BAPC 2026 Preliminaries

Preliminaries for the
2026 Benelux Algorithm Programming Contest

Problems

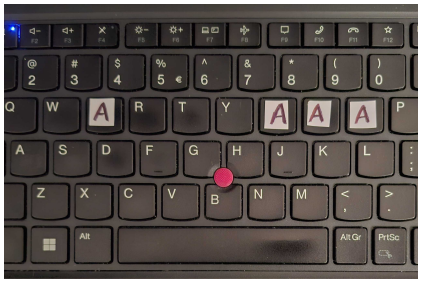
A	Ambagaata
B	Butter, Cheese, and Eggs
C	Chaotic Crossing
D	Dicey Decision
E	Exhausted Economist
F	Football Forecast
G	Grabbing Goods
H	Handing Over
I	Irritating Interruption I
J	Joyful Jogging
K	Khoikhoi Dralinpome
L	Lock In
M	Manic Ceramic

Copyright © 2026 by The BAPC 2026 Jury. This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.
https://creativecommons.org/licenses/by-sa/4.0/

A Ambagaata

Time limit: 4s

Anthonio (also known as Anthanaa) thinks it is funny to replace all vowels with the letter ‘a’. When he types, he consistently replaces any ‘a’, ‘e’, ‘i’, ‘o’, ‘u’ with an ‘a’. Any ‘y’ is either kept or replaced by an ‘a’ arbitrarily with no consistency.



Anthanaa asas tha QWARTY kaabaard.

You are given a dictionary of words of lowercase letters and a list of messages by Anthonio with only lowercase letters and spaces. Each word in the dictionary contains the letter ‘y’ at most four times.

Determine for each message whether the message is possibly consistent with the dictionary, and if so, whether it is ambiguous. If the message is consistent with the dictionary and not ambiguous, reconstruct Anthonio’s intended message.

Input

The input consists of:

- One line with two integers n and q ($1 \leq n \leq 50\,000$, $1 \leq q \leq 10^3$), the number of words in the dictionary and the number of messages you need to analyze.
- n lines, each with a string w ($1 \leq |w| \leq 100$), a word in the dictionary. Each word in the dictionary only consists of lowercase English letters (`a-z`), at most four of which are ‘`y`’.
- q lines, each with an integer m ($1 \leq m \leq 10^5$), followed by m strings s ($1 \leq |s| \leq 100$), describing a message. Each word in the messages only consists of lowercase English letters (`a-z`), at most four of which are ‘`y`’, and the vowels ‘`e`’, ‘`i`’, ‘`o`’, and ‘`u`’ do not occur.

All words in the dictionary are distinct. The total number of letters in the n words in the dictionary is at most $2 \cdot 10^5$. The total number of letters in the q messages is at most $5 \cdot 10^5$.

Output

For each message, if there is a unique message consisting of words in the dictionary that Anthonio could have meant, output “`possible`” followed by Anthonio’s intended message. Otherwise, if there are multiple messages that Anthonio could have meant, output “`ambiguous`”. Otherwise, output “`impossible`”.

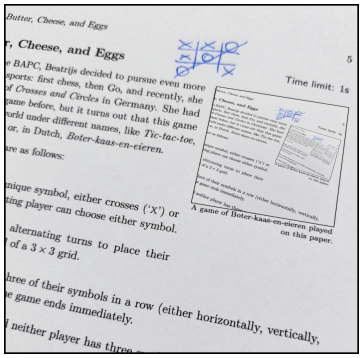
Sample Input 1	**Sample Output 1**
7 3 hello world i have a yoyo lmao 2 halla warld 4 a hava a yaya 2 aaay lmaa	possible hello world ambiguous impossible

Sample Input 2	**Sample Output 2**
3 3 aayayy ayyaay yyyyaa 1 ayaaay 1 ayaaya 1 aayaay	possible ayyaay impossible ambiguous

B Butter, Cheese, and Eggs

Time limit: 1s

After winning the BAPC, Beatrijs decided to pursue even more demanding mind sports: first chess, then Go, and recently, she learned the game of *Crosses and Circles* in Germany. She had never heard of this game before, but it turns out that this game is known across the world under different names, like *Tic-tac-toe*, *Noughts and Crosses*, or, in Dutch, *Boter-kaas-en-eieren*.



