Problem A
A-Skew-ed Reasoning
Time limit: 2 seconds

The following is based on a true story – the names have been changed because... well, because you always change names in stories like this one.

Professor Taylor Swift is grading a homework assignment on integer skew heaps. A skew heap is a binary tree with an integer stored in each node such that the value in any node is less than or equal to the values in any of its children. Note that the skew heap need not be a perfect binary tree; that is, the left and/or right subtree of any node may be empty.

Inserting a value x into a skew heap H is done using the following recursive procedure:

- If H is empty, make H a skew heap consisting of a single node containing x.
- Otherwise, let y be the value in the root of H.
  - If $y < x$, swap the two children of the root and recursively insert x into the new left subtree.
  - If $y \geq x$, create a new node with value x and make H the left subtree of this node.

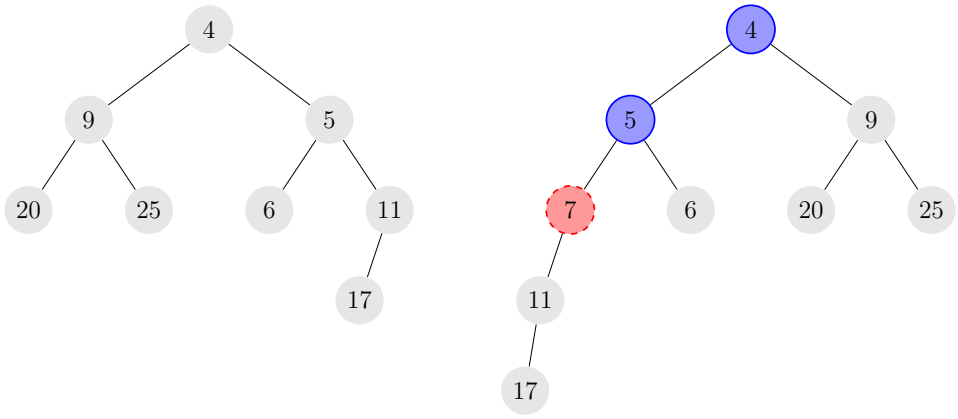

Figure A.1: Example of inserting the value 7 into a skew heap. The nodes storing 4 and 5 (marked in blue) have their children swapped, while the node storing 11 becomes the left child of the newly inserted node (marked in red).

Now, back to Professor Swift. The homework problem she has assigned asks the students to show the heap that results from inserting a given permutation of the numbers from 1 to n, in the given order, into an empty heap. Surprisingly, some of the students have wrong answers! That got Professor Swift wondering: For a given heap, is there an input permutation that would have produced this heap? And if so, what are the lexicographically minimal and maximal such input permutations?

Input

The first line of input contains an integer n ($1 \leq n \leq 2 \cdot 10^5$), the number of nodes in the tree. These nodes contain the numbers from 1 to n exactly. This is followed by n lines, the i^{th} of which contains two integers ℓ_i and r_i ($i < \ell_i \leq n$ or $\ell_i = 0$; $i < r_i \leq n$ or $r_i = 0$), describing the values of the left and right children of the node storing i, where a value of 0 is used to indicate that the corresponding child does not exist. It is guaranteed that this data describes a binary tree.

Output

Output the lexicographically minimal input permutation that produces the given tree under the insertion method for skew heaps, followed by the lexicographically maximal such input permutation. These permutations may coincide, in which case you still need to output both. If no input permutation producing the given tree exists, output `impossible`.

Sample Input 1

```
7
2 3
4 5
6 7
0 0
0 0
0 0
0 0
```

Sample Output 1

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

Sample Input 2

```
2
0 2
0 0
```

Sample Output 2

```
impossible
```

Sample Input 3

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

Sample Output 3

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

Problem B
Blackboard Game
Time limit: 1 second

To help her elementary school students understand the concept of prime factorization, Aisha has invented a game for them to play on the blackboard. The rules of the game are as follows.

The game is played by two players who alternate their moves. Initially, the integers from 1 to n are written on the blackboard. To start, the first player may choose any even number and circle it. On every subsequent move, the current player must choose a number that is either the circled number multiplied by some prime, or the circled number divided by some prime. That player then erases the circled number and circles the newly chosen number. When a player is unable to make a move, that player loses the game.

To help Aisha's students, write a program that, given the integer n, decides whether it is better to move first or second, and if it is better to move first, figures out a winning first move.

Input

The first line of input contains an integer t ($1 \leq t \leq 40$), which is the number of test cases. The descriptions of t test cases follow.

Each test case consists of a single line containing an integer n ($2 \leq n \leq 10^7$), which is the largest number written on the blackboard.

Over all test cases, the sum of n is at most 10^7.

