Problem A. Absenteeism

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds (4 seconds for Java)
Memory limit:	512 mebibytes

> Absenteeism is a habitual pattern of absence from a duty or obligation without good reason.
>
> Wikipedia

Alex works at work. n employees work together with him. Unlike Alex, they strictly follow their schedule: the i-th employee comes to work at the time moment a_i and leaves at the time moment b_i ($a_i < b_i$).

Alex has to come to work not earlier the time moment 0, leave not later the time moment m and work for k hours a day ($k \leq m$). But, of course, he wants to work as less as possible. The problem is, his colleagues don't like lazy people, and if one of them discovers that Alex works less than k hours a day, they complain to the boss and Alex will be fired.

In other words, let's say Alex comes to work at the time moment x and leaves work at the time moment y, where $0 \leq x < y \leq m$. Then Alex is in danger if at least one of the following conditions is satisfied:

- $y - x < k$, and there exists an employee i such that $a_i \leq x$ and $y \leq b_i$ (the segment $[x, y]$ is entirely nested into $[a_i, b_i]$) — in this case the i-th employee sees how Alex works less than k hours with their own eyes;
- there exists an employee i such that $x < a_i \leq y \leq b_i$, and $y < k$ — in this case the i-th employee sees how Alex leaves work earlier than the time moment k;
- there exists an employee i such that $a_i \leq x \leq b_i < y$, and $x > m - k$ — in this case the i-th employee sees how Alex comes to work later than the time moment $m - k$;
- there exists an employee i such that $y < a_i$ or $b_i < x$ (the segments $[x, y]$ and $[a_i, b_i]$ don't intersect), and $a_i \leq k$ and $b_i \geq m - k$ — in this case the i-th employee doesn't see Alex at work at all, but it's exactly the reason they conclude Alex couldn't work for k hours.

What is the minimal time Alex can spend at work such that nobody of his colleagues complains to the boss?

Input

The first line contains three integers n, m, k ($1 \leq n \leq 4000$, $1 \leq m \leq 10^9$, $1 \leq k \leq m$) — the number of employees working together with Alex, the day length and the number of hours Alex has to work in a day.

Each of the following n lines contains two integers a_i and b_i ($0 \leq a_i < b_i \leq m$) — the moments of time when the i-th employee comes to work and leaves work.

Output

If Alex is able to not come to work at all, output "`-1 -1`".

Otherwise, output two integers x, y ($0 \leq x < y \leq m$, $y - x \leq k$) — the moments of time when Alex should come to work and leave work, so that nobody of his colleagues complain to the boss and the working time is minimal.

If there are several possible solutions, output any of them.

Examples

standard input	standard output

2 20 10
0 12
8 20
	7 13

2 20 18
0 10
10 20
	2 18

Problem B. Fakes and Shidget

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds (4 seconds for Java)
Memory limit:	512 mebibytes

Pavel loves the game Fakes and Shidget very much. The game literally consists of the following process. The player uniformly randomly meets one of n characters. Every character offers the player to choose one of two quests. The first quest of the i-th character requires a_i minutes to complete and brings b_i gold, and the second quest requires c_i minutes and brings d_i gold. The player chooses one of these quests, completes it and immediately meets another random character, and so on.

Pavel will play this game infinitely long. How fast can he earn gold if he will play optimally?

More formally, let t is the time Pavel plays this game, and $g(t)$ is the amount of gold he earns for the time t. You should find the limit $\lim\limits_{t\to\infty} \frac{g(t)}{t}$.

Input

The first line contains an integer n ($1 \le n \le 200000$) — the number of characters in the game.

Each of the next n lines contains four integers a_i, b_i, c_i and d_i ($1 \le a_i, b_i, c_i, d_i \le 10^9$) — the duration of the first quest, the reward for the first quest, the duration of the second quest, the reward for the second quest of the i-th character.

Output

Output one floating point number — the maximal possible speed of earning gold.

