Problem A. A day in the woods

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

People from the city really like to spend their weekends at the forester's range. A bus would take them to the foresters hut and theyd hike to forest clearings nearby to run barbecue parthes. But sometimes it happened so that they lost their way and roamed from one clearing to another, taking different trails.

Alex, the head forester, decided to help the silly citizens. He put numbers on all clearings and put arrows on trails so that a citizen walking from the forester's hut along the arrows, passing on to other clearings no more than once, would find himself on clearings with increasing numbers. If this citizen decided that he wants to return, hed have to go against the direction of the arrows. The decreasing numbers of the clearings would mean he is going in the right direction. After renumbering all the clearings should have different numbers from 1 to n.

Alex has a map of his forest where all the clearings and the trails linking the clearings are marked. Any two clearings can be linked with one or more trails. The forester put the arrows correctly, but now he is having trouble with the clearing numbers. Help him to forester put numbers on the clearings according to his plan.

Input

The input contains several testcases.

The first line of each test set contains three integers n, m, and s, — the number of clearings, the number of trails, and the number of the clearing where the forester's hut is located, respectively ($1 \le n \le 1000$, $0 \le m \le 1000$, $1 \le s \le n$). All clearings are enumerated with integers from 1 to n.

The next m lines describe the trails — each of them contains two integers, numbers of the clearings a trail connects in the direction of the arrow put by the forester. Note that loop trails are possible, i.e. in some cases these numbers can be equal.

The sum of n in all tests is not greater than 1000. The sum of m in all tests is not greater than 1000. The input ends with the line with three zeroes, which shouldn't be processed.

Output

For each test case print the numbering of the clearings in a separate line. If no acceptable numbering exists, the output must contain the phrase “`No solution`.

Example

standard input	standard output

3 3 1
1 2
2 3
1 3
4 4 4
4 1
1 2
2 3
3 1
4 6 1
1 2
1 3
1 4
2 3
3 4
4 2
0 0 0
	1 2 3
	2 3 4 1
	No solution

Problem B. Bus Stops

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

A core part of the decision of where to live in a city like Bytesburg is the availability of transport links to interesting places. This is particularly interesting to Barbara, who enlivens her stressful life as an organiser of programming camps by making frequent sightseeing travels around town in a yellow-blue bus.

Barbara's idea of a good time is a visit to a spot that takes exactly one bus journey to get to. She is considering moving herr house to be near one specific spot along her favourite bus route—how many other scenic spots can she reach from there (assuming that on a given trip she can choose a new bus route each time)?

Input

- The first line of input contains three integers: Barbara's starting stop, m $(1 \le m \le s)$. the number of buses, b $(1 \le b \le 50)$, and the number of stops, s $(1 \le s \le 50)$.
- The next b lines contain the bus routes, each written as a string of s characters where having the ith character as '`1`' denotes that this bus route has a stop at i, and '`0`' denotes that it does not.

Output

Output the maximum number of other stops Barbara can reach from the starting stop by taking exactly one bus.

Example

standard input	standard output

`1 3 5`
`01100`
`10011`
`10111`
	`3`

`2 2 3`
`101`
`101`
	`0`

Problem C. Campus for SIT

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Carol is the urbanist, and now she is working on the project of the new campus for Schlaffhausen Institute of Technology (SIT). Territory of the campus can be represented as n by m grid, with beautiful lakes in some cells and forest in another. Carol wants the campus buildings to fulfill the next conditions:

1. Buildings shall be put to forest cells only, no more than one building per forest cell.
2. To make campus compact enough, each forest cell must have a building directly on it, or adjacent to it.
3. For the environmental purposes no two buildings can be adjacent to each other.

Two cells are adjacent if they share a side.

Find any placement of buildings that satisfies these constraints.

Input

The first line of the input consists of two integers n and m ($1 \le n, m \le 100$). The following n lines each contain a string of length m consisting only of the characters '`F`' (forest) and '`L`' (lake). This is the map of the land for the campus. It is guaranteed that the map contains at least one forest cell.

Output

Output a copy of the map, where some of the land cells have been replaced with the letter '`B`', meaning that a building was placed on the corresponding land cell. This placement should satisfy the constraints above. If there are many solutions, any one will be accepted.