Output

For each test case, if the first player has a winning strategy for the given n, output the word `first`, followed by an even integer – any valid first move that can be extended to a winning strategy. If the second player has a winning strategy, output just the word `second`.

Sample Input 1	Sample Output 1
1 5	second

Explanation of Sample 1: For $n = 5$, the first player loses the game regardless of the first move.

- If the first player starts with 2, the second player circles 4, and there are no more valid moves left.
- If the first move is 4, the second player circles 2. The first player must then circle 1, and the second player may pick either of the remaining two numbers (3 or 5) to win.

Sample Input 2	Sample Output 2
2 12 17	first 8 first 6

This page is intentionally left blank.

Problem C
Bride of Pipe Stream
Time limit: 12 seconds

The story continues! For several years now, your town has been gifted with an abundance of Flubber, the adorable-but-slightly-flammable-and-toxic-and-acidic-and-sentient-and-mischievous man-made chemical. The search continues for more (or, well, any) uses for the substance. But in the meantime, the Flubber factory continues to produce it at full capacity. Efforts to shut it down have failed, partly because nobody is sure who is actually running the factory.

You've been tasked with storing the perpetually-flowing Flubber in various Flubber reservoirs for future use (or, at least, to get it out of everyone's hair – literally). To accomplish this, you have access to a complicated network of Flubber ducts, connecting up various Flubber stations and reservoirs.

Every Flubber station has one or more Flubber ducts leading from it, and has various gates that may be raised or lowered so that incoming Flubber will drain into the output Flubber ducts in any desired proportion. For instance, you can send all the Flubber down one duct, or split it between two ducts 25–75, etc.

In contrast, a Flubber duct flows down to one or more lower stations or reservoirs, but the Flubber drains into them in a fixed proportion that you do not control. It is possible that some of the Flubber is lost to the environment as well, but that is a problem for your successor, not you.

You would like to fill all the reservoirs as quickly as possible. That is, you want to maximize the minimum amount of Flubber flowing into any of the reservoirs, among all possible configurations of station drainage.

Figure C.1 illustrates the two sample inputs. Stations and reservoirs are shown as numbered nodes, colored green for stations and blue for reservoirs. Ducts are depicted as white nodes. For example, in the first sample input (left), Flubber can be sent from station 1 in any proportion to its two downstream ducts, but each duct will distribute its inflow according to the percentages printed on its outgoing edges.

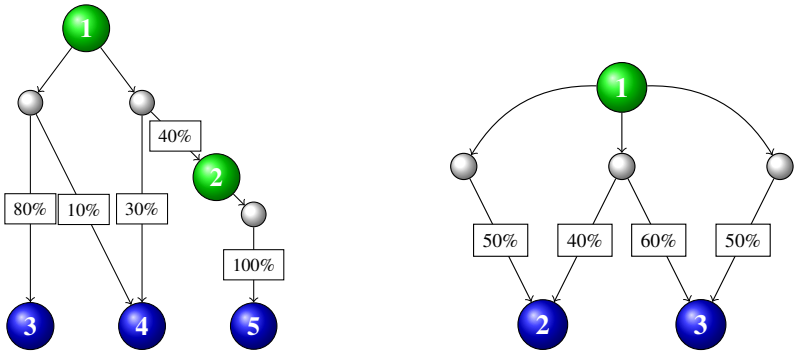

Figure C.1: Illustrations of the two sample inputs.

Input

The first line of input contains three integers s, r, and d, where s ($1 \leq s \leq 10\,000$) is the number of stations, r ($1 \leq r \leq 3$) is the number of reservoirs, and d ($s \leq d \leq 20\,000$) is the number of ducts. The stations are numbered from 1 to s and the reservoirs are numbered from $s + 1$ to $s + r$, in decreasing order of altitude. The factory's Flubber initially flows into station 1.

Each of the remaining d lines starts with two integers i and n, where i ($1 \leq i \leq s$) is the station that can drain into this duct, and n ($1 \leq n \leq 10$) is the number of outputs of this duct. The remainder of the line contains n pairs of integers o and p, where o ($i < o \leq s + r$) is a station or reservoir to which this duct drains, and p ($1 \leq p \leq 100$) is the percentage of the Flubber entering the duct that will drain to o. The o values for a given duct are distinct. Every station has at least one duct that it can drain into. The percentages for a given duct's outputs will sum to at most 100.

Output

Output a single percentage f, which is the highest possible percentage such that, for some configuration of station drainage, all reservoirs receive at least $f\%$ of the factory's produced Flubber. Your answer should have an absolute error of at most 10^{-6}.

