Problem A. The Game of SanGuoSha

The game of SanGuoSha is a well-know BoardGame in China recently. Your task is to simulate the process of a game, and to output the result of the game.

The following is the simplified rule of the game, and you need not consider other rules of this game.

There are 4 kinds of identities in a game:

1 Lord (Zhu Gong)

2 Knights (Zhong Chen)

4 Rebels (Fan Zei)

1 Provocateur (Nei Jian)

Each character except the Lord in a game has an HP of 3 or 4. The Lord has an HP of 4 or 5.

Players play in turns and start from player 1. In each turn:

1. The player gets 2 cards from the top of the deck
2. The player shows a Sha Card and discard that card, at the same time points out a living target player; the target player can either discard a Shan Card or lose 1 HP of his own. A player whose HP reaches zero is dead and out of the game forever.

   A player can show only 1 Sha Cards in a turn.

   Assume that if the target player has at least 1 Shan Card, the target player will discard a Shan Card in order not to lose his HP
3. The player then discards some cards so that the number of cards is less than or equal to his HP.

The following table shows the score each character gets at the end of a game.

	All rebels and provocateurs are dead	The Lord is dead and at least 1 character other than the provocateur is alive	The Provocateur kills the Lord after all other characters are dead
Lord	10 + 5×(number of living knights)	0	3
Knight	10 if alive; 5 if dead	0	0
Rebel	0	12 if alive; 8 if dead	0
Provocateur	15 if he is the 7th player to die; 0 otherwise	5−(number of living rebels) if alive; 0 otherwise	35

If any of the situation happens after a player's 3rd phase of his turn, the game ends immediately.

Input

The input file is smaller than 1MB.

For the first 8 lines, the i-th line shows the identity, HP, and the number of Sha and Shan Cards of Player i, for the identity, 1 represents the Lord, 2 represents a Knight, 3 represents a Rebel, 4 represents the Provocateur, and the deck is sufficient.

The next line shows N, the number of cards in the deck.

The next N numbers separated by spaces or newlines show the card in the deck from top to the bottom. 1 represents a Sha Card and 2 represents a Shan Card.

The next line contains M, the turns they have played.

The next M blocks explain the process of the game.

Each block contains three numbers separated by spaces or newlines.

The first number of a block is K, represents to whom the player showed the Sha Card to. $K = 0$ represents the player did not show a Sha Card.

The following 2 numbers: the number of Sha Card and the number of Shan Card discarded by that player.

Output

If the game is illegal (including wrong numbers of each characters, insufficient cards, etc), output "Wrong Input".

If the game ended at any time of the input, output the score of each player (separated by space), otherwise output "In Progress"

Sample input and output

stdin	stdout

`3 3 0 0`
`3 3 0 0`
`3 3 0 0`
`3 3 0 0`
`1 4 0 0`
`2 3 0 0`
`2 3 0 0`
`4 3 0 0`
`8`
`1 1 1 1 1 1 1 1`
`4`
`5 1 0`
`5 1 0`
`5 1 0`
`5 1 0`
	`12 12 12 12 0 0 0 1`

Problem B. Deer-Proof Fence

Uncle Magnus has planted some young saplings on his farm as part of his reforestation project. Unfortunately, deer like to eat tender sapling shoots and leaves, making it necessary to build protective fences around them. Since deer and other sapling nibblers can reach partway over the fence, every fence must lie at least a minimum distance (a margin) from each sapling.

Deer-proof fencing is quite expensive, so Uncle Magnus wants to minimize the total length of fencing used. Your job is to write a program that computes the minimum length of fencing that is required to enclose and protect the saplings. Fences may include both straight and curved segments. You may design a single fence that encloses all saplings or multiple fences that enclose separate groups of saplings.

Figure 1 shows two example configurations, each consisting of three saplings with different margin requirements. In the top configuration, which corresponds to the first sample input, the minimum-length solution consists of two separate fences. In the bottom configuration, which corresponds to the second sample input, the minimum-length solution consists of a single fence.