A game of Boter-kaas-en-eieren played on this paper.

The rules of the game are as follows:

- Each player has a unique symbol, either crosses ('X') or circles ('O'). The starting player can choose either symbol.
- The two players take alternating turns to place their symbol in an empty cell of a 3×3 grid.
- The first player to place three of their symbols in a row (either horizontally, vertically, or diagonally) wins, and the game ends immediately.
- If all nine cells are filled and neither player has three symbols in a row, the game is a draw.

To improve her skills, Beatrijs has been watching the world championship games of crosses and circles, and one game in particular stuck with her: she remembers a board state where x crosses and o circles were played, but she can recall neither the exact sequence of moves nor the final board state.

On more careful thought, Beatrijs is not even sure if the game was valid at all, and she wonders if it is even possible for a valid game to have x crosses and o circles.

Input

The input consists of:

- One line with two integers x and o ($0 \leq x, o \leq 9$), the number of crosses and circles.

Output

If it is possible to construct a valid board state, output 3 strings of 3 characters, each character being either 'O' for a circle, 'X' for a cross, or '.' for an empty cell. Otherwise, output "impossible".

If there are multiple valid solutions, you may output any one of them.

Sample Input 1	**Sample Output 1**
1 2	OX. .O. ...

Sample Input 2	**Sample Output 2**
3 4	OXO XO. O.X

Sample Input 3	**Sample Output 3**
5 3	impossible

Sample Input 4	**Sample Output 4**
1 0	... .X. ...

Sample Input 5	**Sample Output 5**
6 7	impossible

C Chaotic Crossing

Time limit: 4s

Public domain, SVG by Bouwe Brouwer on Wikimedia Commons

You just arrived at the train station in Leiden, and you would like to walk to the location of the BAPC. However, exiting the station, you encounter a massive intersection with cars, buses, trams, and bicycles coming from all directions! Since the BAPC is at the other side of this intersection, you need to find a safe way to cross. The intersection consists of n patches of pavement where you can safely stand still and m bidirectional zebra crossings. Each zebra crossing has its own traffic light: you can only walk across it when the light is green. Each second, you can either wait on the patch of pavement you are standing on, or you can walk across any connected zebra crossing where the light is currently green. The traffic lights behave randomly: each traffic light has a parameter p, and each second, the traffic light is green with probability p, and red with probability $1 - p$, completely independently from all other traffic lights.

Since you want to be on time for the BAPC, you want to figure out the minimal expected time needed to cross the intersection, starting from the train station. The train station is located next to the pavement patch numbered 1, and you need to go to the pavement patch numbered n.

Input

The input consists of:

- One line with two integers n and m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 4 \cdot 10^5$), the number of patches of pavement and the number of zebra crossings.
- m lines, each containing three integers u, v, and p ($1 \le u < v \le n$, $1 \le p \le 100$), indicating that there is a bidirectional zebra crossing between patch u and patch v, with a p% chance of being green each second.

There is at most one zebra crossing between each pair of patches of pavement.

Output

Output the expected time to cross the intersection. If you cannot reach the BAPC, output "`impossible`".

Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
2 1 1 2 50	2

Sample Input 2	**Sample Output 2**
4 4 1 4 1 1 2 80 2 3 90 3 4 70	3.754898468

Sample Input 3	**Sample Output 3**
4 5 1 2 50 1 3 50 2 3 90 2 4 33 3 4 67	3.007172

Sample Input 4	**Sample Output 4**
3 1 1 2 100	impossible

D Dicey Decision

Time limit: 2s

CC BY 2.0 by GMG Creative on Flickr

You and three friends have been playing the board game known as *Boring Activity of Pure Chance* (BAPC), a variant of *Snakes and Ladders*. BAPC takes place on a board with n spaces numbered 1 through n. All players start on space 1, and they take turns rolling a die to determine how many spaces they advance. For example, if you are located on space 4 and you roll a 3, then you advance 3 spaces, landing you on space 7. If your roll has you advance beyond space n, you simply land on space n instead. The first player to reach the finish at space n wins.