Sample Input 1	Sample Output 1
2 3 3 1 2 3 80 4 10 1 2 2 40 4 30 2 1 5 100	24.0

Sample Input 2	Sample Output 2
1 2 3 1 1 2 50 1 1 3 50 1 2 2 40 3 60	42.8571428571

Problem D
Buggy Rover
Time limit: 2 seconds

The International Center for Planetary Cartography (ICPC) uses rovers to explore the surfaces of other planets. As we all know, other planets are flat surfaces which can be perfectly and evenly discretized into a rectangular grid structure. Each cell in this grid is either flat and can be explored by the rover, or rocky and cannot.

Mars rover being tested near the Paranal Observatory.
CC BY-SA 4.0 by ESO/G.
Hudepohl on Wikimedia Commons

Today marks the launch of their brand-new *Hornet* rover. The rover is set to explore the planet using a simple algorithm. Internally, the rover maintains a *direction ordering*, a permutation of the directions north, east, south, and west. When the rover makes a move, it goes through its direction ordering, chooses the first direction that does not move it off the face of the planet or onto an impassable rock, and makes one step in that direction.

Between two consecutive moves, the rover may be hit by a cosmic ray, replacing its direction ordering with a different one. ICPC scientists have a log of the rover's moves, but it is difficult to determine by hand if and when the rover's direction ordering changed. Given the moves that the rover has made, what is the smallest number of times that it could have been hit by cosmic rays?

Input

The first line of input contains two integers r and c, where r ($1 \leq r \leq 200$) is the number of rows on the planet, and c ($1 \leq c \leq 200$) is the number of columns. The rows run north to south, while the columns run west to east.

The next r lines each contain c characters, representing the layout of the planet. Each character is either '`#`', a rocky space; '`.`', a flat space; or '`S`', a flat space that marks the starting position of the rover. There is exactly one '`S`' in the grid.

The following line contains a string s, where each character of s is '`N`', '`E`', '`S`', or '`W`', representing the sequence of the moves performed by the rover. The string s contains between 1 and 10 000 characters, inclusive. All of the moves lead to flat spaces.

Output

Output the minimum number of times the rover's direction ordering could have changed to be consistent with the moves it made.

Sample Input 1	Sample Output 1
5 3 #.. ... ... .S. NNEN	1

Explanation of Sample 1: The rover's direction ordering could be as follows. In the first move, it either prefers to go north, or it prefers to go south and then north. Note that in the latter case, it cannot move south as it would fall from the face of the planet. In the second move, it must prefer to go north. In the third move, it must prefer to go east. In the fourth move, it can either prefer to go north, or east and then north. It is therefore possible that it was hit by exactly one cosmic ray between the second and third move, changing its direction ordering from `N???` to `EN??` where '`?`' stands for any remaining direction.

Sample Input 2	Sample Output 2
3 5 .###. ....# .S... NEESNS	0

Explanation of Sample 2: It is possible the rover began with the direction ordering `NESW`, which is consistent with all moves it makes.

Sample Input 3	Sample Output 3
3 3 ... ... S#. NEESNNWWSENESS	4

Problem E
Delivery Service
Time limit: 12 seconds

The Intercity Caspian Package Company (ICPC) is starting a delivery service which will deliver packages between various cities near the Caspian Sea. The company plans to hire couriers to carry packages between these cities.

Each courier has a home city and a destination city, and all couriers have exactly the same travel schedule: They leave their home city at 9:00, arrive at their destination city at 12:00, leave their destination city at 14:00 and return to their home city at 17:00. While couriers are in their home or destination cities, they can receive packages from and/or deliver packages to customers. They can also hand off to or receive packages from other couriers who are in that city at the same time. Since ICPC is a personal service, packages are never left in warehouses or other facilities to be picked up later – unless the package has reached its destination, couriers have to either keep the package with themselves (during the day or during the night), or hand it off to another courier.

The company will direct the couriers to hand off packages in such a way that any package can always be delivered to its destination. Or so it is hoped! We'll say that two cities u and v are *connected* if it is possible to deliver a package from city u to city v as well as from v to u. To estimate the efficiency of their hiring process, the company would like to find, after each courier is hired, the number of pairs of cities (u, v) that are connected ($1 \leq u < v \leq n$).

Input

The first line of input contains two integers n and m, where n ($2 \leq n \leq 2 \cdot 10^5$) is the number of cities, and m ($1 \leq m \leq 4 \cdot 10^5$) is the number of couriers that will be hired. Couriers are numbered 1 to m, in the order they are hired. This is followed by m lines, the i^{th} of which contains two distinct integers a_i and b_i ($1 \leq a_i, b_i \leq n$), denoting the home and destination cities, respectively, for courier i.

Output

Output m integers, denoting the number of pairs of connected cities after hiring the first $1, 2, \ldots, m$ couriers.