Input

The first line of each test case contains integers $N (0 < N \leq 10)$, which is the number of saplings, and $M (0 < M \leq 200)$, which is the margin required around each sapling. This line is followed by N additional lines. Each of these N lines contains two integers x and y that describe the Cartesian coordinates of a

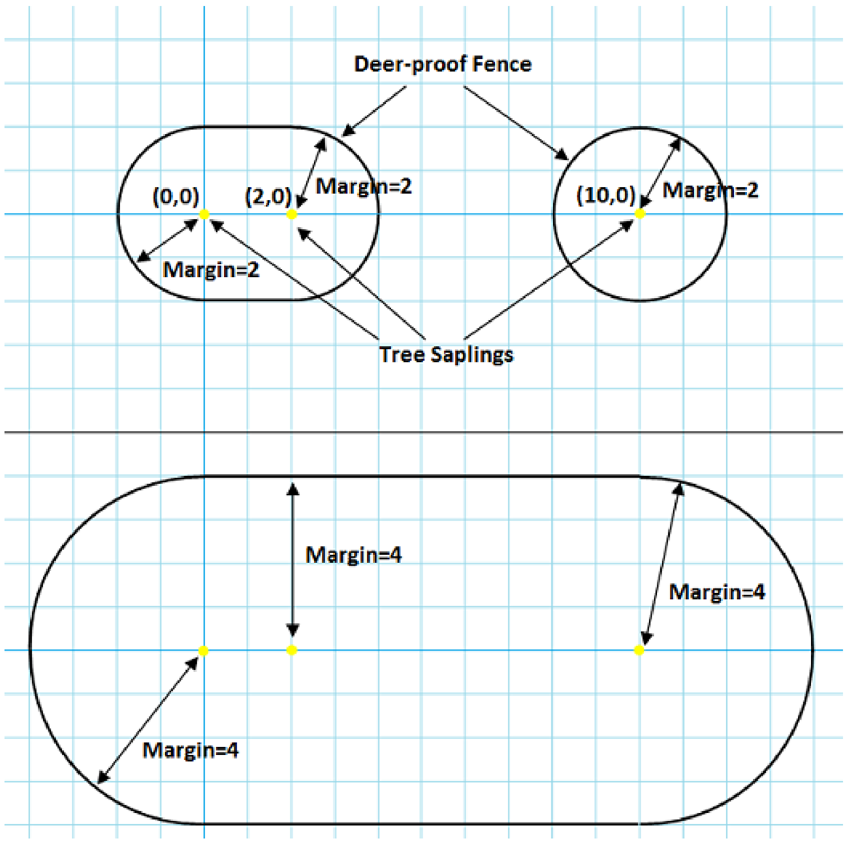

Figure 1: Deer-proof fences.

sapling ($|x| \le 100, |y| \le 100$). No two saplings are in the same location. For simplicity the saplings can all be considered as points and the thickness of deer-proof fences can be considered zero.

Output

For each test case, print the minimum total length of fencing required to protect the saplings with the given margin. Print the length with two digits to the right of the decimal point. Follow the format of the sample output.

Sample input and output

stdin	stdout

3 2
0 0
2 0
10 0
	length = 29.13

Problem C. Stupid Rabbit

Three stupid rabbits are playing in the grassland. They are playing on a number line, each occupying a different integer. In a single move, one of the outer rabbits jumps into the space between the other two. At no point may two rabbits occupy the same position.

Help them play as long as possible.

Input

Three integers A, B and $C (0 < A < B < C < 100)$, the initial positions of the rabbits.

Output

Output the largest number of moves the rabbits can make.

Sample input and output

stdin	stdout
`2 3 5`	`1`

Problem D. Intervals

You are given a set of intervals. A subset of the given set is valid only if the following two conditions are satisfied:

No two intervals in the subset intersect each other. Two intervals intersect each other if they both contain at least one common number.

Adding any remaining interval from the original set would make the subset invalid.

Output the number of distinct valid subsets of the given set. Two subsets are distinct if there is at least one interval in the first subset which is not in the second. Two intervals are distinct if their indexes are different. Correct result will always fit in 32-bit signed integer.

Input

The first line contains $N (1 \leq N \leq 200)$, the number of intervals.

Each of the next N lines contains two integers between 0 and 100, representing start point and the end point of an interval.

Output

An integer, represents the correct answer.

Sample input and output

stdin	stdout

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

Problem E. MST Counting

You are given a connected, weighted graph with N vertexes and M edges (self-loop and parallel edges are allowed), and you should find the number of different minimum spanning trees of the graph. Two trees are considered different if and only if their edge sets are different. Parallel edges are considered different.

Input

The first line: $N(N \leq 50000), M(M \leq 100000)$

The following M lines: x, y, w. To describe a edge which is connected to vertex x and vertex y with weight w. The number of edges wich have the same weight will not exceed 4.

