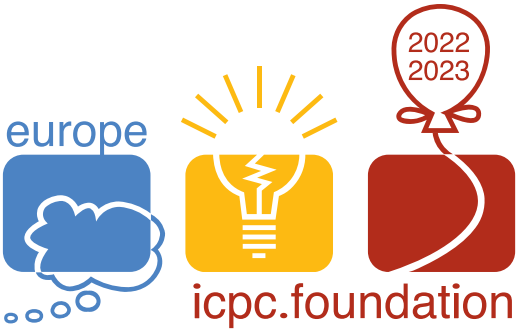
icpc international collegiate programming contest

ICPC Europe Contests

Northwestern Europe Regional Contest

Official Problem Set



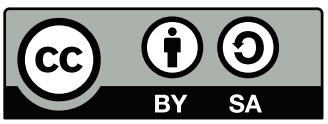
Northwestern Europe Regional Contest 2022

NWERC 2022

November 27, 2022

TU Delft Delft University of Technology

Problems



Problem A
Alternating Algorithm
Time limit: 8 seconds

CPUs. CC0 by Martijn Boer on Flickr

In recent years, CPU manufacturers have found it increasingly difficult to keep up with Moore's law of doubling the number of transistors on integrated circuit chips every two years. To address this, manufacturers have instead started creating CPUs with an increasingly higher number of cores. In fact, you just purchased a CPU with a staggering n number of cores, no less!

Incidentally, you also have an array of $n + 1$ integers, $a_0, a_1, \ldots, a_n$, that you need to sort. To make good use of the large number of cores on your CPU, you have devised a parallel sorting algorithm in which there is a dedicated core for comparing each adjacent pair of integers. As long as the array is not sorted in non-decreasing order, the algorithm proceeds in rounds that alternate between:

- Odd rounds (starting with the first): The first core compares a_0 and a_1, the third core compares a_2 and a_3, the fifth core compares a_4 and a_5, and so on. If a pair of compared elements are out of order, the corresponding core will swap their positions. If n is even, a_n will be left untouched.
- Even rounds: The second core compares a_1 and a_2, the fourth core compares a_3 and a_4, the sixth core compares a_5 and a_6, and so on. If a pair of compared elements are out of order, the corresponding core will swap their positions. If n is odd, a_n will be left untouched, and a_0 will be left untouched no matter what the parity of n is.

Note that in both types of rounds some cores may be idle.

Before implementing this algorithm, you have decided to do some analysis. In particular, you noticed that the time complexity of the algorithm does not depend on the value of n, but rather it depends on the number of rounds that the algorithm runs. Given the initial contents of the array, determine the number of rounds that the parallel sorting algorithm runs before the array becomes sorted.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 4 \cdot 10^5$), the number of cores and the size of the array.
- One line with $n + 1$ integers $a_0, a_1, \ldots, a_n$ ($0 \leq a_i \leq 10^9$ for each i), the initial contents of the array.

Output

Output the number of rounds that the parallel sorting algorithm runs before the array becomes sorted in non-decreasing order.

Sample Input 1	Sample Output 1
3 8 13 4 10	3

Sample Input 2	Sample Output 2
5 13 12 14 10 14 12	3

Sample Input 3	Sample Output 3
2 2 2 1	3

Problem B
Bottle Flip
Time limit: 1 second

Three bottles with different amounts of water.

It marks the year 2022 where the *bottle flip challenge* finally reached the last people on earth; the NWERC Jury. As you may know, the objective of the challenge is to flip a bottle of water $360°$ through the air and hope that it lands standing upright. Figure B.1 demonstrates a successful bottle flip.

After many failed attempts, we noticed that this task gets significantly easier if the bottle is filled with just the right amount of water. The simple reason for this is that the amount of water affects the centre of mass of our bottle as it is about to land.[1] A lower centre of mass makes it easier for the bottle to stay upright after it lands. Unfortunately, the optimal amount of water depends on the bottle, and we already wasted enough time on this challenge...