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

Explanation of Sample 1:

1. After the first courier is hired, cities 1 and 2 are connected.
2. After the second courier is hired, cities 2 and 3 are connected. Note, however, that cities 1 and 3 are still not connected. Even though there's a courier moving between cities 1 and 2, and a courier moving between cities 2 and 3, they never meet each other.
3. After the third courier is hired, cities 3 and 4 are connected and cities 2 and 4 are connected. For example, one way to deliver a package from city 2 to city 4 is:
   - hand it to courier 2 in city 2 at 19:00;
   - the next day, courier 2 arrives in city 3 at 12:00, and hands the package to courier 3 who is also in city 3;
   - at 18:00, courier 3 delivers the package to city 4.
4. After the fourth courier is hired, all six pairs of cities are connected.

Problem F
Herding Cats
Time limit: 2 seconds

You are opening a cat cafe in Baku and would like to take a promotional photograph of all the cats sitting in the front window. Unfortunately, getting cats to do what you want is a famously hard problem. But you have a plan: you have bought a collection of m catnip plants, each of a different variety, knowing that each cat likes some of these varieties. There is a row of m pots in the window, numbered 1 to m in order, and you will place one plant in each pot. Each cat will then be persuaded (by means of a toy on a string) to walk along the row of pots from 1 to m. As soon as a cat reaches a pot with a catnip plant that it likes, it will stop there, even if there already are other cats at that plant.

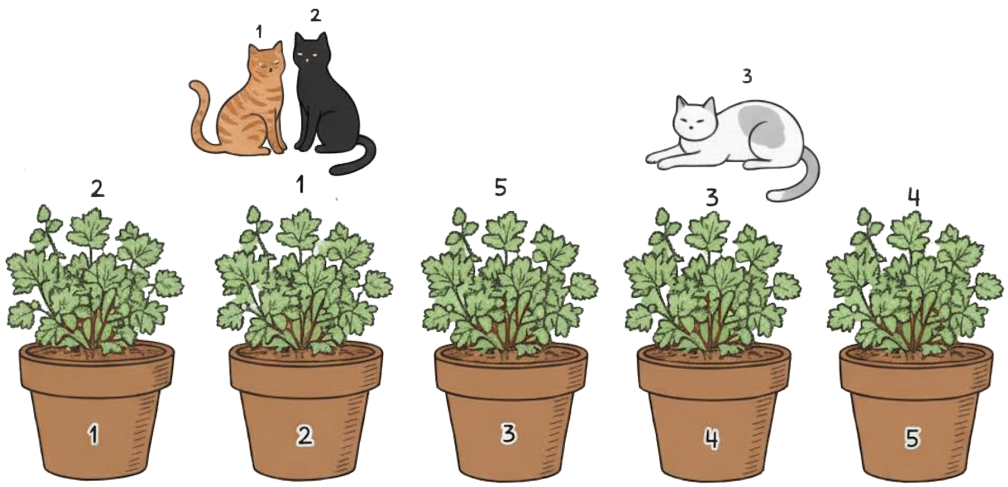

Figure F.1: One possible plant ordering for the first sample test case.

You know which pot you would like each cat to stop beside. Can you find a way in which to place the plants in the pots to achieve this?

Input

The first line of input contains an integer t ($1 \leq t \leq 10\,000$), which is the number of test cases. The descriptions of t test cases follow.

The first line of each test case contains two integers n and m, where n ($1 \leq n \leq 2 \cdot 10^5$) is the number of cats, and m ($1 \leq m \leq 2 \cdot 10^5$) is the number of catnip plants (and also the number of pots). Catnip plants are numbered from 1 to m.

The following n lines each describe one cat. The line starts with two integers p and k, where p ($1 \leq p \leq m$) is the pot at which the cat should stop, and k ($1 \leq k \leq m$) is the number of catnip plants the cat likes. The remainder of the line contains k distinct integers, which are the numbers of the plants that the cat likes.

Over all test cases, the sum of n is at most $2 \cdot 10^5$, the sum of m is at most $2 \cdot 10^5$, and the sum of all k is at most $5 \cdot 10^5$.

Output

For each test case, output either `yes` if it is possible to arrange the catnip plants as described above, or `no` if not.

Sample Input 1	Sample Output 1
2 3 5 2 2 1 5 2 3 1 4 5 4 2 3 4 3 5 2 2 1 5 2 3 1 4 5 5 2 3 4	yes no

Explanation of Sample 1: In the first test case, a possible ordering of the plants is $[2, 1, 5, 3, 4]$. This way, cat 1 will stop at pot 2, as it is the first pot with a plant variety that it likes. Cat 2 will stop there as well. Cat 3 will continue all the way to pot 4, as shown in Figure F.1.

Problem G
Lava Moat
Time limit: 4 seconds

These pesky armies of good are coming to disturb the quiet and peaceful lands of the goblins again. Building a huge wall didn't work out that well, and so the goblins are going to turn to the tried and true staple of defense: a moat filled with lava. They want to dig this moat as a boundary between the goblin lands in the north and the do-gooder lands in the south, crossing the whole borderlands west-to-east.