The absolute or relative error of the answer shouldn't exceed 10^{-9}.

Examples

standard input	standard output

2
1 10 10 70
1 1 10 20
	6.454545454545455

2
1 20 100 100
2 1 2 1
	7.000000000000000

Problem C. Cyclically Shifted Maze

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Let's call a maze a rectangular field of cells, where cells can either be empty or contain a wall, and one can move from an empty cell to its empty neighbour cells in four directions.

Let's call a maze connected if it's possible to reach any its empty cell from any other empty cell by moving in four directions.

There was a connected maze of size $n \times m$. It was cyclically shifted some rows down and some columns right, but nobody knows the exact shifts. Find all possible shifts.

Input

The first line contains two integers n and m $(1 \le n, m \le 200)$ — the sizes of maze.

Each of the next n lines contains m characters "`.`" or "`#`" — empty cells and walls, correspondingly.

There is at least one empty cell in the maze.

Output

In the first line output a single integer k $(0 \le k \le n \cdot m)$ — the number of possible shifts.

In each of the next k lines output two integers r_i and c_i $(0 \le r_i < n, 0 \le c_i < m)$ — the number of rows the original maze was shifted down and the number of columns it was shifted right. Pairs (r_i, c_i) should be output in lexicographical order. Original maze must be connected for each of these cases.

Examples

standard input	standard output

5 6
..####
.###..
...#.#
##...#
.###..
	9
	0 2
	0 3
	0 4
	1 2
	1 3
	1 4
	4 2
	4 3
	4 4

8 10
##########
....
#.####..##
..###..##.
#....##...
######..##
....###...
....###...
	2
	0 5
	1 5

Problem D. Two Pirates - 2

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

Two pirates are dividing looted treasures. There are n treasures, the i-th of which costs a_i gold. They move in rotation, each turn a pirate picks one of the remaining treasures. The thing is, the second pirate is drunk, so he doesn't make optimal picks and each turn just picks a random available treasure, uniformly. The first pirate knows that and always makes the optimal picks.

Find the expected costs of treasures picked by both pirates.

Input

The first line contains an integer n ($1 \le n \le 500$) — the number of treasures.

The second line contains n integers a_i ($1 \le a_i \le 10^9$) — the costs of the treasures.

Output

Output two floating point numbers: the expected costs of treasures picked by the first and the second pirates.

Absolute or relative error of the numbers shouldn't exceed 10^{-9}.

Examples

standard input	standard output
2 1 3	3.000000000000000 1.000000000000000
3 2 1 4	5.500000000000000 1.500000000000000

Problem E. Powerless Mage

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

A mage knows n spells, the i-th of which requires q_i blue mana, w_i purple mana and e_i orange mana ($q_i + w_i + e_i > 0$). Let's say the mage has Q blue mana, W purple mana and E orange mana, and denote the total amount of mana $Q + W + E$ as R. Knowing that the mage is unable to cast any spell, what can be the maximal value of R?

Input

The first line contains an integer n ($1 \le n \le 5000$) — the number of spells.

Each of the next n lines contains three integers q_i, w_i, e_i ($0 \le q_i, w_i, e_i \le 10^9, q_i + w_i + e_i > 0$) — the manacosts of the i-th spell.

Output

Output the maximal total amount of mana R such that the mage is unable to cast any spell.

If this number is infinitely large (for any total amount of mana R, there could be a situation so that the mage is unable to cast any spell), output "`Infinity`".

Examples

standard input	standard output

4
0 0 100
0 100 0
100 0 0
61 71 81
	278

6
0 0 100
0 100 0
100 0 0
0 11 61
11 61 0
61 0 11
	180

3
3 1 1
1 3 1
1 1 3
	Infinity

Problem F. Exactly One Point

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds (4 seconds for Java)
Memory limit:	512 mebibytes

There are n segments on a line. You should place some points onto this line so that:

- every point is contained in at least one segment,
- every segment contains exactly one point.