The board also contains m shortcuts. Each shortcut has an entrance space a and an exit space b, and landing on the entrance space of a shortcut sends you to its exit space. The exit space can be either ahead of or behind the entrance space, either sending you forwards closer to the finish, or sending you backwards further from the finish. The entrance space cannot be 1 or n, and it cannot coincide with the entrance or exit space of another shortcut.

You and your friends are getting bored of playing BAPC due to its simplicity. To make the game slightly more interesting, you have created four special dice. These dice are shaped like regular fair six-sided dice, but each die has its own selection of six numbers on the sides. Each player must choose one of the dice, and they must use that die for the entire duration of the game.

Before choosing the dice, the players agree on a player order, from player 1 to player 4. The players choose their die in reverse order, so player 4 is first to pick a die, and player 1 will get the final leftover die. After the dice have been distributed, the players take turns playing the game in their assigned order, so player 1 begins, and player 4 is last to play their first move.

To prevent the game from going on for too long, you also add a turn limit t. When each player has played t turns and nobody has reached the finish, the game ends immediately, and the player located on the highest numbered space wins. In case multiple players are tied for highest numbered space, the player order breaks the tie, so that player i beats player j if $i < j$.

You have just started a game of BAPC, and you are assigned player 4, so you are first to pick a die. Assuming every player picks the optimal die, maximizing their probability of winning, what is your probability of winning the game of BAPC?

As an example, consider the first sample case. The game lasts for only one turn, so the only way you can win as player 4 is by throwing the highest number. Your optimal choice of die is thus the fourth die, giving you a $\frac{1}{6}$ chance of throwing a 4 and winning the game.

For the second sample case, it can be shown that each player has a $\frac{1}{4}$ chance of winning, regardless of which die they pick.

Input

The input consists of:

- One line containing three integers n, m, and t ($2 \le n \le 1000$, $0 \le m \le n-2$, $1 \le t \le 1000$), the number of spaces on the board, the number of shortcuts, and the turn limit.
- Four lines, each with six integers d ($0 \le d \le 9$), the numbers on the sides of each die.
- m lines, each with two distinct integers a and b ($2 \le a \le n-1$, $1 \le b \le n$, $a \ne b$), the entrance space and the exit space of each shortcut. It is guaranteed that no shortcut shares its entrance space with the entrance space or the exit space of another shortcut.

Output

Output your probability of winning if every player picks the optimal die.

Your answer should have an *absolute* error of at most 10^{-6}.

Sample Input 1	Sample Output 1
100 0 1 3 3 3 3 3 3 2 3 3 3 3 3 2 2 2 2 2 2 1 1 1 1 1 4	0.1666666666666667

Sample Input 2	Sample Output 2
20 3 2 0 0 0 8 8 8 1 1 2 2 3 3 0 0 0 7 7 7 5 5 5 5 5 5 5 13 8 12 19 1	0.25

Sample Input 3	Sample Output 3
2 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0

E Exhausted Economist

Time limit: 2s

Your virtually infinite supply of BAPCs.
CC BY 2.0 by fdecomite on Flickr

After countless hours spent designing, optimizing and perfecting, you finally did it: you have found a way of producing Beautiful and Absolutely Perfect Cubes (BAPCs). Because you would like to make a lot of money from this invention, you decided to hire an economist. However, instead of just telling you how to become rich, they started giving you some useless lecture about the difference between profit and revenue, the sunk cost fallacy, and the importance of market research. Instead of letting them finish their speech, you decided to just tell them to do the market research themselves, while you focus on the important matters, such as producing as many BAPCs as possible.

Now it is one year later, you have a virtually infinite supply of BAPCs, and no money. You will need some income soon, to start paying your economist, for example. You would like to ask them for the optimal selling price of a BAPC, however, they have stopped answering your messages (they are probably exhausted from their market research). They did still send you the results of the market research, so you will need to use this to figure out the optimal selling price yourself.

The results consist of a list of all people interested in buying a BAPC, together with the maximal price at which they would buy. If you fix the selling price at p euros, then everyone on this list who indicated a maximal price of p or higher will buy a single BAPC for p euros. All others will not buy any BAPC.

The number of BAPCs you produced is way larger than the number of people on the list, so you do not need to worry about running out of BAPCs. What is your maximal revenue if you choose the optimal selling price?