Examples

standard input	standard output
`5 6`	`BFFBLF`
`FFFFLF`	`FFBFLB`
`FFFFLF`	`LFFBFF`
`LFFFFF`	`FBFFFB`
`FFFFFF`	`LLBFBL`
`LLFFFL`	
`10 16`	`LLLLLLLLBLFBFBLL`
`LLLLLLLLFLFFFFLL`	`LLLLLLBFFFBFLLLL`
`LLLLLLFFFFFFLLLL`	`LLLLLBFBFBLLLLLL`
`LLLLLFFFFFLLLLLL`	`LLLBFFBFBFLLLLLL`
`LLLFFFFFFFLLLLLL`	`LLBFFBFBFLLLLLLL`
`LLFFFFFFFLLLLLLL`	`LLFBFFBFFBFLLLLL`
`LLFFFFFFFFFLLLLL`	`LLBFFBFBLLLLLLLL`
`LLFFFFFFLLLLLLLL`	`LLFBFFLLLLLLLLLL`
`LLFFFFLLLLLLLLLL`	`LLLFLLLLLLLLLLLL`
`LLLFLLLLLLLLLLLL`	`LLBFBLLLLLLLLLLL`
`LLFFFLLLLLLLLLLL`	

Problem D. Data Consistency

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Diana is a system administrator in the datacenter of the Byteland Astronomical Institute. The datacenter is planning to buy the special workstation for loseless real-time data communication with the satellites. Each satellite sends block of D mebibytes of the data to the Earth per session. All this data must be kept.

Workstation supports the SSD modules of S mebibytes each. Modules can be installed in multiple layers, one module per layer. When module is full, it must be immediately replaced with the empty one, but replacement takes some time and cause data loss, so no SSD module can be replaced until transaction of current block is finished.

To prevent data loss, Diana use very recent solution from Acronis called Acronis Infinite Space. It works in the next way. Each new block of the data is written to SSD in the first layer. When the SSD runs out of the free space and current block is still not received, last part of the block is written on the SSD in the second layer; if this SSD runs out of the free space while receiving this part, remaining part goes to the SSD in third layer and so on. When the block is received, all full SSD's are moved for the further processing and replaced with empty ones.

To perform the initial setup of the Acronis Infinite Space Diana wants to know minimal number of layers L needed to prevent the data loss.

Input

The input consists of a single line containing the two integers S and D ($1 \le D \le S \le 10^{10}$).

Output

Output the smallest integer L such that L layers wil be enough to prevent the data loss. If it is impossible to prevent the data loss with any number of layers, print -1 instead.

Example

standard input	standard output
31 6	4

Note

In the sample, at each 6'th session the first layer SSD is full and $6 \cdot 6 - 31 = 5$ mebibytes go to second layer SSD. At each $6 \cdot 7$ session the second layer SSD is full, and $7 \cdot 5 - 31 = 4$ mebibytes go to third layer SSD. At each $6 \cdot 7 \cdot 8$'th session the third layer SSD is full, and $8 \cdot 4 - 1 = 1$ mebibyte goes to fourth layer SSD. After each $6 \cdot 7 \cdot 8 \cdot 31$ time the fourth SSD is full right at the end of transaction, and no fifth SSD is needed.

Problem E. Evenly Divided

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 second
Memory limit:	512 mebibytes

The Byteland ICPC Community oversaw a sharp resurgence in membership this year, and must now face the inevitable strains of growth: the group photo they usually take can no longer fit everyone in one long row.

Edward, the photographer, asks members to split into two groups: Tall and Short, so that the picture can be doubled up with taller people standing behind shorter people in two rows of $\frac{n}{2}$ each.

The ICPC Community annual meetings is an opportunity for the members to meet people. Many new joiners were assigned a mentor from the members who had already signed up before they joined. And Edward wants to choose a way of arranging the rows such that nobody is standing directly in front of or behind their mentor, assuming they have one.

Find a way of arranging the two rows such that this is possible. The number of tall people is always the same as the number of short people.

Input

The input consists of:

- a line consisting of the number of members in the mountaineering society, which is a positive even integer m $(1 \le m \le 10^5)$.
- m further lines, with the ith line $(1 \le i \le m)$ consisting of an integer indicating whether the ith member is short (0) or tall (1), then the number of the ith member's mentor, t_i $(0 \le t \le m)$. When $t_i = i$, this indicates that the ith member did not have a mentor.

Output

If an arrangement is possible, output 2 lines of $\frac{n}{2}$ numbers each to show which member should stand where.

Every number of type 1 should occur somewhere on the first row, and every number of type 0 should occur somewhere on the second row. Nobody should share a column with their mentor.

Otherwise, output `impossible`.