Input

The first line contains an integer n ($1 \le n \le 5000$) — the number of segments.

Each of the next n lines contains two integers L_i and R_i ($L_i < R_i$) — the endpoints of the i-th segment.

For convenience, all endpoints of all segments are even numbers from 0 to $4n - 2$.

Output

If it's impossible to place points in a required way, output "`-1`".

Otherwise, in the first line output an integer m — the number of points that should be placed onto the line.

In the next line, output m distinct integers from 0 to $4n - 2$ — the coordinates of the points.

You don't have to minimize the number of points. If there are several possible solutions, output any of them.

Examples

standard input	standard output

3
0 10
2 4
6 8
	-1

2
0 6
2 4
	1
	4

3
0 2
2 4
4 6
	3
	1 3 6

Problem G. Lexicographically Minimal Subsequence

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

Your are given a string s and an integer k. Find the lexicographically minimal subsequence of s which length is k.

Input

The first line contains a string s ($1 \le |s| \le 10^6$). It consists of lowercase Latin letters.

The second line contains an integer k ($1 \le k \le |s|$) — the length of the resulting subsequence.

Output

Output the lexicographically minimal subsequence of s which length is k.

Example

standard input	standard output

`bcaabac`
`4`
	`aaac`

Note

String $s_{p_1} s_{p_2} \dots s_{p_k} (1 \le p_1 < p_2 < \dots < p_k \le |s|)$ is called a *subsequence* of string s.

String $x = x_1 x_2 \dots x_k$ is *lexicographically less* than string $y = y_1 y_2 \dots y_k$, if there exists such number i ($1 \le i \le k$), that $x_1 = y_1, x_2 = y_2, \dots, x_{i-1} = y_{i-1}$ and $x_i < y_i$. Characters in strings are compared as their ASCII codes.

Problem H. Video Reviews - 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds (5 seconds for Java)
Memory limit:	64 mebibytes

> Easy problem, just binary search the answer. Oh wait...
>
> You

The studio "Lodka Gaming" is engaged in advertising of their new game ".C.O.N.T.E.S.T: Unexpected Behaviour". The studio's marketer is planning to communicate with n videobloggers one by one (*in the predetermined order, starting from the 1-st and ending with the n-th*), offering them to record a video review on the game. All people are different and videobloggers are as well, therefore the i-th videoblogger will record a review in two cases: either he is interested in this game, or there are already at least a_i video reviews on this game.

The studio wants to have at least m video reviews in the Internet. The game designer of "Lodka Gaming" understands these video reviews possibly would not appear by themselves, so he wants to *convince* some video bloggers that they are actually interested in this game. Which minimal number of videobloggers are needed to be *convinced*?

Input

The first line contains two integers n and m ($1 \le n \le 5 \cdot 10^7, 1 \le m \le n$) — the number of videobloggers and the required number of video reviews.

As n can be too large, the a_i values will be generated by linear congruential random number generators.

The second line contains two integers a_1 and k ($0 \le a_i \le 5 \cdot 10^7, 0 \le k \le 10^5$).

Each of the following k lines contains 4 integers c_j, x_j, y_j and z_j ($1 \le c_j < 5 \cdot 10^7, 1 \le z_j \le 5 \cdot 10^7, 1 \le x_j < z_j, 0 \le y_j < z_j$). z_j are prime numbers. All a_i, except the first one, will be generated using these numbers. The first c_1 of them will be generated using the formula $a_i = (x_1 \cdot a_{i-1} + y_1) \mod z_1$, the next c_2 — using the formula $a_i = (x_2 \cdot a_{i-1} + y_2) \mod z_2$, and so on. It is guaranteed that the sum of all c_j is $n - 1$.

Output

Output a single integer — the minimal number of videobloggers who have to be *convinced* to record a video review on the game in order to achieve at least m total video reviews in the Internet.