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^5$), the number of people willing to buy a BAPC.
- One line with n integers p ($1 \leq p \leq 10^{12}$), the maximal price in euros that each person is willing to pay.

Output

Output your maximal revenue.

Sample Input 1	Sample Output 1
3 4 1 5	8

Sample Input 2

```
6
3 6 9 3 7 10
```

Sample Output 2

```
24
```

Input 3

```
3
1234567890 9876543210 2468013579
```

Output 3

```
9876543210
```

F Football Forecast

Time limit: 1s

CC BY 2.0 by Djuradj Vujcic on Wikimedia Commons

You would like to forecast the winner of the FIFA World Cup. This championship works in the following way:

First, there is group phase. In this phase, all participating teams are divided into groups of four in the order given in the input, and every pair of teams in such a group plays exactly one match. The winner scores 3 points, the loser 0 points. (For simplicity, we ignore the possibility of ties.) After the group phase, the two teams of each group with the most points proceed to the elimination phase.

In the elimination phase, matches are played according to a tournament bracket, which is a complete, rooted binary tree. The deepest layer of the tree has as many head-to-head matches as the number of groups. The teams ranked first in each group are assigned to these matches in the order given in the input, such that each match has one group winner. The teams ranked second in each group are then assigned to these matches in reverse order. For each match in this bracket, the loser is eliminated and the winner proceeds to the next layer in the bracket (ties are impossible here). The team winning the final match is declared World Champion.

Every team has a (unique) quality score, indicating how good it is. For simplicity, you assume that every match is won by the team with the higher quality score.

Given the quality score of every team and the complete structure of the tournament, determine who will win the World Cup.

Input

The input consists of:

- One line with an integer n ($4 \leq n \leq 1024$, n is a power of 2), the number of teams.
- n lines, each with a string t and an integer q ($1 \leq |t| \leq 20$, $1 \leq q \leq 10^6$), the name of the team and its quality score.

The team names only consist of English lowercase letters (`a-z`).

The team names are pairwise distinct. The quality scores of the teams are pairwise distinct.

Output

Output the name of the team that will win the World Cup.

Sample Input 1	**Sample Output 1**
4 netherlands 42 japan 31 sweden 24 tunisia 8	netherlands

Sample Input 2	**Sample Output 2**
8 australia 1337 kazakhstan 42 argentina 173205 belgium 314159 luxembourg 271828 canada 141421 antarctica 1 morocco 161803	belgium

G Grabbing Goods

Time limit: 2s

You are participating in the game show Bombastic Auction for Paired-up Contestants. You are facing a single opponent, and the goal for each of you is to win as much money as possible. There is a collection of n goods indexed $1, \ldots, n$ that must be split between you and your opponent. The two of you alternate grabbing one of the goods at a time, with you going first. Each good i has a value a_i known to both of you, and after the goods are distributed, you each add up the values of your grabbed goods to determine how much money you end up winning.

The optimal strategy for this game used to be incredibly simple: always grab the remaining good with the highest value. Critics were already referring to the show as the Boring Auction for Pinheaded Children. For this edition, the hosts have thus decided to change the rules so that the old strategy is no longer viable.

In the new edition, the goods are laid out in a formation so that they form the vertices of a rooted tree. Good 1 is at the root of the tree, and all the remaining goods i have a unique parent good $p_i < i$. Following the parent sequence of any good always leads to the root; there are no cycles. The new rule then states that you may only ever grab either the root, or a good whose parent has already been grabbed.

Under the new rule, this means that the first grabbed good must always be the good at the root of the tree. However, if any good had the root as its parent, then that good effectively becomes a new root of a new smaller tree. This means that the goods are always distributed among a number of disjoint rooted trees, and you must always grab the good at the root of one of the trees, splitting up the rest of that tree into a number of new disjoint rooted trees.

Both you and your opponent aim to maximize the amount of money you end up winning. With you grabbing the first good, and with both you and your opponent playing optimally, how much money do you end up winning?

