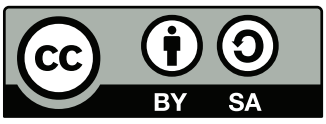
Preliminaries

for the Benelux Algorithm Programming Contest

Benelux Algorithm Programming Contest
2016

Problems

A	Block Game
B	Chess Tournament
C	Completing the Square
D	Hamming Ellipses
E	Lost In The Woods
F	Memory Match
G	Millionaire Madness
H	Presidential Elections
I	Rock Band
J	Target Practice
K	Translators' Dinner



A Block Game

You are attending the International Construction by Preschoolers Contest. Unfortunately, you are too old to participate, but you still enjoy watching the competition.

In between rounds, you are walking around the contest area when you see a toddler, one of the contestants, playing with her blocks. Annoyed that she is having all the fun, you decide to challenge her to a game.

You set up two stacks of blocks of a certain height. Then, you and the toddler take turns removing some number of blocks from the stack which contains the largest number of blocks (if both stacks have the same number of blocks, the current player can choose either stack to remove blocks from). The number of blocks removed must be a *positive multiple* of the number of blocks in the smaller stack. For instance, if there is a stack with 5 blocks, and one with 23 blocks, then the current player can remove 5, 10, 15 or 20 blocks from the stack of 23 blocks. The player who empties one of the stacks wins the game.

You have graciously decided to take the first move, but then a worry strikes you – might this devious preschooler still be able to beat you?

Input

One line with two integers N and M, satisfying $1 \leq N, M \leq 10^{18}$, the initial sizes of the two stacks of blocks.

Output

Output a single line containing a single word: the word "`win`" if you are guaranteed to win if you play correctly, and the word "`lose`" if your opponent can force you to lose.

Sample Input 1	**Sample Output 1**
`3 2`	`lose`

Sample Input 2	**Sample Output 2**
`3 3`	`win`

Sample Input 3	**Sample Output 3**
`5 2`	`win`

Sample Input 4	**Sample Output 4**
`5 3`	`win`

Sample Input 5	**Sample Output 5**
`13 10`	`lose`

This is not a blank page.

B Chess Tournament

Your friend is an organizer of the International Chess Playing Championship. He is worried that some of the contestants may be cheating, and he has asked you to help out. The chess players are allowed to report matches to the jury themselves, and this is not checked with the reported opponent. So, it is possible for competitors to make up matches and falsely report themselves as the winners.

Since chess is a game of skill, and not of chance, a player will *always* beat their opponent if their skill level is higher. A game will result in a draw if and only if the two players' skills are exactly equal.

However, the skill level of the players is not known. He has therefore asked you to write a program that, given a list of reported matches, determines whether this list is consistent or not. The list is inconsistent if we can determine that at least reported match is falsely reported, otherwise it is consistent.

Input

The first line contains two integers N ($2 \leq N \leq 50\,000$) and M ($1 \leq M \leq 250\,000$), to describe a championship with N players and M reported matches.

The following M lines each consist of an integer K, a symbol which is either '=' or '>', and another integer L. The integers K and L each uniquely identify a player ($0 \leq K, L < N$). If the symbol is '=', then the game between K and L was a draw. If the symbol is '>', then K beat L in a match.

You may assume that there is at most one reported match between any given pair of players. Also, each player takes part in at least one reported match.

Output

Output a single line containing a single word: "`consistent`" if the list of recorded matches is consistent, and "`inconsistent`" if it is not.

