluckiest will be unlucky to the minimum possible degree: of course, they will have to present the greatest number of souvenirs compared to the other students, but this number will be as small as possible.

Organizers are very busy, and they asked you to determine for each pair of first-year friends who and to whom should present a souvenir.

Input

The first line contains two integers n and m ($1 \le n \le 5000$, $0 \le m \le \min(5000, n \cdot (n-1)/2)$) — the number of the first year students and the number of pairs of the students that know each other. The students are numbered from 1 to n.

Each of the following m lines contains two integers x_i, y_i ($1 \le x_i, y_i \le n$, $x_i \ne y_i$) — the students in each pair.

It is guaranteed that each pair is present in the list exactly once. It is also guaranteed that if there is a pair (x_i, y_i) in the list, then there is no pair (y_i, x_i).

Output

Print a single integer into the first line — the smallest number of souvenirs that the unluckiest student will have to present.

Following should be m lines, each containing two integers — the students which are familiar with each other. The first number in the pair must be the student that will present the souvenir to the second student in the pair.

Pairs can be printed in any order. If there are many solutions, print any of them.

Examples

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

Problem C
Apple Market

You are managing a market with some stores. The stores are arranged in an $n \times m$ grid. Each store sells apples. Apples cost exactly 1 Malaysian Ringgit per apple at every store.

There will be several customers who walk through this market. Each customer will only visit stores within a subrectangle of the market, and each customer has a fixed amount of money to spend. Also, each store has a limited inventory of apples, which will not be replenished between customers; the number available differs from store to store. Assuming you can control how many apples each store sells to each customer, what is the most money you can make?

Input

Each input will consist of a single test case. Note that your program may be run multiple times on different inputs. The first line of input will contain three space-separated integers n, m, and k, where the market has n rows and m columns ($1 \leq n, m \leq 50$), and there will be k ($1 \leq k \leq 10^5$) customers.

Each of the next n lines will have m integers a ($0 \leq a \leq 10^9$). This is a matrix in row major order, indicating the number of apples in the inventory of each store. $a[r, c]$ is the number of apples in the store in the r^{th} row, c^{th} column. The rows range from $1..n$ and the columns from $1..m$. The top left corner is $a[1, 1]$, and the bottom right corner is $a[n, m]$.

Each of the next k lines will describe a customer, with five integers: t, b ($1 \leq t \leq b \leq n$), l, r ($1 \leq l \leq r \leq m$), and x ($0 \leq x \leq 10^9$). The customer will only shop in the subrectangle from (t, l) to (b, r) inclusive (t=top, b=bottom, l=left, r=right). The customer has exactly x Malaysian Ringgits to spend.

Output

Output a single integer, representing the maximum amount of money to be made by controlling how many items each store sells to each customer.

Sample Input 1

```
2 3 2
1 2 3
4 5 6
1 2 2 3 20
2 2 1 3 15
```

Sample Output 1

```
20
```

D. Binary Tree on Plane

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

A root tree is a directed acyclic graph that contains one node (root), from which there is exactly one path to any other node.

A root tree is binary if each node has at most two outgoing arcs.

When a binary tree is painted on the plane, all arcs should be directed from top to bottom. That is, each arc going from u to v must meet the condition $y_u > y_v$.

You've been given the coordinates of all tree nodes. Your task is to connect these nodes by arcs so as to get the binary root tree and make the total length of the arcs minimum. All arcs of the built tree must be directed from top to bottom.

Input

The first line contains a single integer n $(2 \le n \le 400)$ — the number of nodes in the tree. Then follow n lines, two integers per line: x_i, y_i $(|x_i|, |y_i| \le 10^3)$ — coordinates of the nodes. It is guaranteed that all points are distinct.

Output

If it is impossible to build a binary root tree on the given points, print "-1". Otherwise, print a single real number — the total length of the arcs in the minimum binary tree. The answer will be considered correct if the absolute or relative error doesn't exceed 10^{-6}.