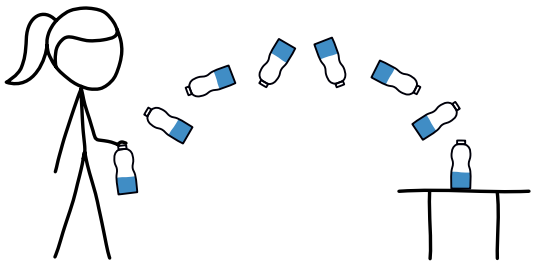


Figure B.1: Sketch of a bottle flip. The bottle is filled to roughly 33% to ease the challenge.

Given that our bottle is a perfect cylinder of height h and radius r, determine the optimal amount of water the bottle should contain so that our chances of landing a successful bottle flip are maximised. You can assume that both the water and the air contained in the bottle have uniform density, and that the weight of the bottle itself is negligible.

Input

The input consists of:

- One line with four integers h, r, d_a, and d_w ($1 \leq h, r, d_a, d_w \leq 1000$, $d_a < d_w$), where h and r are the height and radius of the bottle, and d_a and d_w are the densities of air and water, respectively.

Output

Output the height such that filling the bottle with water up to this height results in the lowest possible centre of mass while the bottle is standing upright. Your answer should have an absolute or relative error of at most 10^{-6}.

[1] For the sake of completeness, we define the centre of mass as the unique point at which the whole mass of the bottle could be concentrated without changing the bottle's reaction to gravity, regardless of the orientation of the bottle. Note that we implicitly assume that the water will stay at the bottom of the bottle. This defines exactly what you would intuitively think of as centre of mass.

Sample Input 1	Sample Output 1
22 4 1 4	7.3333333333

Sample Input 2	Sample Output 2
7 2 655 988	3.1415941720

Problem C
Circular Caramel Cookie
Time limit: 1 second

A traditional circular caramel cookie (stroopwafel).

Stroopwafels – two crispy round waffles with a grid-like pattern on top, separated by a thin layer of gooey and delicious caramel – are simply the most amazing Dutch treat ever. Everybody loves them and your factory is known for making the best and the biggest stroopwafels in town... At least, until now.

This year, your archrival Rob had the audacity to open up another factory for stroopwafels and they have already announced that their stroopwafels will be even bigger than yours. Although the exact size of the new stroopwafels is a well-kept secret, your industrial spy managed to find out that the grid-like pattern of the stroopwafel consists of at most s whole squares. You know for a fact that the area of each square is 1 cm^2 and that the centre point of the stroopwafel always contains the common corner of the four adjacent squares in the centre (i.e., the squares are aligned to a Cartesian grid), as shown in Figure C.1.

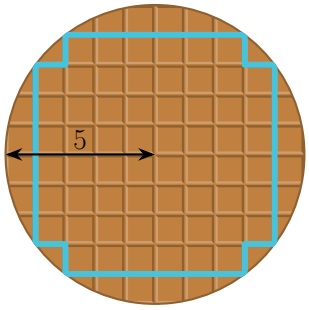


Figure C.1: Illustration of Sample Input 2, with the blue-enclosed region depicting the 60 whole squares that the cookie contains.

Needless to say, there is no way that you will let Rob outdo you and you plan on releasing a new edition of bigger stroopwafels. Since the production of bigger stroopwafels is more expensive, you naturally want to make them as small as possible. Thus, you are interested in the minimum radius of a stroopwafel with strictly more than s squares.

Input

The input consists of:

- One line with an integer s ($1 \leq s \leq 10^9$), the number of whole squares Rob's stroopwafel has at most.

Output

Output the minimum radius in centimetres of a stroopwafel with strictly more than s whole squares. Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
11	2.2360679775

Sample Input 2	Sample Output 2
59	5.0000000000

Problem D
Delft Distance
Time limit: 5 seconds

You are currently in your hotel at the north-west corner of Delft, and want to go to the contest site at the university in the south-east corner of Delft. To get there, you have to go right through the historical centre of the city. Like Manhattan, the city consists of a grid of $h \times w$ buildings. But unlike Manhattan, the city does not only contain square residential buildings but also some round medieval towers. All the square buildings are axis aligned with a side length of 10 m and all round towers have a diameter of 10 m. There is just enough space for a small alley of negligible width between two neighbouring buildings.