Sample Input 1	**Sample Output 1**
3 3 0 > 1 1 = 2 0 = 2	inconsistent

Sample Input 2

```
5 5
0 = 1
1 = 2
3 = 4
0 > 3
1 > 4
```

Sample Output 2

```
consistent
```

Sample Input 3

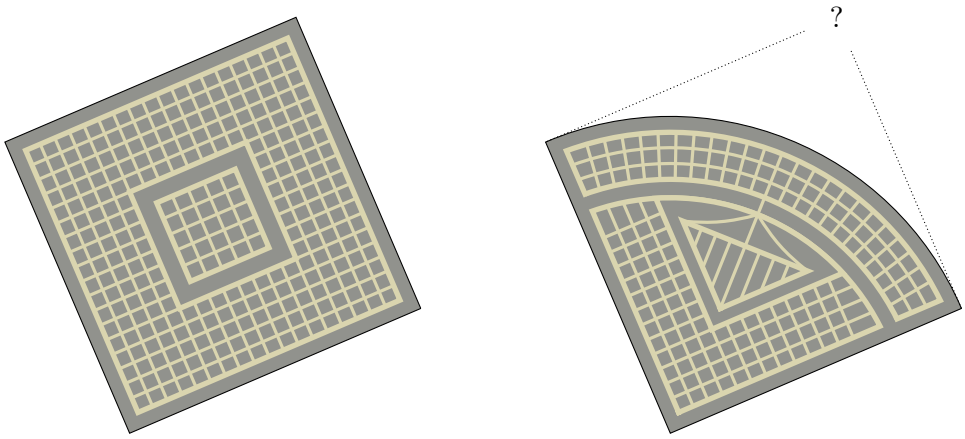
```
6 5
0 > 1
1 > 2
3 = 4
4 = 5
5 > 3
```

Sample Output 3

```
inconsistent
```

C Completing the Square

In the heart of your home city, there is an old square, close to the train station, appropriately called *Station Square*. It used to look like a perfect square: four sides of equal length joined by right angles. However, it hasn't looked like this for decades, as one of the four corners was destroyed by bombings in the Second World War. After the war, the square was rebuilt as a quarter circle, and it has looked like that ever since. (In other words, it looks like an isosceles right triangle, except that the hypothenuse is not a straight line but a circular arc.) This is illustrated in the figure below, which corresponds with Sample Input 1.



(a) The old square, in the shape of an actual square.

(b) The current square, looking more like a quarter circle.

Recently, the city counsil voted to completely remodel the train station and its surroundings, which includes restoring Station Square to its original state. Therefore they need to determine the exact location of the fourth corner. This task is too complicated for ordinary aldermen, so the city decided to hire a top scientist. That would be you! Please help the city complete the square, and you will be greatly rewarded!

Input

There are three lines of input. Each line contains two integers denoting the x and y coordinates of one of the corners of the current square ($-10^4 \leq x, y \leq 10^4$).

Output

Output one line with two space-separated integers denoting the x and y coordinates of the long-lost fourth corner.

Sample Input 1	**Sample Output 1**

`2 -5`
`-8 -1`
`-5 -8`
	`-1 2`

Sample Input 2

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

Sample Output 2

```
0 1
```

Sample Input 3

```
2 -1
-2 3
2 7
```

Sample Output 3

```
6 3
```

D Hamming Ellipses

In geometry, ellipses are defined by two focal points f_1, f_2 and a length D. The ellipse consists of all points p such that distance(f_1, p) + distance$(f_2, p) = D$.

When one normally thinks of ellipses, it is in the context of the Euclidean 2D plane, with the Euclidean distance as a distance measure.

This problem explores a different kind of ellipse. The space we will work in is the space of words of length n using an alphabet of q different symbols, often denoted as F_q^n. As you can probably see, for given values of q and n, there are q^n different words (points) in the space F_q^n.

For a distance measure, we use the *Hamming distance*. The Hamming distance between two words $x, y \in F_q^n$ is simply the number of positions where the symbols that make up the words x and y differ. For example, the Hamming distance between words `01201` and `21210` is 3 because there are 3 positions where the words have different symbols. The Hamming distance between any two words in F_q^n will always be an integer between 0 and n, inclusive.

Within the space F_q^n, we now define the *Hamming ellipse* as the set of all points p such that hammingdistance(f_1, p) + hammingdistance$(f_2, p) = D$. Given values q and n, focal points f_1 and f_2 and distance D, we ask you to determine the number of points $p \in F_q^n$ that lie on this Hamming ellipse.