Output

The answer mod 1000003 (A prime, as you know)

Sample input and output

stdin	stdout

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

Problem F. Scourage Invasion

The Lich King, you, has led a scourge region to invade Azeroth. There are two kinds of unit in your region: Death Knight and Lich. Now there are several Archmages in the battlefield, you want eliminated them all as soon as possible.

Lich has infinite mana, so it can cast Nova (A kind of powerful magic, damage 1 HP) any time it wants, but this Magic has a cooldown time *cd* (a positive integer). That means the spacing time between two successive Nova should be at least *cd*.

Death Knights mana are limited, so each Death Knight can only cast limited time of Death Coil (Another kind of powerful magic). The Death Coil can either target a ally (Heal 1 HP) or target a enemy (Damage 1 HP). The Death Coil has no cooldown time. That means a Death Knights can cast several Death Coil at the same time.

Any time you attack a Archmage, he will fight back immediately. A Lichs life is limited, so if a Lich attacks a Archmage, the result is that both of Lich and Archmage will take 1 HP damage at the same time. But Death Knights have infinite HP. He just ignored the reflected damage.

Some specification:

Once a Lichs HP reduce to 0, the Lich dies. Death Coil cant save a dead Lich.

A Lichs HP cant exceed its upper bound by healing.

If a unit with cast range R is at point (x, y), it can hit the unit at point (x', y') which $(x-x')^2+(x-y')^2 \leq R$ holds.

All of the coordinates and ranges are integer and will not be very large.

Units can stay at the same position.

Assume that at the very beginning (time 0), all of the magics are available.

Input

The first line: N_d (The number of Death Knight, $1 \leq N_d \leq 100$) N_l (The number of Lich, $1 \leq N_l \leq 100$) N_a (The number of Archmage, $1 \leq N_a \leq 100$) R_d (The cast range of Death Knight) R_l (The cast range of Lich).

The following N_d lines: x y k, means that there is a Death Knight positioned at (x,y) with k Death Coil. $1 \leq k \leq 1000$.

The following N_l lines: x y k t, means that there is a Lich positioned at (x,y) with k HP and cooldown time t. $1 \leq k \leq 1000, 1 \leq t \leq 10000$

The following N_a lines: x y k, means that there is a Archmage positioned at (x,y) with k HP. $1 \leq k \leq 1000$.

Output

The shortest time that The Scourge can eliminate all the Archmages. Output -1 if it can't happened.

Answer will not exceeded 1000000 (=10^6)

Sample input and output

stdin	stdout

`1 1 1 100 100`
`0 0 1`
`0 0 3 3`
`0 0 4`
	`6`

Problem G. NP=P?

TSP (Traveling Salesman Problem) is a well known NPC problem. The problem was first formulated as a mathematical problem in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, a large number of heuristics and exact methods are known, so that some instances with tens of thousands of cities can be solved.

TSP (Traveling Salesman Problem) can be modeled as a complete graph, such that cities are the graph's vertices, paths are the graph's edges, and a path's distance is the edge's length. A TSP tour becomes a Hamiltonian cycle, and the optimal TSP tour is the shortest Hamiltonian cycle. A Hamiltonian cycle (or Hamiltonian circuit) is a cycle in an undirected graph which visits each vertex exactly once and also returns to the starting vertex.

Now its your turn to solve this NPC problem. Output the length of the optimal TSP tour of the given graph.

Input

An integer N, indicating how many vertex are in the given graph ($3 \leq N \leq 10$)

An $N \times N$ symmetric matrix, the j-th element of the i-th row(≤ 100) represents the length between vertice i and vertice j.

Output

An integer, represents the answer to the problem.

Sample input and output

stdin	stdout

3
0 1 1
1 0 1
1 1 0
	3

Problem H. Chess Training

Alice and Bob are hobby chess players. They would like to become the best in the world so they are training a lot. Today they are going to play the following game with queens on a 100×100 chessboard to improve their offensive skills.

In each turn a player must choose one queen and move with it. From position (x, y) a queen can be moved to $(x - k, y)$ or $(x, y - k)$ or $(x - k, y - k)$, where $k > 0$. (Queens can move through squares containing other queens, and multiple queens can coexist on a single square). The two players alternate turns, and Alice goes first. The first player who moves any queen to the position $(0, 0)$ will become the winner.