This presents them with a challenge. The borderlands are hilly, if not outright mountainous, while a lava moat has to be all on one level – otherwise the lava from the higher parts will flow down and out of the moat in the lower parts. So, the goblins have to choose a path that is all on one elevation, and connects the western border of the borderlands to its eastern border. For obvious economic reasons, they want this path to be as short as possible.

This is where you come in. You are given an elevation map of the borderlands, and your task is to determine how short the moat can be.

The map is in the form of a fully triangulated rectangle with dimensions $w \times \ell$, with all triangles having positive area. No vertex of a triangle lies on the interior of an edge of another triangle. The southwestern corner of the map has coordinates $(0, 0)$, with the x-axis going east and the y-axis going north. Furthermore, the western border (the line segment connecting $(0, 0)$ and $(0, \ell)$, including the endpoints) is a single edge. Similarly, the eastern border (between points $(w, 0)$ and (w, ℓ)) is also a single edge.

Of course, this map is just a 2D projection of the actual 3D terrain: Every point (x, y) also has an elevation z. The elevation at the vertices of the triangulation is directly specified by the map, and all of these given elevations are distinct. The elevation at all other points can be computed by linear interpolation on associated triangles. In other words, the terrain is shaped like a collection of triangular faces joined together by shared sides. These faces correspond to the triangles on the map.

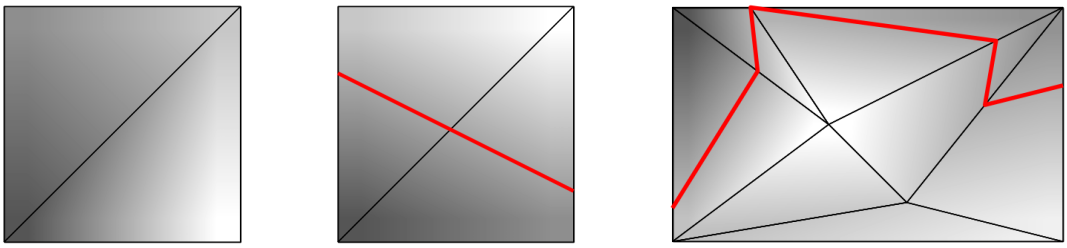

Figure G.1: Illustration of the sample test cases. Shading denotes elevation, and the thick red lines denote optimal moats.

Input

The first line of input contains an integer t ($1 \le t \le 10\,000$), which is the number of test cases. The descriptions of t test cases follow.

The first line of each test case contains four integers w, ℓ, n, and m, where w ($1 \le w \le 10^6$) is the extent of the borderlands from west to east, ℓ ($1 \le \ell \le 10^6$) is the extent from south to north, n ($4 \le n \le 50\,000$) is the number of vertices, and m ($n - 2 \le m \le 2n - 6$) is the number of triangles in the provided triangulation.

This is followed by n lines, the i^{th} of which contains three integers x_i, y_i, and z_i ($0 \le x_i \le w$; $0 \le y_i \le \ell$; $0 \le z_i \le 10^6$), denoting the coordinates and the elevation of vertex i. The only vertices with $x_i = 0$ or $x_i = w$ are the four corners. All pairs (x_i, y_i) are distinct. All z_is are distinct.

Each of the following m lines contains three distinct integers a, b, and c ($1 \le a, b, c \le n$), denoting a map triangle formed by vertices a, b, and c in counter-clockwise order. These triangles are a complete triangulation of the rectangle $[0, w] \times [0, \ell]$. Each of the n vertices is referenced by at least one triangle.

Over all test cases, the sum of n is at most $50\,000$.

Output

For each test case, if it is possible to construct a lava moat at a single elevation that connects the western border to the eastern border, output the minimum length of such a moat, with an absolute or relative error of at most 10^{-6}. Otherwise, output `impossible`.

Sample Input 1

```
3
6 6 4 2
0 0 1
6 0 4
6 6 3
0 6 2
1 2 3
1 3 4
6 6 4 2
0 0 1
6 0 2
6 6 4
0 6 3
1 2 3
1 3 4
10 6 7 7
6 1 8
10 0 10
10 6 4
2 6 6
0 6 0
4 3 11
0 0 7
2 1 7
2 3 1
3 6 1
3 4 6
6 4 5
5 7 6
7 1 6
```

Sample Output 1

```
impossible
6.708203932
15.849260054
```

Problem H
Score Values
Time limit: 2 seconds

Generated by ChatGPT

Ever since you arrived at your university, you have been a tireless advocate for introducing the brand-new martial-arts-plus-card-based sport of Contact Bridge to the school (and the world). Finally, after a great deal of (really persistent and annoying) advocacy on your part, you have obtained permission and funding from your dean to build a grand new arena for the sport! Well, technically it is not so much an "arena" as a "broom closet," and maybe not "grand" so much as "cramped," and the "new" is also debatable. But the sport of the future has to start somewhere!