Examples

standard input	standard output

7 4
2 6
1 1 49999990 49999991
1 1 2 49999991
1 1 0 49999991
1 1 1 49999991
1 1 49999989 49999991
1 1 1 49999991
	1

7 4
2 5
1 1 96 97
1 1 2 97
1 1 0 97
1 1 1 97
2 1 96 97
	2

Note

In the first sample, $a = [2, 1, 3, 3, 4, 2, 3]$.

In the second sample, $a = [2, 1, 3, 3, 4, 3, 2]$.

Problem I. Chess Tournament

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

Alex organizes a chess tournament in his company. The tournament is a round-robin tournament of n people, and each pair of players will face each other exactly once.

The problem is, there are only k chess boards in the office ($k \leq \frac{n}{2}$). So only k games can be played at the same time. Let's call g games, being simultaneously played by $2g$ distinct players, where $1 \leq g \leq k$, a round.

Your task is to help Alex to set a schedule with a minimal number of rounds.

Input

The input contains two integers n and k ($2 \leq n \leq 200, 1 \leq k \leq \frac{n}{2}$) — the number of players and the number of chess boards.

Output

In the first line output an integer r — the number of rounds.

Then output r sections, describing rounds. In the first line of each section, output an integer g ($1 \leq g \leq k$) — the number of games in this round. Then output g lines with two integers each — the pairs of players that will play in this round. All these $2g$ integers must be distinct integers from 1 to n.

If there are several possible solutions, output any of them.

Examples

standard input	standard output

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

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

6 2	8
	2
	3 2
	6 4
	2
	5 1
	4 3
	2
	6 1
	2 5
	2
	1 3
	5 6
	2
	2 4
	5 3
	2
	4 1
	2 6
	2
	6 3
	5 4
	1
	2 1

Problem J. Lost Island

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

There is an island far away occupied by a tribe. Among people in this tribe, n eye colors occur: there are a_i people with the eye color i ($a_i > 0$), and there are no other colors known to the tribe. More formally, the members of the tribe know the value of n, but they don't know that all eye colors actually occur (that all $a_i > 0$). The tribe follows a specific religion: if someone can deduce their own eye color, they commit suicide the next day. It must be said, the people in the tribe are incredibly smart.

Once, at day 0, a traveler arrives to the island, meets the tribe and says n true sentences, in which $b_i \geq 0$, and at least one $b_i > 0$:

- Wow, there are at least b_1 people among you with the eye color 1!
- Wow, there are at least b_2 people among you with the eye color 2!
- ...
- Wow, there are at least b_n people among you with the eye color n!

Find the last day when the suicides will take place and the total number of people committed suicide.

Input

The first line contains an integer n ($2 \leq n \leq 200000$) — the number of eye colors.

Each of the next n lines contains two integers a_i and b_i ($1 \leq a_i \leq 10^9, 0 \leq b_i \leq a_i$, at least one $b_i > 0$) — the number of people with the eye color i and the lower bound of this number said by the traveler.

Output

Output two integers — the number of the last day when the suicides will take place, and the total number of people committed suicide.

Examples

standard input	standard output

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

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

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

Note

Let's show what happens in the first sample.

The person with the eye color 1 doesn't see anyone with the eye color 1 around, but hears that there is at least one person with this color. So they deduce who can be this person and commit suicide at the day 1.

All other people know that only two colors occur and that one person with the eye color 1 is dead. So they deduce all of them have eye color 2 and commit suicide at the day 2.

Problem K. Bloodseeker

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

Bloodseeker is facing n enemies. At the beginning, he has m hit-points, and every second his hit-points are decreased by 1. If his hit-points become 0, he dies. But he can kill the enemies to regenerate his hit-points.

The i-th enemy is to be hit t_i times to kill. Bloodseeker makes one hit per second. Every second, he is able to hit any enemy. After the i-th enemy receives a last hit, Bloodseeker regenerates h_i hit-points (but his hit-points can't become greater than m). Note that if Bloodseeker had 1 hit-point before he last-hits the i-th enemy, he doesn't die.