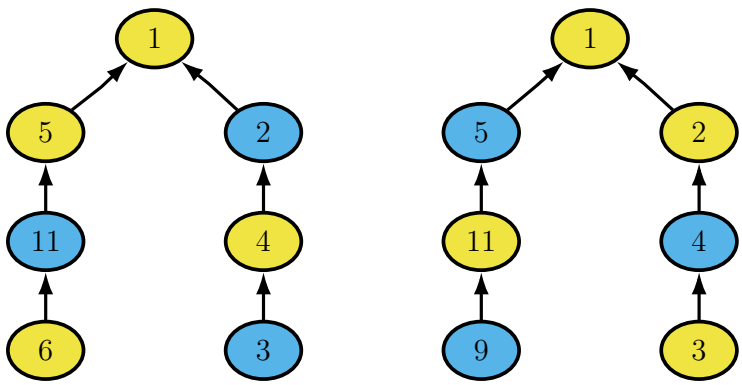


Figure G.1: Illustration for first two samples. With both you and your opponent playing optimally, you end up with the goods in the yellow vertices, and your opponent ends up with the goods in the blue vertices.

Input

The input consists of:

- One line with an integer n ($2 \leq n \leq 10^5$), the number of goods.
- One line with n integers $a_1, \ldots, a_n$ ($0 \leq a_i \leq 10^9$ for each i), the value of each good.
- One line with $n-1$ integers $p_2, \ldots, p_n$ ($1 \leq p_i < i$ for each i), where p_i is the index of the parent of good i. Recall that good 1 is at the root.

Output

Output the amount of money that you end up winning (the sum of the values of your grabbed goods) if both you and your opponent play optimally.

Sample Input 1	Sample Output 1
7 1 5 11 6 2 4 3 1 2 3 1 5 6	16

Sample Input 2	Sample Output 2
7 1 5 11 9 2 4 3 1 2 3 1 5 6	17

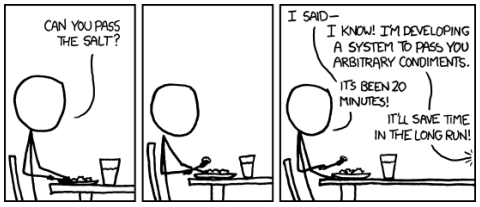
Sample Input 3	Sample Output 3
4 3 6 9 4 1 2 1	9

Sample Input 4	Sample Output 4
6 0 2 9 1 6 3 1 2 1 4 4	8

H Handing Over

Time limit: 5s

After winning the BAPC, you decided to host a celebratory dinner in your living room. For the main course, you prepared a huge circular table with room for n people to sit at. Every now and then, one person would like a condiment currently in use by another guest. To get the condiment, the person now in possession of the condiment will hand it over along a chain of people until it reaches the destination.



CC BY-NC 2.5 by Randall Munroe on xkcd.com

The n guests are sitting in n chairs at equidistant positions around the round table, and the guests are labeled in counterclockwise order from 1 to n. It is rude to interrupt people eating, so the only way to hand something over, is by reaching behind the chairs.

For each chair, you are given the wingspan of the person sitting there, which is measured in "how many chairs can one of their arms reach behind to pass something on".

Condiments are passed around very often. Since you and your guests are algorithm enjoyers, of course, these condiment handovers should be done as efficiently as possible.

Answer q queries of the type "What is the minimum number of handovers needed to hand over a condiment from person u to person v?" Some condiment is first in possession of person u, and through a series of handovers must be handed to person v.

Person x can hand over the condiment to person y, if the sum of their wingspans is large enough to bridge the number of chairs. More formally, if their wingspans are w_x and w_y, the handover can happen if $\min(|x-y|, n-|x-y|) \leq w_x + w_y$.

Input

The input consists of:

- One line with two integers n and q ($3 \leq n \leq 2 \cdot 10^5, 1 \leq q \leq 2 \cdot 10^5$), the number of chairs at the table and the number of handing over queries.
- One line with n integers $w_1, \ldots, w_n$ ($1 \leq w_i \leq n$ for each i), the wingspan of the person sitting at the ith chair.
- q lines, each containing two integers u and v ($1 \leq u, v \leq n, u \neq v$), describing a query of handing over a condiment from person u to person v.

Output

For each query, output the minimum number of handovers.