Delft water tower.
CC BY-SA 3.0 by Michiel1972 on Wikipedia

Since you are already late for the contest start, you need to find a shortest path from your hotel to the contest site. Fortunately, you have a map of the city. See Figure D.1 for an example.

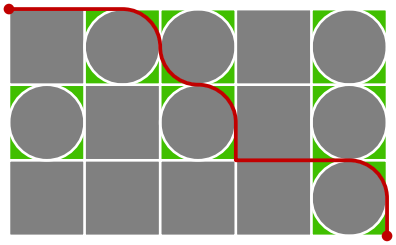


Figure D.1: Illustration of Sample Input 1, with a shortest path shown in red.

Input

The input consists of:

- One line with two integers h and w ($1 \leq h, w \leq 700$), the number of rows and the number of columns of buildings shown on the map of the city.
- h lines, each with w characters which are either '`O`' (for round towers) or '`X`' (for square buildings) describing the shapes of the buildings.

The map is oriented with the north side up.

Output

Output the length of a shortest path from the north-west corner to the south-east corner of Delft in metres. Your answer may have a relative or absolute error of at most 10^{-6}.

Sample Input 1	Sample Output 1
3 5 XOOXO OXOXO XXXXO	71.4159265359

Sample Input 2	Sample Output 2
1 4 XOOX	45.7079632679

Problem E
ETA
Time limit: 2 seconds

You want to design a level for a computer game. The level can be described as a connected undirected graph with vertices numbered from 1 to n. In the game, the player's character is dropped at one of the n vertices uniformly at random and their goal is to reach the exit located at vertex 1 as quickly as possible. Traversing an edge takes exactly 1 second.

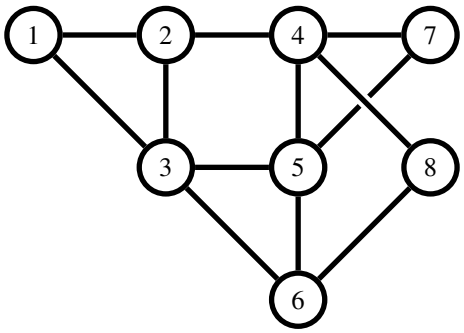


Figure E.1: Illustration of Sample Output 3, a level where the average optimal time to reach vertex 1 is $\frac{7}{4}$.

The difficulty of the level is determined by the average optimal time to reach the exit. Given a target value for this average optimal time, construct a level so that this target value is reached. See Figure E.1 for an example.

Input

The input consists of:

- One line with two coprime integers a and b ($1 \leq a, b \leq 1000$) separated by a '`/`', giving the desired average optimal time to reach the exit as the fraction $\frac{a}{b}$.

Output

If no connected graph with the average optimal time $\frac{a}{b}$ to reach vertex 1 exists, output "`impossible`". Otherwise, output one such graph in the following format:

- Two integers n and m ($1 \leq n, m \leq 10^6$), the number of vertices and the number of edges.
- m pairs of integers u and v ($1 \leq u, v \leq n$), indicating an edge between vertices u and v.

The graph may include self loops and parallel edges. You are given that if there exists a valid graph, then there also exists one with $1 \leq n, m \leq 10^6$.

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

Sample Input 1	Sample Output 1
`1/2`	`2 1` `1 2`

Sample Input 2	Sample Output 2
1/3	impossible

Sample Input 3	Sample Output 3
7/4	8 12 1 2 1 3 2 3 2 4 3 5 3 6 4 5 5 6 4 7 5 7 4 8 6 8

Problem F
Faster Than Light
Time limit: 8 seconds

Faster Than Light (FTL) is a space-based top-down strategy game by Subset Games. You control a ship which belongs to the *Galactic Federation* and have to fight a ship of the *Rebel Faction*. The enemies' spaceship is represented by a set of axis-aligned non-intersecting rectangles in the plane which correspond to the rooms of the ship. Your ship is close to destruction, but your weapon, the *hull beam*, is ready to fire.

The hull beam shoots an infinite beam in a direction of your choice that deals one damage to each room it intersects. Coincidentally, the enemies' ship consists of n rooms and has exactly n health points. Thus, you need to hit every room to destroy the enemy before they destroy you. Now you quickly need to check if it is possible to position the beam in such a way. Note that the beam deals damage even when it only touches the boundary of a room. See Figure F.1 for an example.