Unfortunately, you just realized that you are going to need a score display in order to run the games. In Contact Bridge, the score for a team starts at 0 and, after various repeatable actions, may be incremented by certain fixed amounts. There is also a maximum value – if the team's score would be incremented above the maximum, it will instead be capped there. You want the team's score to be visible at all times, so you will need to prepare some signs, each with a single digit printed on it, that can be arranged to show the score.

Unfortunately the dean's "funding" is running short, and these signs are expensive. Figure out the minimum set of signs you need to purchase to show any score that is possible to achieve during the game. Note that you won't need any 9 signs, as any 6 sign can be turned upside-down to make a 9.

Input

The first line of input contains two integers m and n, where m ($1 \leq m \leq 10^{18}$) is the maximum score value, and n ($1 \leq n \leq 10$) is the number of different ways of scoring. This is followed by n lines, each containing an integer p ($1 \leq p \leq 1\,000$), which is the number of points awarded for a type of action in the game. No two types of action are awarded the same number of points.

Output

For each digit from 0 to 8 in increasing order, output two integers: the digit and the number of signs with that digit that you need to purchase. Omit digits where the number of signs needed is 0.

Sample Input 1	Sample Output 1
1000 4 60 100 222 650	0 3 1 1 2 3 3 1 4 3 5 1 6 3 7 2 8 3

Sample Input 2	Sample Output 2
967 1 1000	0 1 6 2 7 1

Problem I
Slot Machine
Time limit: 2 seconds

Imperial Chance & Play Casino offers games using a slot machine that has n wheels arranged next to each other. Each of the wheels has n distinct symbols on it, and these symbols appear in the same order on each wheel. Each wheel shows one of its symbols through a window on the front of the machine, which results in a sequence of n symbols being shown next to each other.

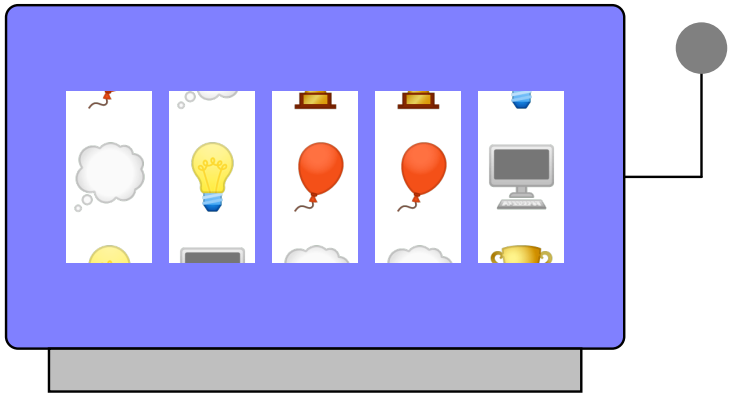

Figure I.1: The initial configuration in Sample Interaction 1.

You are standing behind the machine and notice that a maintenance panel has been left open. When you stick your hand inside, you are able to secretly rotate any of the wheels by any number of steps, thus changing the symbol shown on that wheel. You want to win a jackpot, which will happen if all the wheels show the same symbol at the same time. Unfortunately, you cannot see the symbols from your position, so you asked your good friend to help you. The friend is standing in front of the machine and she tells you the number of distinct symbols in the sequence she can currently see. Can you win the jackpot by manipulating the wheels if your friend updates the information after every action you make?

Interaction

The first line of input contains an integer n ($3 \leq n \leq 50$), giving the number of wheels and symbols in the machine.

Interaction then proceeds in rounds. In each round, one line of input becomes available, containing an integer k ($1 \leq k \leq n$), the number of distinct symbols in the current sequence. If $k > 1$, output two integers i and j ($1 \leq i \leq n$; $-10^9 \leq j \leq 10^9$), representing your action: rotating the i^{th} wheel by j positions, where negative numbers indicate rotating in the opposite direction. Otherwise, if $k = 1$, indicating that all wheels show the same symbol, your program must exit without printing more output.

At most 10 000 actions are allowed – if your submission uses more rounds, it will not be accepted. It is guaranteed that the initial configuration of wheels does not already have all wheels showing the same symbol ($k > 1$ in the first round).

The judge program will not behave in an adversarial way, which means the initial configuration is fixed before the first action.