Sample Input 1

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

Sample Output 1

```
1
2
1
2
1
```

Sample Input 2

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

Sample Output 2

```
1
```

Sample Input 3

```
10 4
1 1 1 1 1 1 1 1 1 1
1 6
5 2
2 7
9 1
```

Sample Output 3

```
3
2
3
1
```

Sample Input 4

```
13 1
1 1 1 1 1 1 1 1 1 1 1 1 5
1 6
```

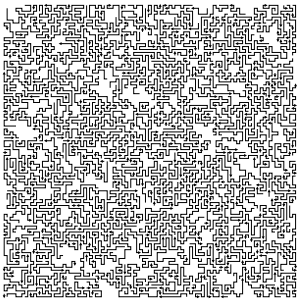
Sample Output 4

```
2
```

I Irritating Interruption I

Time limit: 5s

Like many people, Tonias is frequently plagued by mobile game ads. He is just watching some videos, ignoring the ads, when he notices something strange in one of them.



One ad showed Tonias this grid.

The game is quite simple: given some arrows in a rectangular grid, the goal is to remove all arrows. An arrow may be removed if it is not pointing at any arrow, including itself (i.e. no part of any arrow is straight in front of the arrow head at any distance). However, Tonias is fairly sure that it is not possible to remove all arrows!

This interruption has already taken too much time. He is now behind on his anime episodes, which means he is behind on going to bed, and thus on getting up and working on his thesis. It is, frankly, rather irritating. Write a program to help him determine how many arrows can actually be removed.

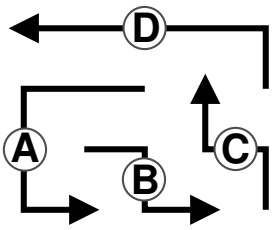


Figure I.1: Visualization of Sample Input 1. Arrows are given in the order $ABCD$ in the input. Since arrow A points at both B and C, B points at C, and C points at D, the arrows may only be removed in the order $DCBA$.

Input

The input consists of:

- One line with two integers w and h ($1 \leq w, h \leq 10^6$, $2 \leq w \cdot h \leq 10^6$), the width and height of the grid.
- One line with one integer k ($1 \leq k \leq \frac{w \cdot h}{2}$), the number of arrows.
- $2k$ lines, each pair of lines describing an arrow:
  - One line with two integers x and y ($0 \leq x < w$, $0 \leq y < h$), the coordinates of the tail of the arrow. The origin $(0, 0)$ is located in the bottom left.
  - One line with a string of characters c ($c \in \{\texttt{U}, \texttt{D}, \texttt{L}, \texttt{R}\}$), indicating the direction of the arrow from the tail onwards (up/down/left/right respectively, or the directions $+y/-y/-x/+x$). It is guaranteed the arrows do not leave the grid. They also do not overlap: any coordinate may only be part of at most one arrow.

Output

Output the number of arrows that can be removed.

Sample Input 1	Sample Output 1
5 4 4 2 2 LLDDR 1 1 RDR 4 0 ULU 4 2 ULLLL	4

Sample Input 2	Sample Output 2
3 3 2 0 2 DDR 2 0 UUL	0

J Joyful Jogging

Time limit: 2s

A group of people, joyfully jogging.
CC BY-SA 4.0 by wal on Pixabay

In the weekend of 26–27 September, the 100 Bruggenloop (100 Bridges Run) takes place in Leiden. In the past few years, the 20 km route across the 100 bridges has always been the same, but this year, the organization wants to offer more variety.

The city council has already approved a selection of roads to be closed for the race. To maximize the variety, the organization wants to create as many unique courses as the closed roads allow. Each course must form a cycle that starts and ends on the same road and does not reuse any road (it may reuse intersections). Of course, the different courses will have different numbers of bridges, which will contribute to the variety.

To make a course viable, it must include a start zone located on one of its roads. Since each course is a cycle, the start zone also serves as the finish zone. However, installing start zones is expensive, and having fewer of them creates a nicer atmosphere by letting more participants start together.

To support budget planning, determine the minimum number of start zones needed so that every course passes through *at least* one.

As an example, consider Sample Input 2. Having a single start zone on the edge between vertices 7 and 8 is not enough, because there is also a cycle along all the outer edges. Placing a second start zone in any of those edges makes sure that every course passes through a start zone.