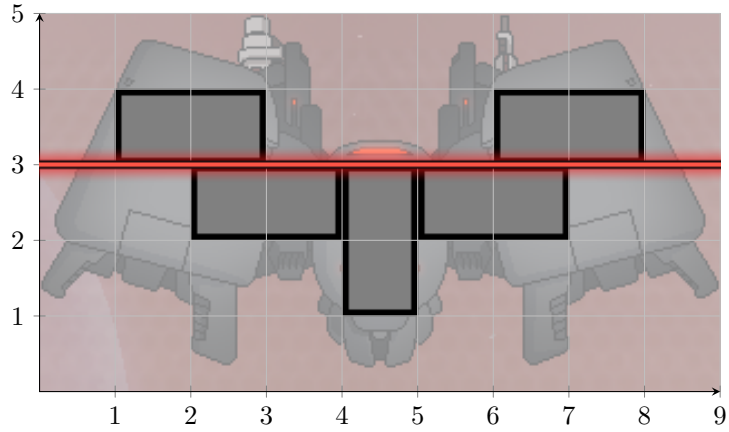


Figure F.1: Illustration of Sample Input 1, which consists of five grey rooms. The hull beam in red hits all rooms and is the only valid solution in this case.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 2 \cdot 10^5$), the number of rooms.
- n lines, each with four integers x_1, y_1, x_2, and y_2 ($0 \leq x_1 < x_2 \leq 10^9$ and $0 \leq y_1 < y_2 \leq 10^9$), describing the coordinates of two opposite corners (x_1,y_1) and (x_2,y_2) of a room.

It is guaranteed that no two rooms have a common interior point.

Output

If it is possible for the hull beam to pass through all rooms at once, output "`possible`". Otherwise, output "`impossible`".

Sample Input 1

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

Sample Output 1

```
possible
```

Sample Input 2

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

Sample Output 2

```
impossible
```

Sample Input 3

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

Sample Output 3

```
possible
```

Problem G
Going in Circles
Time limit: 2 seconds

Train. Unsplash License by Robin Ulrich on Unsplash

Hercule Poirot, world-renowned detective, was having a lovely cup of tea in his compartment on the Disorient Express when the train conductor rushed in. "We have lost track of the number of train carriages," exclaimed the conductor. You see, this was no ordinary train, and had no first or last train carriage. Instead, the train carriages were connected to form a large cycle, causing the conductor's confusion.

Hercule thought for a moment. "That is most peculiar," he said. "But I may be able to help you." He reached over and grabbed the lamp from the side table. "You see, each train carriage has a light switch like this one. By moving between the carriages and toggling these switches, we can determine the number of train carriages."

The conductor was sceptical, but agreed to try it. "We are in a hurry," he said, "so please determine n, the number of carriages that the train consists of, in at most $3n + 500$ steps." Here a step counts as either moving to an adjacent carriage or toggling a light switch in the current carriage. "The only thing I am certain of is that n is at least 3 and at most 5000."

Interaction

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

The interactor first sends an integer s ($s \in \{0, 1\}$), the state of the light switch in Hercule's initial carriage.

Your submission may then send at most $3n+500$ steps[2] for Hercule to carry out, each by printing one line of the form "`?` a", where a is one of the following:

- "`left`", to move to the adjacent train carriage in the clockwise direction,
- "`right`", to move to the adjacent train carriage in the counter-clockwise direction, or
- "`flip`", to toggle the light switch in the current train carriage.

After every step, the interactor replies with an integer s ($s \in \{0, 1\}$), the state of the light switch in the train carriage that Hercule is currently in, after he has carried out the requested step.

When you have determined the number of train carriages n, print one line of the form "`!` n" ($3 \leq n \leq 5000$), after which the interaction will stop. Printing the answer does not count as a query.

The interactor is not adaptive; the initial state of the light switches is determined by the interactor before any interaction takes place. Make sure you flush the buffer after each write. Using more than $3n + 500$ queries will result in a wrong answer verdict.

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

[2]Note that n is the actual number of train carriages in the current test case, not the maximum possible number of train carriages.