Examples

standard input	standard output

4
0 1
1 1
1 2
0 3
	3 2
	1 4

4
0 1
1 1
0 1
1 1
	impossible

10
0 1
1 1
1 1
1 1
0 1
0 4
0 6
1 1
0 7
1 2
	10 8 3 2 4
	1 6 7 9 5

Problem F. Formula-1

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Those of you, who watch Formula-1 races, know that for several seasons Acronis supports one of teams: they started with Toro Rosso and now they are IT partners of Williams team.

Traditionally, the information for the drivers is shown by teams using special boards with big letters on it. Usually its time gap to the next (or previous) driver. Sure, this information is sent to the onboard electronics by the radio channel... but sometimes radio does not work properly.

Farid, one of the engineers of Williams come to Acronis with brilliant idea: let in this cases the video recognition software does all the work and reads the data from the tables.

Important part of this module is the recognition of the single letters. Your task is to implement it.

Input

First line of the input contains one integer n ($1 \le n \le 100$) — number of the possible letters.

Then the n letter descriptions come. Description of i-th letter starts with the two integers h_i and w_i ($1 \le h_i, w_i \le 200$) — numbers of rows and columns in the letter shape. Each of the next h_i lines contains exactly w_i characters. '`.`' means that pixel is white, while '`X`' means that pixel is black. It is guaranteed that each letter is 4-connected, i.e. between any two '`X`' pixels exists a path consisting entirely from '`X`' pixels where two consecutive pixels share a side, and that any two letters are pairwise distinct and cannot be transformed one to another by a rotation of multiple of 90 degrees.

Then come description of the picture of the board with message. The first line of the description contains two integers h and w — number of rows and columns of the picture. Each of the next h lines contains exactly w characters '`.`' and '`X`'. You may assume that:

- Each '`X`' pixel in the messages are parts of exactly one letter.
- Letters does not overlap neither touch by sides (but they may touch diagonally).
- All the letters in the message are the letters described in the previous section; they may be rotated by 0, 90, 180 or 270 degrees.

Output

For each letter in order they are listed in the input data, print number of its occurences in the message.

Examples

standard input

```
2
5 8
....XXXX
...XX.XX
..XX..XX
.XXXXXXX
XX....XX
5 6
XXXXXX
..XX..
..XX..
..XX..
..XX..
11 13
........X....
........X....
........XXXXX
........XXXXX
........X....
........X....
....XXXX.....
...XX.XX.....
..XX..XX.....
.XXXXXXX.....
XX....XX.....
```

standard output

```
1 1
```

standard input

```
2
3 1
X
X
X
5 7
.XXXXX.
XX...XX
X.....X
XX...XX
.XXXXX.
5 7
.XXXXX.
XX...XX
X.XXX.X
XX...XX
.XXXXX.
```

standard output

```
1
1
```

Problem G. Generator

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Gregor is an engineer. He now is working on the project of new generator for water power plants. One of important parts of generators is a water wheel.

A water wheel is a device for converting the potential energy of water into the kinetic energy of rotation. We have attached buckets along our otherwise perfectly-balanced wheel. Each bucket has a specific weight when empty, and a specific weight when full of water, both in kilograms.

Whenever a bucket reaches the top of the water wheel, it is filled to its maximum capacity. When the bucket reaches the bottom, diametrically opposite, it is emptied again.

The acceleration of the wheel depends on the horizontal distance between the centre of mass of the wheel and its axis of rotation. Centre of mass of the wheel at some time is defined as the sum of the co-ordinates of the buckets at that time multiplied by their weight.

Our water wheel has a radius of exactly 1 metre.

To calculate effectivity of his construction Gregor wants to know maximum positive x-component of centre of mass achieved by the wheel at any angle. Help him to find it.

Input

- The first line of input contains the number of buckets on the wheel, n ($2 \le n \le 10^5$).
- The remaining n lines of input each contain a description of a bucket, in angular order:
  - the decimal clockwise angle of the ith bucket in degrees, d_i ($0.0 \le d \le 360.0$),
  - the decimal weights of this bucket when empty and full, e_i and f_i ($0 \le e_i \le f_i \le 10^4$).

Output

Output the maximum horizontal component of the centre of mass of the wheel over all possible angles. Your answer should be within an absolute or relative error of at most 10^{-6}.