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

Read	Sample Interaction 1	Write
5 4		
	1 1	
3		
	4 2	
3		
	3 1	
3		
	3 1	
2		
	5 4	
1		

Read	Sample Interaction 2	Write
3 3		
	2 -1	
2		
	3 -1	
2		
	2 -1	
1		

Problem J
Stacking Cups
Time limit: 2 seconds

You have a collection of n cylindrical cups, where the i^{th} cup is $2i - 1$ cm tall. The cups have increasing diameters, such that cup i fits inside cup j if and only if $i < j$. The base of each cup is 1 cm thick (which makes the smallest cup rather useless as it is only 1 cm tall, but you keep it for sentimental reasons).

After washing all the cups, you stack them in a tower. Each cup is placed upright (in other words, with the opening at the top) and with the centers of all the cups aligned vertically. The height of the tower is defined as the vertical distance from the lowest point on any of the cups to the highest. You would like to know in what order to place the cups such that the final height (in cm) is your favorite number. Note that all n cups must be used.

For example, suppose $n = 4$ and your favorite number is 9. If you place the cups of heights 7, 3, 5, 1, in that order, the tower will have a total height of 9, as shown in Figure J.1.

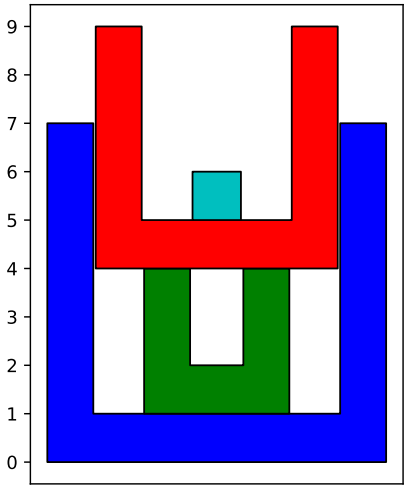

Figure J.1: Illustration of Sample Output 1.

Input

The input consists of a single line containing two integers n and h, where n ($1 \leq n \leq 2 \cdot 10^5$) is the number of cups and h ($1 \leq h \leq 4 \cdot 10^{10}$) is your favorite number.

Output

If it is possible to build a tower with height h, output the heights of all the cups in the order they should be placed to achieve this. Otherwise, output `impossible`. If there is more than one valid ordering of cups, any one will be accepted.

Sample Input 1	Sample Output 1
4 9	7 3 5 1

Sample Input 2	Sample Output 2
4 100	impossible

Problem K
Treasure Map
Time limit: 4 seconds

After years of searching you have come across Captain Blackbeard's old map showing where his long-lost treasure is hidden, deep on the ocean floor. The map was once a hypsometric map – that is, it showed the ocean depth for the region around the treasure – but many of the elevation marks have faded away over time and are no longer legible.

Specifically, the map covers a rectangular part of the ocean, subdivided into an $(n-1) \times (m-1)$ rectangular grid of unit squares. The map originally showed the ocean depth $d(p)$ for each point $p = (x, y)$ with integer coordinates $1 \le x \le n$ and $1 \le y \le m$. There are no islets in the region. In other words, it is known that $d(p) \ge 0$ for all points.

Preparing the map must have been quite a struggle for Blackbeard, since there is no unique natural way to interpolate the depths of points with non-integer coordinates. Consider a unit square on the grid, with corners at the grid points A, B, C, and D in clockwise order, and some depth $d(p)$ stored for each $p \in \{A, B, C, D\}$. One natural way is to interpolate the depth in the triangle ABC linearly, and likewise in CDA. Another equally natural way is to interpolate linearly within BCD, and likewise within DAB. Usually, the results of those two interpolations are different. For example, if $d(A) = d(B) = d(C) = 0$ and $d(D) = 1$, the first method results in depths across all of ABC being equal to zero (Figure K.1 left), while the second method results in the depths being positive in the whole interior of the square (right).

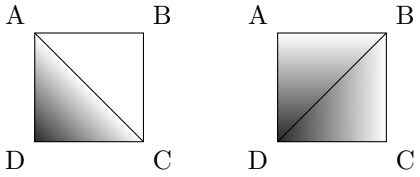

Figure K.1: Two ways of interpolating depths within a unit square.

However, Blackbeard was as stubborn as he was cruel and would not let such pesky ambiguities stop him. To find the perfect hiding spot for his treasure, he scoured the seven seas for a region of the ocean where the two methods described above yield the same results for each unit square (or maybe he forced some of his pirates to do a bit of terraforming work to achieve this – scholars disagree).

Back in the present, you are preparing an expedition to retrieve the treasure, and would like to figure out at what depth the treasure could be buried. Specifically, given the remaining depth data of the map, you should calculate the smallest possible depth at the treasure location.

Input

The first line of input contains five integers n, m, k, t_x, and t_y, where n and m ($2 \le n, m \le 3 \cdot 10^5$) denote the maximum coordinates of the grid, k ($1 \le k \le 3 \cdot 10^5$) is the number of known depths, and (t_x, t_y) is the location of the treasure ($1 \le t_x \le n$; $1 \le t_y \le m$). Each of the next k lines contains three integers x, y, and d ($1 \le x \le n$; $1 \le y \le m$; $0 \le d \le 10^9$), indicating that the depth at coordinate (x, y) of the grid equals d. Each pair (x, y) appears in the input at most once.