Input

The first line contains three integers q ($2 \leq q \leq 10$), n ($1 \leq n \leq 100$) and D ($1 \leq D \leq 2n$).

The second and third lines specify the two focal points f_1 and f_2, each formatted as a string of length n using digits $\{0, 1 \ldots q-1\}$.

Output

Output one line containing a single integer, denoting the number of points on the ellipse.

The input is chosen such that the answer is less than 2^{63}.

Sample Input 1	**Sample Output 1**

`3 5 9`
`01201`
`21210`
	`24`

Sample Input 2	**Sample Output 2**

`4 6 5`
`123031`
`231222`
	`0`

Sample Input 3	**Sample Output 3**
2 32 32 01010101010101010101010101010101 01010101010101010101010101010101	601080390

E Lost In The Woods

Your friend has gotten lost in the woods. He has called and asked for you to come get him, but you are very busy and would rather just stay home. You quickly look up a map of the woods. It appears that the woods consist of a small number of clearings, with paths connecting them. You hope that the woods are sufficiently small and simple that your friend can get out easily, even if he is just randomly running around.

From your friend's description, you can figure out at which clearing he is. Assuming that every time he reaches a clearing, he runs in a uniformly random direction (including back the way he came), and that it takes him exactly one minute to get from clearing to clearing, can you predict how long it will take him to get out on average?

Input

The first line contains two integers N and M, where N is the number of clearings in the woods ($2 \leq N \leq 20$), and M is the total number of paths between clearings. The clearings are numbered 0 through $N-1$, such that clearing 0 is the one where your friend is right now and clearing $N-1$ is the exit of the woods.

The next M lines each contain two integers K and L, indicating a path between clearing K and clearing L ($0 \leq K, L < M$).

You may assume that it is possible for your friend to reach the exit by following paths, that paths do not cross, and that there is at most one path between any two clearings.

Output

Output a single line containing a single number: the expected value of the number of minutes it will take your friend to get out of the woods.

Your answer may have an absolute error of at most 10^{-5}.

Sample Input 1	**Sample Output 1**
3 3 0 1 1 2 0 2	2.000000

Sample Input 2	**Sample Output 2**
5 6 0 1 0 2 1 2 2 4 0 3 3 4	6.727273

Sample Input 3	**Sample Output 3**
4 4 0 1 1 3 3 0 0 2	3.333333

F Memory Match

You are playing the game "Memory Match".

This game revolves around a set of N picture cards. The cards are organized in pairs: there are $N/2$ different pictures, each picture occurring on exactly two cards.

At the beginning of the game, the cards are shuffled and laid face down on the table. Players then take turns in guessing a pair of cards with the same picture. Each turn consists of picking a face-down card and turning it over to reveal its picture, then picking another face-down card and turning that card over as well. If the pictures on the two turned cards are identical, the cards remain face-up, the player scores a point and may take another turn. If the pictures are different, both cards are turned face-down again and the turn goes to the next player.

It is now your turn! Given a description of all previous actions in the game, pick as many matching pairs as possible.

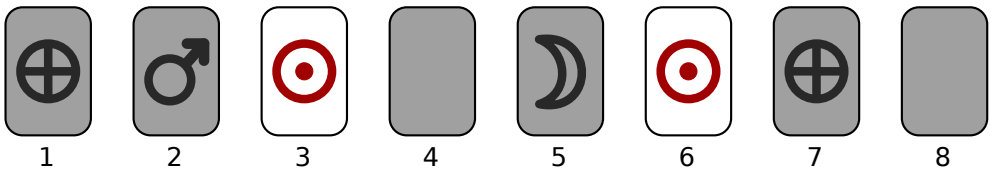


Figure 1: Illustration of the first example input. Only cards 3 and 6 have been matched correctly, all other cards are face-down. How many pairs can you score?

Input

The first line contains an even integer N, the number of cards on the table ($2 \leq N \leq 1000$).

The second line contains an integer K, the number of turns played thus far in the game ($0 \leq K \leq 1000$).