Examples

standard input	standard output

9
0.000 0.5 1.8
22.50 1.0 1.7
90.00 0.5 1.0
115.0 1.0 1.2
135.0 1.0 1.2
180.0 1.0 1.2
225.0 1.0 1.2
270.0 1.0 1.2
295.0 1.0 1.2
	1.65664252966349700991

Note

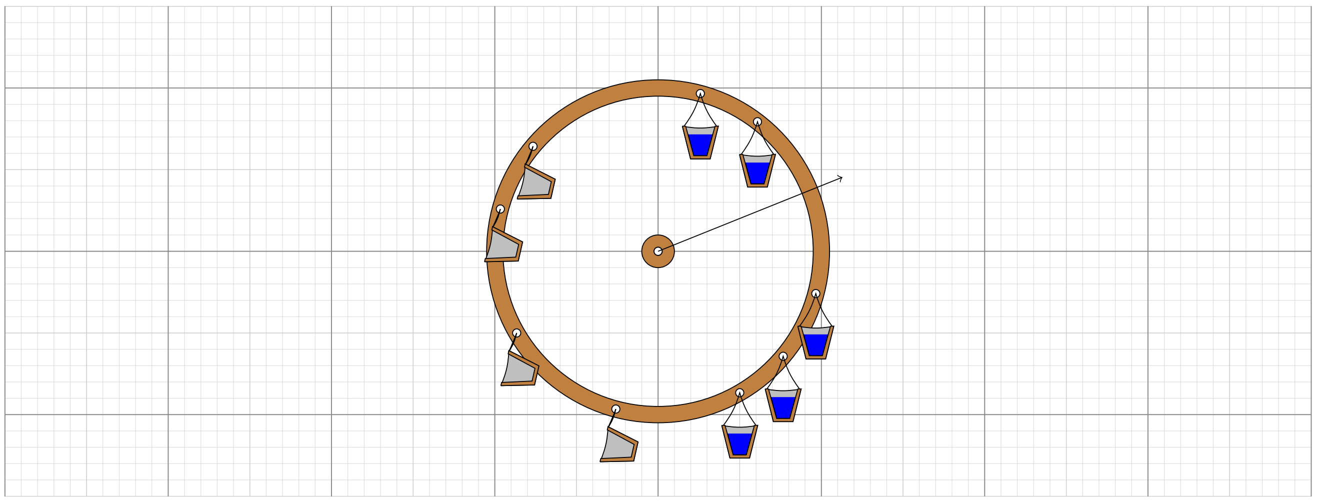

Illustration of Sample Input 1 at approximately 15.0 degrees clockwise from the start. The centre of mass is drawn as a vector from the origin.

Problem H. Hacker's Heist

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Hatiko is a skilled hacker and works for the Byteland Intelligence Service.

She tricked her way into the server of Berland embassy through their local WiFi and managed to copy the files the informant told her about. Unlickily, he Berland authorities caught the informant, so they know what Hatiko did and sealed off the city. Hatiko need to upload the data to the internet before they catch her.

Luckily the city is full of coffee shops with free wifi and Hatiko researched their details before she started her heist. The only downside is that they have strange opening hours and she may need to move between them. When travelling between coffee shops Hatiko can't upload any data at all.

Given the list of coffee shops, their opening hours and Wifi speed, what is the earliest time when all of important data can been uploaded?

Input

- The first line contains an integer d ($1 \le d \le 10^9$), the size of the data in megabytes.
- The second line contains an integer n ($1 \le n \le 100$), the number of cafes.
- The next n lines each contain:
  - The integer opening and closing times of the cafe in seconds, o and c ($0 \le o \le c \le 24 * 60 * 60$).
  - The wifi speed of the cafe w ($1 \le c \le 1000$) in megabytes per second.
- n lines, each with n integers, the time to travel to each cafe from the n^{th} cafe, in seconds ($0 \le d_i \le 24{*}60{*}60$).

Hatiko may start in any single cafe when it opens and start using wifi the moment she arrives at a cafe.

Output

One line containing an integer number of seconds, the smallest integer time by which all of the data can be uploaded.

Examples

standard input	standard output

200
2
10 20 15
10 40 5
0 5
5 0
	35

Problem I. Interesting Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Igor and Ilona, both enjoying playing games very much, have discovered a new type of game: tree game. In this game, first player chooses a vertex of a tree. Then players (beginning with second player) alternately choose a neighboor of the last chosen vertex that was not chosen before, until a player can't make a move. This player is then declared loser and the other one is the winner. Igor moves first but unfortunately Ilona a is very experienced player and she will never make a mistake. Therefore Igor has asked you for help.

Our tree has N vertices conventionally numbered $1 \dots N$ and exactly $N-1$ edges connecting them. Write a program, which determines all vertices in which Igor can start the game and win despite Ilona playing perfectly.