Read	Sample Interaction 1	Write
0		
	? right	
1		
	? right	
1		
	? right	
0		
	? flip	
1		
	? left	
1		
	? left	
1		
	? left	
1		
	! 3	

Problem H
High-quality Tree
Time limit: 3 seconds

The binary search tree is one of the most useful data structures in computer science. Many methods exist to keep them balanced, such as using tree rotations (like in AVL-trees) or randomness (like in treaps). One thing that these methods have in common is that they are all somewhat slow and complicated.

But this is all about to change! Rob the computer scientist has invented a new method to keep rooted binary trees balanced, that is much better than the current state-of-the-art. The main idea is to repeatedly perform an operation Rob calls a *robbery*. One robbery is to take a leaf from the tree, and remove it. By applying robberies at the right times, Rob is able to keep a tree balanced in $\mathcal{O}(1)$ amortized time.

Some people have criticised Rob's revolutionary discovery, saying that removing elements from the tree makes the algorithm incorrect. Rob does not agree that this is a big problem; if you plan to store $2 \cdot 10^5$ numbers, you probably do not need all of them. But Rob accepts the criticism and decides to find a way to minimise the number of robberies.

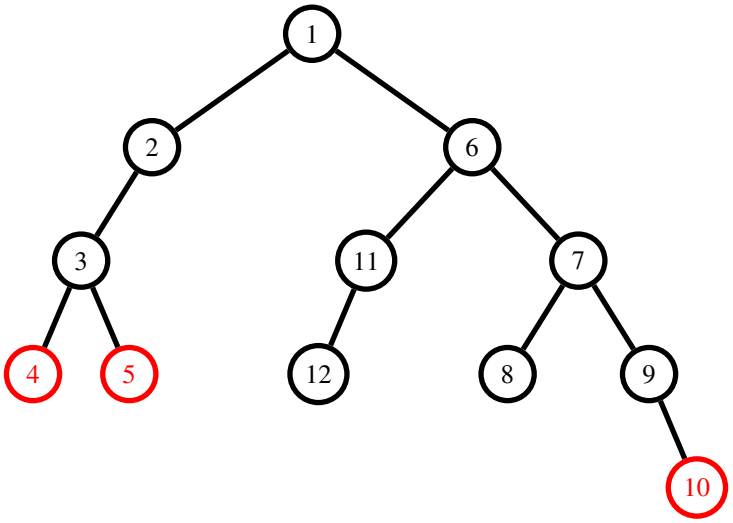


Figure H.1: Illustration of Sample Input 2. The tree becomes strongly balanced after removing the three vertices marked in red (4, 5, and 10); the minimum number of vertices that need to be removed to make the tree strongly balanced.

You are given a rooted binary tree with n vertices. The vertices are numbered from 1 to n, and vertex 1 is the root. Your task is to find the minimum number of robberies to make the tree *strongly balanced*, meaning that all of its subtrees are balanced. A rooted binary tree is called balanced if the depth of its left and right subtrees differ by at most one. Recall that a robbery is simply to take a leaf and remove it, and doing so may turn its parent vertex into a leaf. See Figure H.1 for an example.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 2 \cdot 10^5$), the number of vertices in the tree.
- $n - 1$ lines, each with two integers u and v ($1 \leq u, v \leq n$, $u \neq v$), indicating an edge between vertices u and v.

It is guaranteed that the given edges form a valid binary tree with vertex 1 as the root. However, the edges can appear in any order and are not directed: u can be the parent of v or the other way around.

Output

Output the minimum number of leaves you need to remove to make the tree strongly balanced.

Sample Input 1

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

Sample Output 1

```
1
```

Sample Input 2

```
12
1 2
2 3
3 4
3 5
1 6
6 7
7 8
7 9
9 10
6 11
11 12
```

Sample Output 2

```
3
```

Problem I
Interview Question
Time limit: 1 second

Fizz Buzz is a party game that is often used as a programming exercise in job interviews. In the game, there are two positive integers a and b, and the game consists of counting up through the positive integers, replacing any number by `Fizz` if it is a multiple of a, by `Buzz` if it is a multiple of b, and by `FizzBuzz` if it is a multiple of both a and b. The most common form of the game has $a = 3$ and $b = 5$, but other parameters are allowed.