The following K lines each describe a turn. A turn is described by integers C_1 and C_2 followed by words P_1 and P_2. The numbers C_1 and C_2 refer to card positions on the table ($1 \leq C_1, C_2 \leq N$ and $C_1 \neq C_2$). The words describe the pictures on the two selected cards. Each word consists of between 1 and 20 lowercase letters in range 'a' ... 'z'. If $P_1 = P_2$, the two cards stay face-up and the corresponding positions C_1 and C_2 may not be chosen again.

The input is such that at least two cards are still in face-down position.

Output

Output one line with an integer S, the number of matching pairs you can score with certainty.

Sample Input 1

```
8
5
1 3 earth sun
2 6 mars sun
6 3 sun sun
7 5 earth moon
2 7 mars earth
```

Sample Output 1

```
3
```

Sample Input 2

```
10
6
1 2 moon earth
9 10 venus sun
8 7 moon venus
1 8 moon moon
4 10 sun sun
9 6 venus mars
```

Sample Output 2

```
3
```

Sample Input 3

```
8
2
1 3 moon earth
2 6 sun earth
```

Sample Output 3

```
1
```

G Millionaire Madness

A close friend of yours, a duck with financial problems, has requested your help with a matter that will help him pay off his debts. He is the nephew of an extremely wealthy duck, who has a large vault, filled with mountains of coins. This wealthy duck has a certain coin in his possession which has a lot of sentimental value to him. Usually, it is kept under a protective glass dome on a velvet cushion.

However, during a recent relocating of the coins in the vault, the special coin was accidentally moved into the vault, leading to an extremely stressful situation for your friend's uncle. Luckily, the coin has recently been located. Unfortunately, it is completely opposite to the entrance to the vault, and due to the mountains of coins inside the vault, actually reaching the coin is no simple task.

He is therefore willing to pay your friend to retrieve this coin, provided that he brings his own equipment to scale the mountains of coins. Your friend has decided he will bring a ladder, but he is still uncertain about its length. While a longer ladder means that he can scale higher cliffs, it also costs more money. He therefore wants to buy the shortest ladder such that he can reach the special coin, so that he has the largest amount of money left to pay off his debts.

The vault can be represented as a rectangular grid of stacks of coins of various heights (in meters), with the entrance at the north west corner (the first height in the input, the entrance to the vault is at this height as well) and the special coin at the south east corner (the last height in the input). Your avian companion has figured out the height of the coins in each of these squares. From a stack of coins he can attempt to climb up or jump down to the stack immediately north, west, south or east of it. Because your friend cannot jump or fly (he is a very special kind of duck that even wears clothes), successfully performing a climb of n meters will require him to bring a ladder of at least n meters. He does not mind jumping down, no matter the height; he just lets gravity do all the work.

Input

The first line contains two integers: the length M, and the width N of the vault, satisfying $1 \leq M, N \leq 1000$.

The following M lines each contain N integers. Each integer specifies the height of the pile of coins in the vault at the corresponding position. (The first line describes the north-most stacks from west to east; the last line describes the south-most stacks from west to east). The heights are given in meters and all heights are at least 0 and at most 10^9 (yes, your friend's uncle is very rich).

Output

Output a single line containing a single integer: the length in meters of the shortest ladder that allows you to get from the north west corner to the south east corner.

Sample Input 1

```
3 3
1 2 3
6 5 4
7 8 9
```

Sample Output 1

```
1
```

Sample Input 2

```
1 4
4 3 2 1
```

Sample Output 2

```
0
```

Sample Input 3

```
7 5
10 11 12 13 14
11 20 16 17 16
12 10 18 21 24
14 10 14 14 22
16 18 20 20 25
25 24 22 10 25
26 27 28 21 25
```

Sample Output 3

```
3
```

H Presidential Elections

In a few weeks time, a new president will be elected in the country of Marecia. There are two political parties: the Constituents and the Federals. Each party has one presidential candidate; one of these will become the new president.