Input

The input consists of:

- One line with two integers n and m ($2 \leq n \leq 2 \cdot 10^5$, $1 \leq m \leq 2 \cdot 10^5$), the number of intersections and the number of roads.
- m lines, each with two integers u and v ($1 \leq u, v \leq n$, $u \neq v$), describing a bidirectional road between intersections u and v.

It is guaranteed that every intersection is reachable from every other intersection. Note that there might be multiple roads connecting the same pair of intersections.

Output

Output the minimum number of start zones needed to cover all courses.

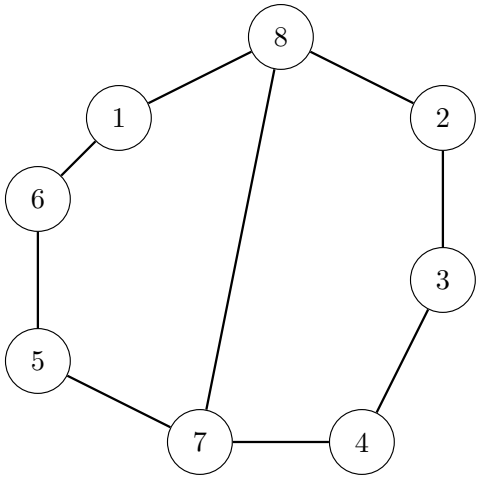


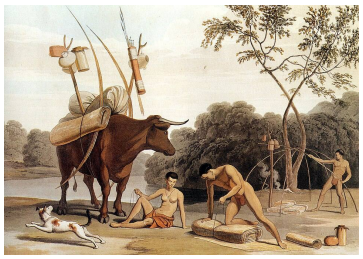
Figure J.1: Visualization of Sample Input 2, where two start zones are needed to cover all courses.

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

Sample Input 2	**Sample Output 2**
8 9 1 8 8 7 8 2 2 3 3 4 4 7 7 5 5 6 6 1	2

K Khoikhoi Dralinpome

Time limit: 1s



The Khoikhoi are a people from Southern Africa.
Public Domain by Samuel Daniell on Wikmedia Commons

Despite being barred from speaking at the Boring Annual Palindrome Convention for being too progressive, your girlfriend Jeannetta has continued her research on dralinpomes. A dralinpome is, of course, a word in which the letters can be rearranged to form a palindrome. For example, `khoikhoi` is a dralinpome, because you can rearrange the letters to form the string `khoiiohk`, which is a palindrome, since it reads the same forwards and backwards.

Last year, Jeannetta found all the dralinpomes in the dictionary, but she's not satisfied yet. Luckily, she discovered that in Dutch, you can concatenate nouns to make a compound word. Even if it means setting a stand somewhere in a dark corner outside of the convention center, people have to hear her breakthrough talk How the Dutch Dominate Dralinpomes!

Unfortunately, Jeannetta is not quite satisfied with her talk. The longest dralinpome she could come up with was `khoikhoihuttententoonstellingsbudgetbeheerder` (treasurer for the famously long exposition of African housings), with only 45 characters! She's fairly sure no one will show up to her talk without a longer example, so she has asked you to help.

Given a list of n nouns, determine the longest dralinpome that can be formed by concatenating some of them. You may use each noun no more times than it is provided.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 32$), the number of nouns.
- n lines, each with a string w ($1 \leq |w| \leq 20$), a noun. The nouns only consist of English lowercase letters (`a-z`).

Output

If it is possible to form a dralinpomic compound word, output "`possible`" followed by a longest possible such word. Otherwise, output "`impossible`".

If there are multiple valid solutions, you may output any one of them.

Sample Input 1	Sample Output 1
2 multinomiaal coefficient	possible multinomiaalcoefficient

Sample Input 2

```
5
auto
band
ventiel
dopjes
fabriek
```

Sample Output 2

```
impossible
```

Input 3

```
7
khoikhoi
soldaten
hutten
tentoonstellings
terreinen
budget
beheerder
```

Output 3

```
possible
khoikhoihuttententoonstellingsbudgetbeheerder
```

Sample Input 4

```
3
ei
ei
ei
```

Sample Output 4

```
possible
eiei
```

L Lock In

Time limit: 10s

Audrey is playing a game ranking her favourite 100 competitive programmers in a top 10 list. She already has objectively ranked all 100 of them. Now, one by one, she is given 10 distinct programmers uniformly at random, and must rank them among each other, in a relative top 10 list. Once she has made a choice, she is not allowed to move them any more, and she must stick to her choice. She cannot bear her assignments list being incorrect, so she tries to give a correct list as often as possible. Of course, she knows that it is impossible to always be right, so she is satisfied if she is correct at least 1% of the time. She is not allowed to assign the same spot to two different people. Help Audrey fill in her top 10 lists.