Your task here is to solve the reverse problem: given a transcript of part of the game (not necessarily starting at 1), find possible values of a and b that could have been used to generate it.

Fizz Buzz implemented in Hexagony.
CC BY-SA 3.0 by M L on codegolf.stackexchange.com

Figure I.1 shows some sample sequences for various values of a and b.

$a = 3, b = 5$: `1 2 Fizz 4 Buzz Fizz 7 8 Fizz Buzz 11 Fizz 13 14 FizzBuzz`
$a = 6, b = 2$: `1 Buzz 3 Buzz 5 FizzBuzz 7 Buzz 9 Buzz 11 FizzBuzz 13`
$a = 4, b = 4$: `1 2 3 FizzBuzz 5 6 7 FizzBuzz 9 10 11 FizzBuzz 13 14`

Figure I.1: Example sequences for *Fizz Buzz*.

Input

The input consists of:

- One line with two integers c and d ($1 \leq c \leq d \leq 10^5$), indicating that your transcript starts at c and ends at d.
- One line with $d - c + 1$ integers and strings, the contents of the transcript.

It is guaranteed that the transcript is valid for some integers a and b with $1 \leq a, b \leq 10^6$, according to the rules laid out above.

Output

Output two positive integers a and b ($1 \leq a, b \leq 10^6$) that are consistent with the given transcript.

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

Sample Input 1	Sample Output 1
`7 11` `7 8 Fizz Buzz 11`	`3 5`

Sample Input 2	Sample Output 2
`49999 50002` `49999 FizzBuzz 50001 Fizz`	`2 125`

Sample Input 3	Sample Output 3
8 11 Buzz Buzz FizzBuzz Buzz	10 1

Sample Input 4	Sample Output 4
10 15 10 11 12 13 14 15	8 23

Problem J
Justice Served
Time limit: 6 seconds

Rob. Pixabay License by Henning on Pixabay

You finally managed to produce the ultimate stroopwafel with just the right number of squares on it. After the hard work, you left the stroopwafel unattended for a few seconds to get yourself a hot drink. Full of anticipation for your delicious treat, you came back just to see that your stroopwafel was gone! Even though you were only away for a short time, someone used the opportunity to steal it.

You looked at the security recordings and saw a total of n suspects that had been in the room where you left your stroopwafel, each entering and leaving the room exactly once. After seeing this, you already had a good idea who took it since your archrival Rob – who has some background in robberies – was among the suspects. But you wanted to give him the benefit of the doubt and decided to interrogate every suspect. Unsurprisingly, every suspect claimed their own innocence. However, some suspects also provided an alibi for other suspects. Specifically, a suspect A provided an alibi for suspect B if and only if A was in the room for the entire duration B was in the room.

You feel like a suspect is more convincing if they have an alibi provided by a suspect who is convincing themselves. Formally, a suspect without an alibi has *convincingness* 0. Otherwise, their convincingness is 1 more than the convincingness of the most convincing suspect who provides them with an alibi. Your task is to compute the convincingness of each suspect.

Input

The input consists of:

- One line with a single integer n ($1 \leq n \leq 2 \cdot 10^5$), the number of suspects.
- n lines, each with two integers a and t ($1 \leq a, t \leq 10^9$), the time at which each suspect arrived and the duration they stayed.

It is guaranteed that no two suspects who arrived at the same moment stayed for the same duration.

Output

Output the convincingness of each suspect.

Sample Input 1	Sample Output 1
4 2 8 1 7 4 5 5 2	0 0 1 2

Sample Input 2	Sample Output 2
5 2 4 3 3 2 2 4 2 4 1	0 1 1 2 3

Problem K
Kebab Pizza
Time limit: 4 seconds

Alice loves pizza and plans to throw a pizza party for her birthday, where she will make one big, round pizza for all her guests. She knows that her friends are a bit special when it comes to pizza: Julie does not want any vegetables but loves kebab meat on her pizza. Katy would never eat pizza with cheese on top. Alice herself is a huge fan of pineapple on pizza, while Mickey absolutely hates it. Instead, he wants to put spaghetti on it, which everybody else finds weird. The list goes on and on. Briefly put, it is absolutely impossible to accommodate everybody.