Picture by David Hill via ICPCNews.com

This year you are more involved than ever before: you are the candidate for the Constituents! With only a few weeks to go, you decide to take stock of the situation. Instead of touring blindly through the country, you feel that it is better to focus your attention on certain particular states.

The Marecian electoral system is quite complicated. Every registered voter will vote for exactly one of the two candidates, but the totals are not calculated nationwide. Instead, every state has a fixed number of delegates. After the popular vote is in, the delegates from all states travel by foot to the capital of Marecia, the city of Ashwington. One final round of voting is done in Ashwington: every delegate gets to vote for one of the two candidates. A delegate is required by law to vote for the candidate who received the majority of the votes in his own home state.

Your campaign team has painstakingly compiled a very accurate list of voting preferences per state. Therefore you now have the following information about every state:

- The number of voters who will definitely vote for the Constituents;
- The number of voters who will certainly vote for the Federals;
- The number of voters who have not made up their minds yet.

Voters from the first two categories are not susceptible to any further campaigning, so your focus for the remaining weeks is on people of the third category. Specifically, you would like to know the minimum number of people you still have to convince in order to secure a victory.

If there is a tie on either state or national level, the law states that it is resolved in favour of the oldest of the two political parties, which unfortunately is the Federal party.

Input

The first line contains a single integer S, the number of states.

The following S lines each contain four integers D_i, C_i, F_i, U_i, where

- D_i denotes the number of delegates for the i-th state,
- C_i denotes the number of registered voters in the i-th state who will definitely vote for the Constituents,
- F_i denotes the number of registered voters in the i-th state who will certainly vote for the Federals,
- U_i denotes the number of undecided voters in the i-th state.

There are at most 2016 delegates in total, and the total number of registered voters is at most 10^9. Every state has at least one delegate and at least one registered voter. There is at least one state.

Output

Output one line with a single integer: the minimum number of voters you have to convince to secure a victory. If it is not possible for you to win the election, output "`impossible`" instead.

Sample Input 1	**Sample Output 1**
3 7 2401 3299 0 6 2401 2399 0 2 750 750 99	50

Sample Input 2	**Sample Output 2**
3 7 100 200 200 8 100 300 200 9 100 400 200	impossible

Sample Input 3	**Sample Output 3**
3 32 0 0 20 32 0 0 20 64 0 0 41	32

I Rock Band

Every day after school, you and your friends get together and play in a band. Over the past couple of months, the band has been rehearsing a large number of songs. Now it's time to go out and perform in front of a crowd for the first time. In order to do so, a set list for the concert needs to be determined.

Picture by Nihad Qulanzade via Wikimedia Commons

As it turns out, every band member has a different taste in music. (Who would have thought?) Everybody is very picky: a band member doesn't want to play any particular song X unless he also gets to play all songs he likes better than song X. This holds for every band member and for every song X. Furthermore, obviously at least one song should be performed.

The organisers of the concert do not want you to play too many songs, so a selection needs to be made that is as small as possible. As the unofficial leader of the band, you have taken it upon yourself to find a minimum length set list that meets the requirements.

Input

The first line contains two integers M and S, satisfying $M \geq 1$ and $S \geq 1$ as well as $M \times S \leq 10^6$. These denote the total number of band members and the number of songs, respectively.

The following M lines each contain S integers per line, where the i-th line denotes the preference list of the i-th band member, starting with his favourite song and ending with his least favourite song. The songs are numbered 1 through S.

No two band members have the exact same preference lists.

Output

Output the smallest possible set list, using the following format:

- One line with an integer L: the length of the smallest possible set list.
- One line with L space-separated integers, denoting a sorted list of the songs to be played.