The podium for the best programmers. CC BY-SA 4.0 by Santeri Viinamäki on Wikimedia Commons

Interaction

This is an interactive problem. Your submission will be run against an *interactor*, which reads from the standard output of your submission and writes to the standard input of your submission. This interaction needs to follow a specific protocol:

Each game, exactly 10 times, the interactor will send one line with an integer n ($1 \leq n \leq 100$), the objective rank Audrey assigned to a competitive programmer. Then, your program should print one line with an integer r ($1 \leq r \leq 10$), the spot in the relative ranking Audrey should assign this programmer to.

Writing the same value of r twice in the same game will result in a wrong answer.

The 10 objective ranks given in each game are pairwise distinct.

After each game, the interactor will print one line with either "`win`" or "`loss`", depending on whether your list is correct or not. Note that, in each game, exactly 10 programmers need to be assigned a relative rank, even when it is clear that your list can no longer be correct.

Your submission will be run on exactly one test case, in which the interactor will play exactly 10 000 games. Winning fewer than 100 of these games will result in a wrong answer. Note that the sample interaction on the next page shows only two games for illustration purposes only.

The interactor is not adaptive: for each of your submissions, the 10 objective ranks of each independent game are uniformly randomly generated, and do not depend on your queries.

Make sure you flush the buffer after each write.

A testing tool is provided to help you develop your solution.

Read	Sample Interaction 1	Write
1		
	1	
2		
	2	
3		
	3	
4		
	4	
5		
	5	
6		
	6	
7		
	7	
8		
	8	
9		
	9	
10		
	10	
win 26		
	3	
3		
	1	
8		
	2	
50		
	5	
46		
	4	
52		
	6	
92		
	10	
95		
	9	
84		
	8	
42		
	7	
loss		

M Manic Ceramic

Time limit: 1s

The bathrooms at the Leiden University library; one of BAPC's biggest clients.

I do not know who is supplying these mathematicians with all this paint, but this has to stop. They have been defacing the tiling in practically every bathroom they visit with different colours and patterns, all apparently for "research purposes". All fun and games, until your friend decided your bathroom was the perfect place to get started on his own mathematical career. Now you're left having to look for new tiles, right in the middle of a global shortage due to the mathematical epidemic.

Luckily, there are still some tiles available at the Bespoke Arbitrary Products Company (BAPC), known for having everything you need, and everything you do not need! At the BAPC, they normally have three different types of patterned bathroom tiles: besides the standard tile with a single pattern on it, they also have multi- and partial-pattern tiles! Any multi-pattern tile contains not one, but x copies of the pattern. The partial-pattern tiles contain only part of the pattern, so that exactly y partial-pattern tiles are required to make up a single instance of the pattern. You would have preferred to keep it simple, and tile your entire bathroom with the standard tiles. However, the customer before you has just bought the last standard-pattern tiles! Thus, you are left with the two other types of tiles. Wanting to keep the same balance of patterns to tiles as you would have gotten with the standard tiles, you wonder what the minimum number of multi- and partial-pattern tiles is such that the total number of patterns across all tiles is exactly equal to the number of tiles.

You should buy at least one tile.

As an example, consider the first sample case, where the minimum case is achieved by four partial-pattern tiles and one multi-pattern tile, that together make up precisely $4 \cdot \frac{1}{4} + 4 = 5$ instances of the pattern.

Input

The input consists of:

- One line with two integers x and y ($2 \leq x, y \leq 10^8$), the number of patterns on a multi-pattern tile and the number of partial-pattern tiles required to make a single instance of the pattern.

Output

Output the minimum number of tiles you should buy so that you have exactly as many patterns as you have tiles.

Sample Input 1	Sample Output 1
4 4	5

Sample Input 2	**Sample Output 2**
2 9	17