Can Bloodseeker kill all enemies?

Input

The first line contains an integer T ($1 \le T \le 200000$) — the number of test cases.

The first line of each test case contains two integers n and m ($1 \le n \le 200000, 1 \le m \le 10^9$) — the number of enemies and the maximal Bloodseeker's hit-points.

Each of the next n lines in each test case contains two integers t_i and h_i ($1 \le t_i, h_i \le 10^9$) — the time required for killing the i-th enemy and the number of hit-points regenerated after it.

It is guaranteed that the sum of all n does not exceed 200000.

Output

For each test case, if it's possible to kill all the enemies, output "YES", otherwise output "NO".

Example

standard input	standard output

4
2 10
7 3
6 1
2 10
7 3
7 1
3 10
5 7
5 7
14 1
3 10
5 7
5 7
15 1
	YES
	NO
	YES
	NO

Problem L. Not the Longest Increasing Subsequence

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

There is an array of n integers. Each element a_i in this array is between 1 and k.

What is the smallest number of elements that should be removed from this array, so that its longest increasing subsequence has length smaller than k?

Input

The first line contains two integers n and k ($1 \le n \le 10^5, 1 \le k \le n$) — the length of the array and the upper bound for its elements.

The second line contains n integers a_i ($1 \le a_i \le k$) — the elements of the array.

Output

In the first line output an integer m — the number of elements to remove.

In the second line output m integers — the indices of the removed elements. The indices are numbered from 1 to n.

Examples

standard input	standard output
3 2 1 2 2	1 1
2 2 2 1	0
8 3 1 2 2 1 1 3 2 3	2 1 7

Problem M. Binary Search Tree

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds (3 seconds for Java)
Memory limit:	512 mebibytes

> How to check if a tree is a binary search tree?
>
> Someone in a Telegram chat

Binary search tree is a rooted tree, in which:

- each vertex can have at most one left child and at most one right child,
- for each non-leaf vertex x, all vertices in its left subtree are less than x. and all vertices in its right subtree are greater than x.

You are given a tree with n vertices. Can this tree, being rooted at some vertex, be a binary search tree, and if it can, what vertices can be a root?

Input

The first line contains an integer n ($1 \le n \le 500000$) — the number of vertices in the tree.

Each of the next $n-1$ lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$) — the edges of the tree.

Output

If this tree can't be a binary search tree, output "`-1`".

Otherwise, output all vertices that can be a root, in increasing order.

Examples

standard input	standard output

3
1 2
2 3
	1 2 3

3
1 3
3 2
	1

4
1 3
3 2
2 4
	-1

4
1 2
1 3
1 4
	-1

Problem N. Premove Checkmate

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

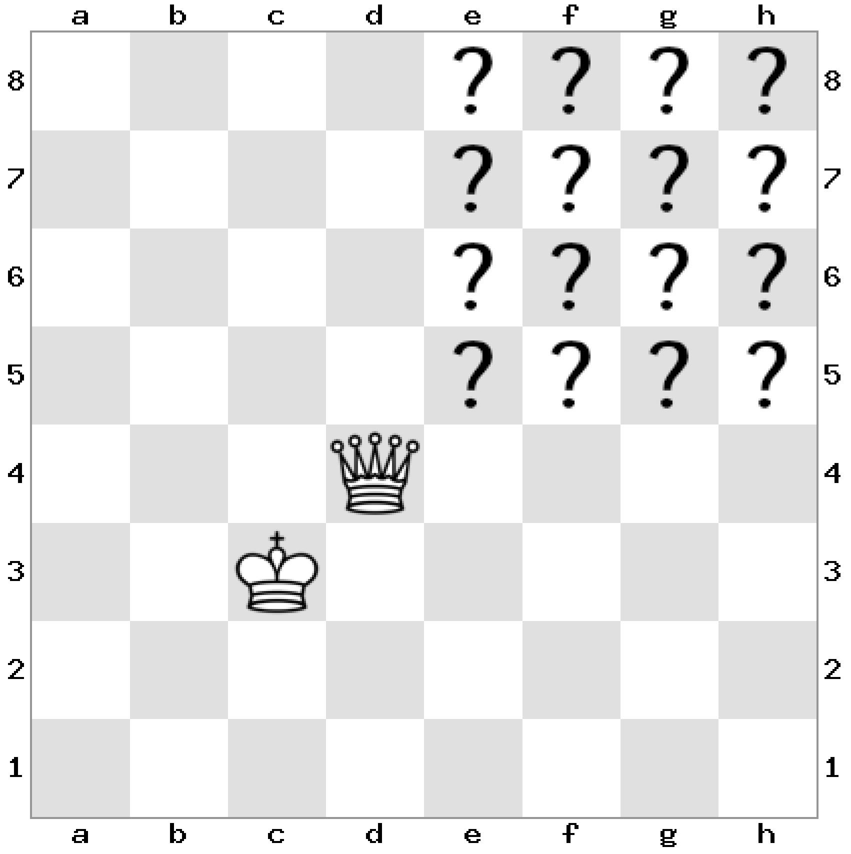
Be careful, the premove description in this problem is not the same as used in a popular chess websites.