Sample Input 1

```
3 8
4 5 2 1 6 8 3 7
5 2 4 8 6 1 3 7
2 5 4 8 1 6 3 7
```

Sample Output 1

```
3
2 4 5
```

Sample Input 2

```
2 8
6 2 8 7 1 3 4 5
2 8 7 1 3 4 5 6
```

Sample Output 2

```
8
1 2 3 4 5 6 7 8
```

Sample Input 3

```
6 3
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
```

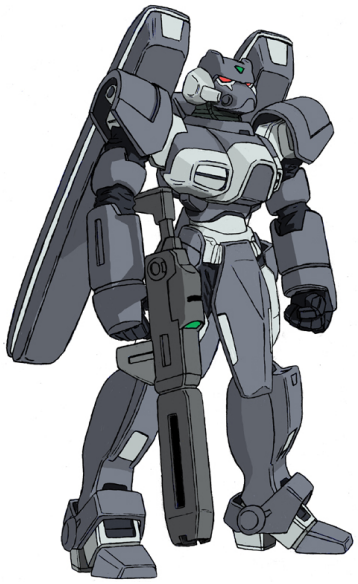
Sample Output 3

```
3
1 2 3
```

J Target Practice

You are a newly designed and created robot. As a robot, you are excellent at shooting lasers: your lasers always go perfectly straight, go infinitely far, and are infinitely thin. To test you, the scientist who made you set out a number of targets for you to hit with your laser beam. The targets (point-like objects) are set up in a large, open room.

Unfortunately, you are running low on battery. You only have enough charge left to fire two shots. The targets are transparent, so you might be able to shoot multiple targets with your laser beam. In fact, you are able to hit an infinite number of targets with a single shot, as long as they are on a straight line. In addition, you can move anywhere before and between the two shots. Can you figure out if it is possible to hit all targets with at most two shots from your laser beams?



Picture by Darcad via DeviantArt

Input

The first line contains an integer N, satisfying $1 \leq N \leq 100\,000$, the number of targets.

The following N lines each contain two integers X_i and Y_i, satisfying $-10^9 \leq X_i, Y_i \leq 10^9$. Each pair (X_i, Y_i) specifies the coordinates of one of the N targets. No two targets are placed at the same coordinates.

You may assume that height does not play a role in this problem.

Output

Output a single line containing a single word: "`success`" if it is possible to line up the two shots so that they hit all the targets, and "`failure`" if it is not.

Sample Input 1	**Sample Output 1**
6 -1 0 0 0 1 0 -1 1 0 2 1 1	failure

Sample Input 2	**Sample Output 2**
6 1 1 3 5 0 -1 1 0 5 0 0 0	success

K Translators' Dinner

It is time again for the annual International Convention for Phonetic Communication. Because there are attendees from all over the world, and they do not all speak the same languages, the organizers have hired translators to help out.

To thank the translators at the end of the conference for their hard work, the organizers want to arrange a dinner at a nice local restaurant. However, the restaurant only has small, two person tables, hence the translators will have to be divided into pairs. As the organizers would like the translators to have a nice evening, they prefer that two translators sitting at the same table are both able to speak the same language. Write a program to help the organizers determine a way to match up the translators in pairs, so that each of the translators speaks a language that the other also speaks.

Input

The first line contains two numbers N and M, the number of languages spoken at the convention, and the number of hired translators respectively ($2 \leq N \leq 100$, $1 \leq M \leq 200$).

The following M lines each describe a translator. Each of these lines contains two integers specifying the two languages that the translator speaks. Languages are identified by integers in the range $[0, N-1]$.

Translators are identified by integers in the range $[0, M-1]$. Translators are listed in order of increasing identifier (i.e. the first listed translator has identifier 0).

There are no two translators who both speak the same two languages. Translators have been chosen such that any two languages spoken at the conference can be translated into one another, although it may take several translators.

Output

If it is possible to match up all translators such that each pair speaks a common language, output a possible matching: print $M/2$ lines, each line containing the two identifiers of a pair of matched translators. The pairs, as well as the translators within a pair, may be listed in any order.

There may be multiple possible matchings. In that case, print any one of them.

If it is not possible to match up all translators, output a single line containing the word "`impossible`".

Sample Input 1

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

Sample Output 1

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

Sample Input 2

```
3 3
0 1
1 2
2 0
```

Sample Output 2

```
impossible
```