You are given the initial positions of the queens, where (x_i, y_i) is the position of the i-th queen. Suppose Alice and Bob are playing optimally. Return "Alice will win" if Alice can win or "Bob will win" otherwise (all quotes for clarity).

Input

First line shows the number N of queens on the chessboard. ($0 < N \leq 50$)

The next 2 lines show the positions of the queens include $2N$ integers $x_0 \ldots x_{N-1}$ and $y_0 \ldots y_{N-1}$, separated by space. ($0 \leq x_i, y_i < 100$)

Output

One line includes "Alice will win" or "Bob will win"

Sample input and output

stdin	stdout

2
3 4
3 5
	Alice will win

Problem I. Messy Cubes

In the nearby kindergarten they recently made up an attractive game of strength and agility that kids love.

The surface for the game is a large flat area divided into $N \times N$ squares.

The children lay large spongy cubes onto the surface. The sides of the cubes are the same length as the sides of the squares. When a cube is put on the surface, its sides are aligned with some square. A cube may be put on another cube too.

Kids enjoy building forts and hiding them, but they always leave behind a huge mess. Because of this, prior to closing the kindergarten, the teachers rearrange all the cubes so that they occupy a rectangle on the surface, with exactly one cube on every square in the rectangle.

In one moving, a cube is taken off the top of a square to the top of any other square.

Write a program that, given the state of the surface, calculates the smallest number of moves needed to arrange all cubes into a rectangle.

Input

The first line contains the integers N and M ($1 \le N \le 100, 1 \le M \le N^2$), the dimensions of the surface and the number of cubes currently on the surface.

Each of the following M lines contains two integers R and C ($1 \le R, C \le N$), the coordinates of the square that contains the cube.

Output

Output the smallest number of moves. A solution will always exist.

Sample input and output

stdin	stdout

4 3
2 2
4 4
1 1
	2

Problem J. Snow White and Dwarfs

Snow White and the N dwarfs live in the forest. While the dwarfs mine away Snow White hangs around social networks.

Each morning the dwarfs form a long line and go whistling away to the mine. Snow White runs around them and snaps pictures to upload onto her favorite social network.

When dwarfs enter the mine, Snow White goes back to their house and goes through the pictures, selecting pretty ones. Each dwarf has a colored cap, and there are C different colors. A picture is pretty if more than half caps on it are of the same color. In other words, if there are K dwarfs on the picture, it is pretty if strictly more than $K/2$ dwarfs have same colored caps.

Write a program that will check for a set of M pictures if they are pretty, and what color is dominating if they are.

Input

First line contains two integers N and C ($3 \leq N \leq 300000, 1 \leq C \leq 10000$) number of dwarfs and number of colors.

Second line contains N integers between 1 and C (inclusive), colors of dwarves hats, ordered the way they formed the line that morning.

Third line contains M ($1 \leq M \leq 10000$), number of pictures.

Next M lines contain two integers A and B ($1 \leq A \leq B \leq N$). Each line describes one picture. On it there are all dwarves starting from A-th all the way to the B-th.

Output

Output M lines. For each picture output "no" if Snow White doesn't think the picture is pretty, and "yes X", where X is the color dominating on the picture, if she does.

Sample input and output

stdin	stdout

10 3
1 2 1 2 1 2 3 2 3 3
8
1 2
1 3
1 4
1 5
2 5
2 6
6 9
7 10
	no
	yes 1
	no
	yes 1
	no
	yes 2
	no
	yes 3

Problem K. Stamps

You have N stamps. Each stamp has a face value of A_i.

You can change the face value of one of the stamps. Your goal is to get the most denominations from the stamps after doing so.

For example, you have four stamps, with face values 1,3,4,4. If you do not change the face value, it can spell the face value of 1,3,4,5,7,8,9,11,12. Now if you change the face value of either of the stamps with the face value of 4, the face value of four stamps will be 1,3,9,4. It can form 1,3,4,5,7,8,9,10,12,13,14,16,17. You can get 13 different denominations in total.

Input

The first line contains an integer $N(2 \leq N \leq 100)$, the second line contains N integers $A_i(A_i \leq 7000)$, respectively, representing N kinds of face value.

Output

Output two integers P, Q. it means you change face value P to Q to get the most denominations. If there are multiple solutions, choose the one with the smallest P. if there are still multiple solutions, choose the one with the smallest Q.

Sample input and output

stdin	stdout
4 1 3 4 4	4 9