Output

If the provided data points can be extended to a valid map (that is, a map where, for each unit square, the two methods of interpolation yield the same results, and all points have non-negative depth), output one integer: the smallest possible depth of (t_x, t_y) – it can be shown that this is always an integer. Otherwise, output `impossible`.

Sample Input 1

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

Sample Output 1

```
3
```

Sample Input 2

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

Sample Output 2

```
1
```

Sample Input 3

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

Sample Output 3

```
0
```

Sample Input 4

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

Sample Output 4

```
impossible
```

Sample Input 5

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

Sample Output 5

```
impossible
```

Explanation of Sample 5: Even though the depth of $(2, 2)$ is given in the input, the provided data points cannot be extended to a valid map, so the correct answer is `impossible`.

Problem L
Walking on Sunshine
Time limit: 2 seconds

I'm walking on sunshine, and it don't feel good – my eyes hurt!

Baku has plenty of sunshine. If you walk away from the sun, or at least perpendicular to its rays, it does not shine in your eyes. For this problem assume that the sun shines from the south. Walking west or east or in any direction between west and east with a northward component avoids looking into the sun. Your eyes will hurt if you walk in any direction with a southward component.

Baku also has many rectangular areas of shade, and staying in these protects your eyes regardless of which direction you walk in. For example, Figure L.1 shows two shaded areas.

Find the minimum distance you need to walk with the sun shining in your eyes to get from the contest location to the awards ceremony location.

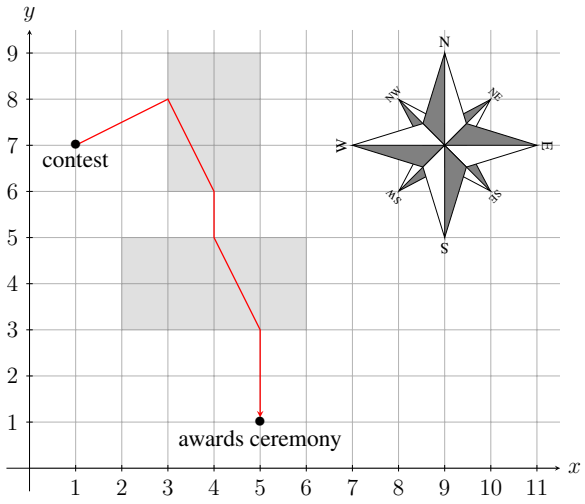

Figure L.1: Sample Input 1 and a path that minimizes the sun shining in your eyes.

Input

The first line of input contains five integers n, x_c, y_c, x_a, and y_a, where n ($0 \le n \le 10^5$) is the number of shaded areas, (x_c, y_c) is the location of the contest, and (x_a, y_a) is the location of the awards ceremony ($-10^6 \le x_c, y_c, x_a, y_a \le 10^6$). The sun shines in the direction $(0, 1)$ from south towards north. You look into the sun if you walk in direction (x, y) for any $y < 0$ and any x.

The next n lines describe the shaded areas, which are axis-aligned rectangles. Each of these lines contains four integers x_1, y_1, x_2, and y_2 ($-10^6 \le x_1 < x_2 \le 10^6$; $-10^6 \le y_1 < y_2 \le 10^6$). The southwest corner of the rectangle is (x_1, y_1) and its northeast corner is (x_2, y_2). The rectangles describing the shaded areas do not touch or intersect.

Output

Output the minimum distance you have to walk with the sun shining in your eyes. Your answer must have an absolute or relative error of at most 10^{-7}.

Sample Input 1	Sample Output 1
2 1 7 5 1 3 6 5 9 2 3 6 5	3.0

Explanation of Sample 1: Figure L.1 shows an optimal path from the contest location to the awards ceremony location with 5 segments. On the first segment you walk away from the sun. On the second and fourth segments you walk towards the sun but in a shaded area. On the third and fifth segments you walk towards the sun outside the shaded areas. The total length of these two segments is 3.

Sample Input 2	Sample Output 2
2 0 10 10 0 2 7 3 8 4 3 8 5	7.0

Sample Input 3	Sample Output 3
2 11 -1 -1 11 2 7 3 8 4 3 8 5	0.0

Sample Input 4	Sample Output 4
3 1 5 9 5 -5 6 2 9 4 7 12 8 1 1 7 3	0.0

Sample Input 5	Sample Output 5
3 1 7 9 3 2 6 3 8 4 4 5 6 6 2 7 4	0.0

Sample Input 6	Sample Output 6
1 0 0 0 0 -5 -5 5 5	0.0