You play chess in the internet. You got a position with a king and a queen versus alone opponent's king. The problem is, you don't have enough time on the clock not only to checkmate the opponent, but to make even one move.

Luckily, there is a premove function. The premove mechanic works as follows. You enter a sequence of moves on the board (these can be arbitrary moves, described just by starting and ending squares, it's not necessary that your piece should stay on the starting square). These moves are stored in a queue. You can add moves to the queue only when it's your opponent move.

Then, after your opponent makes a move, if your premove queue is not empty, the moves are popped out one by one. If the popped move is invalid, it's just skipped, and once a popped move is valid, it's done on the board immediately, and your clock time doesn't get decreased.

In the current position, your king is on c3, your queen is on d4, and the opponent's king is on some square in the top-right quarter of the board (e5, e6, e7, e8, f5, f6, f7, f8, g5, g6, g7, g8, h5, h6, h7 or h8). You can't remember where exactly because it's too little time remaining on your clock and you are too stressed!



It's your opponent's move, and, as you don't have clock time remaining, you have to enter a premove queue leading straight to the checkmate. According to chess rules, the checkmate must be made in 50 moves (only your moves count for this rule). The premove queue can store more, up to 500, moves — some of them can be skipped as invalid and therefore don't count.

Input

There is only one test in this problem. We placed a line "`c3 d4`" to the input. You don't need to read it.

Output

Output the sequence of moves. Each move should be a 4-character string, where the first 2 characters denote a starting square, and the last 2 characters denote an ending square. Examples of moves: "`a1b1`", "`c6f3`".

Example

standard input	standard output
`f6 e5`	`f6f7 e5g7 g7g3 g3h4 f6g6 e5g7 g7c7 c7f7`

Note

The sample test describes a position with your king on f6, your queen on e5, and the opponent's king on one of these squares: f8, g8, h8, h7, h6. There is no such test in the testing system.

In the sample test, after black's move, their king still can be only on one of the squares f8, g8, h8, h7, h6. These are all possible endings:

- For black king on f8: 1. Kf6-g6 Kf8-g8 2. Qe5-g7
- For black king on g8: 1. Kf6-g6 Kg8-f8 2. Qe5-g7 Kf8-e8 3. Qg7-c7 Ke8-f8 4. Qc7-f7
- For black king on h8: 1. Kf6-f7 Kh8-h7 2. Qe5-g7
- For black king on h7: 1. Kf6-f7 Kh7-h6 2. Qe5-g7 Kh6-h5 3. Qg7-g3 Kh5-h6 4. Qg3-h4
- For black king on h6: 1. Kf6-f7 Kh6-h7 2. Qe5-g7

Note that in first two cases first four moves in the queue are skipped because they are invalid.