Input

On the first line, there is a single number N, $(1 \leq N \leq 2 \cdot 10^6)$, which is equal to the number of nodes in the tree. $N-1$ lines follows, on the i-th line is a single integer a_i $(1 \leq a_i \leq i)$, which means there is edge in the tree connecting the vertex $i+1$ with the vertex a_i.

Output

Output consists of several lines, on each one there is a single number of a node, where Igor can start the game and win, regardless of how Ilona would play. Numbers in the output should be sorted in ascending order.

Examples

standard input	standard output
3 1 1	2 3
6 1 1 1 4 5	2 3 4 6

Problem J. Jackpot

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

The thunderingly fun new game show of 'Jackpot' has bounded onto televisio screens up and down the land and is the next big thing. It runs along a simple premise: there are a number of doors, behind one of which is hidden a monetary prize and behind all of the others of which are hidden goats. Each contestant chooses how many doors will be opened before they pick a door to look at. The prize decreases as each door is opened. If they manage to open the door and find the money they get to keep it!

Not content with guessing, Jessica decided to try and work out the best number of doors to open to maximise her expected profit, where expected profit is the probability of winning multiplied by the remaining prize fund.

Jessica managed to find out that the remaining prize fund P_R is a function of the initial prize fund m, the number of doors opened d and a scaling factor f, ie.

$$P_R = m - (d \cdot f)^2$$

Given the number of doors, initial prize fund and the scaling factor can you work out how many doors should be opened before Jessica will try to pick a door to maximise her expected profit?

Input

One line of input containing:

- One integer n $(1 \le n \le 10^{18})$, the number of doors.
- One integer m $(1 \le m \le 10^{18})$ the initial prize fund.
- One floating point f $(0 < f \le 1)$, the scaling factor.

There will always only be one door with the best value.

Output

One line containing one floating point number, the value of the expected payout. The output must be accurate to an absolute or relative error of at most 10^{-6}.

Example

standard input	standard output
`4 100 1`	`91.0`
`20000 100500 0.8`	`5.0254931`

Problem K. Kings

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Kevin inventerd the new game, called "chess-tidying". The game starts with a messy board containing several king pieces strewn across the cells. The player's job is to put all of the pieces into a line along the primary diagonal of the board, which runs along black squares.

In each move, unlike in normal chess, one king may be moved one place either horizontally or vertically (but not both) to another unoccupied cell.

Kevin wants, given an instance of the game, to know how many moves you will need in order to finish it. Help him to do that

Input

- One line with the number of rows and columns, n ($1 \le n \le 500$).
- Each of the following n lines contains the two-dimensional integer coordinates c and r ($1 \le c, r \le n$), the position of one of the kings.

Each of the kings starts at a unique position.

Output

Output the minimum number of moves necessary to cover the main diagonal ($r = c$) with kings.

Examples

standard input	standard output

3
1 1
2 3
3 2
	2

8
6 4
6 8
5 5
5 4
4 8
5 7
7 4
3 7
	28

Note

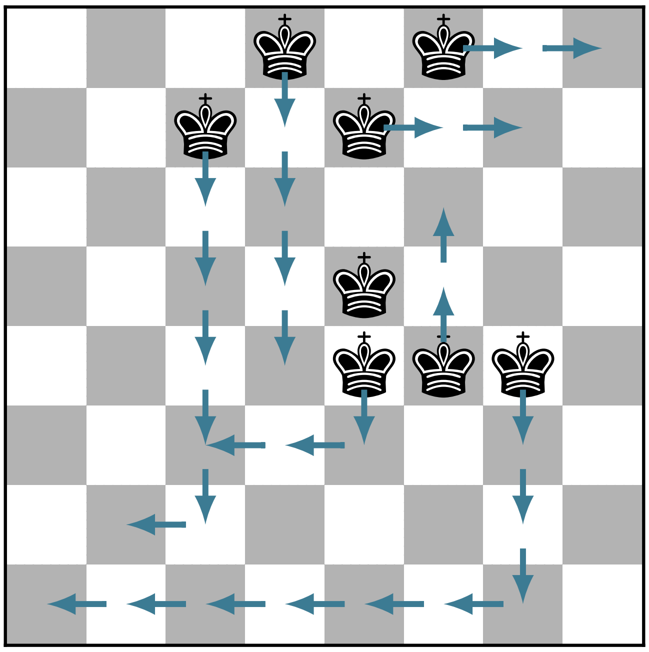

Illustration of Sample Input 2. In this case, the minimum number of moves necessary to put all of the kings along the black diagonal is 28 as pictured.