As a compromise, Alice decides to let each person choose two toppings for their pizza slice in advance. Each person will be happy if their slice has exactly their two chosen toppings and none of the others. Note that a person may choose the same topping twice, in which case they just get twice the amount of that topping.

Figure K.1: Illustration of Sample Input 1, with two toppings numbered on every slice in one possible solution. Note that each topping only occurs on a single range of pizza slices.

On the day of her party, Alice discovers that the rolled out pizza dough takes up so much space on her kitchen counter that she can only prepare one kind of topping at a time. To optimise the workflow and to make sure that the pizza is ready in time for her party, she wants to take out each chosen kind of topping only once and then add it on a consecutive range of pizza slices. Note that the whole pizza forms a consecutive range of pizza slices in a circular fashion. Determine whether it is possible to prepare the pizza in this way while satisfying all the chosen topping combinations. See Figure K.1 for an example.

Input

The input consists of:

- One line with two integers n and k ($3 \leq n, k \leq 10^5$), the number of pizza slices and the number of possible toppings which are numbered from 1 to k.
- n lines, each with two integers a and b ($1 \leq a, b \leq k$), describing the toppings chosen by the ith person.

Output

Output "possible" if it is possible for Alice to select a consecutive range of pizza slices for each chosen topping such that the resulting pizza can be split into n suitable slices. Otherwise, output "impossible".

Sample Input 1

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

Sample Output 1

```
possible
```

Sample Input 2

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

Sample Output 2

```
possible
```

Sample Input 3

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

Sample Output 3

```
impossible
```

Problem L
Last Guess
Time limit: 2 seconds

If you spent any amount of time on social media at the end of last year, you are probably familiar with the word finding game *Wordle*, in which you need to find a five-letter English word using at most six guesses. After each guess, the letters in that guess are marked in either green, yellow or black to provide information about the hidden word:

- Green indicates that the letter is in the hidden word and appears in the same position.
- Yellow indicates that the letter is in the hidden word, but in a different position.
- Black indicates that the letter has no more occurrences in the hidden word.
- If a letter appears more than once in the guessed word, first all the green markings for that letter are placed. Then, from left to right in the guessed word, yellow markings are placed for each remaining occurrence of the letter in the hidden word. Finally, black markings are placed for any surplus of the letter in the guessed word. See Figure L.1 for an example.

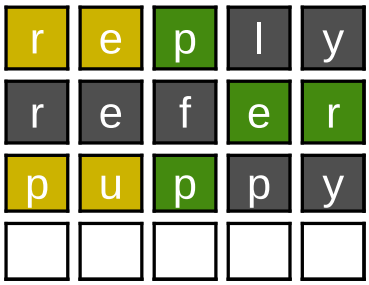


Figure L.1: Illustration of Sample Input 1, where "`upper`" is the only valid last guess.

In this problem, we consider a variant of Wordle where you need to find a hidden word of length ℓ in at most g guesses. Additionally, there is *no* requirement that the hidden word or any of the guesses are words of any language; any string consisting of lowercase English letters is fine.

In your current play of this variant, you have already used all but one of your guesses, and now you need to find the hidden word using your final guess. Find any word that could be the hidden word based on the information you have.

Input

The input consists of:

- One line with two integers g and ℓ ($2 \leq g \leq 500$, $1 \leq \ell \leq 500$), the maximal number of guesses allowed in the game and the length of the word.
- $g - 1$ lines, each with two strings s and t of length ℓ, where s is one of the guesses so far and t gives the colours for that guess according to the rules above. The string s consists of lowercase English letters (`a-z`) and the string t consists of uppercase letters '`G`', '`Y`', and '`B`' indicating green, yellow, and black respectively.

It is guaranteed that the input describes a valid state of a Wordle game before the last guess and that a valid solution exists, in the form of a length ℓ word consisting of lowercase English letters.

Output

Output a valid last guess that gives you a chance of winning.

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

Sample Input 1	Sample Output 1
4 5 reply YYGBB refer BBBGG puppy YYGBB	upper

Sample Input 2	Sample Output 2
2 12 aabbccddeeff GGGYGBYYYBBB	aabdcbegdhij