Examples

input	Copy
3 0 0 1 0 2 1	

output
3.650281539872885

input	Copy
4 0 0 1 0 2 1 2 0	

output
-1

E. Almost Permutation

3 seconds, 512 megabytes

Recently Ivan noticed an array a while debugging his code. Now Ivan can't remember this array, but the bug he was trying to fix didn't go away, so Ivan thinks that the data from this array might help him to reproduce the bug.

Ivan clearly remembers that there were n elements in the array, and each element was not less than 1 and not greater than n. Also he remembers q facts about the array. There are two types of facts that Ivan remembers:

- $1\ l_i\ r_i\ v_i$ — for each x such that $l_i \le x \le r_i\ a_x \ge v_i$;
- $2\ l_i\ r_i\ v_i$ — for each x such that $l_i \le x \le r_i\ a_x \le v_i$.

Also Ivan thinks that this array was a permutation, but he is not so sure about it. He wants to restore some array that corresponds to the q facts that he remembers and is very similar to permutation. Formally, Ivan has denoted the $cost$ of array as follows:

$cost = \sum_{i=1}^{n} (cnt(i))^2$, where $cnt(i)$ is the number of occurences of i in the array.

Help Ivan to determine minimum possible $cost$ of the array that corresponds to the facts!

Input

The first line contains two integer numbers n and q ($1 \le n \le 50$, $0 \le q \le 100$).

Then q lines follow, each representing a fact about the array. i-th line contains the numbers t_i, l_i, r_i and v_i for i-th fact ($1 \le t_i \le 2$, $1 \le l_i \le r_i \le n$, $1 \le v_i \le n$, t_i denotes the type of the fact).

Output

If the facts are controversial and there is no array that corresponds to them, print `-1`. Otherwise, print minimum possible $cost$ of the array.

input
3 0
output
3

input
3 1 1 1 3 2
output
5

input
3 2 1 1 3 2 2 1 3 2
output
9

input
3 2 1 1 3 2 2 1 3 1
output
-1

F. Oleg and chess

6.5 seconds, 256 megabytes

Oleg the bank client solves an interesting chess problem: place on $n \times n$ chessboard the maximum number of rooks so that they don't beat each other. Of course, no two rooks can share the same cell.

Remind that a rook standing in the cell (a, b) beats a rook standing in the cell (x, y) if and only if $a = x$ or $b = y$.

Unfortunately (of fortunately?) for Oleg the answer in this problem was always n, so the task bored Oleg soon. He decided to make it more difficult by removing some cells from the board. If a cell is deleted, Oleg can't put a rook there, but rooks do beat each other "through" deleted cells.

Oleg deletes the cells in groups, namely, he repeatedly choose a rectangle with sides parallel to the board sides and deletes all the cells inside the rectangle. Formally, if he chooses a rectangle, lower left cell of which has coordinates (x_1, y_1), and upper right cell of which has coordinates (x_2, y_2), then he deletes all such cells with coordinates (x, y) that $x_1 \le x \le x_2$ and $y_1 \le y \le y_2$. It is guaranteed that no cell is deleted twice, i.e. the chosen rectangles do not intersect.

This version of the problem Oleg can't solve, and his friend Igor is busy at a conference, so he can't help Oleg.

You are the last hope for Oleg! Help him: given the size of the board and the deleted rectangles find the maximum possible number of rooks that could be placed on the board so that no two rooks beat each other.

Input

The first line contains single integer n ($1 \le n \le 10000$) — the size of the board.

The second line contains single integer q ($0 \le q \le 10000$) — the number of deleted rectangles.

The next q lines contain the information about the deleted rectangles.

Each of these lines contains four integers x_1, y_1, x_2 and y_2 ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$) — the coordinates of the lower left and the upper right cells of a deleted rectangle.

If is guaranteed that the rectangles do not intersect.

Output

In the only line print the maximum number of rooks Oleg can place on the board so that no two rooks beat each other.

input
5 5 1 1 2 1 1 3 1 5 4 1 5 5 2 5 2 5 3 2 3 5
output
3

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

Here is the board and the example of rooks placement in the first example: