Problem A. ASCII Art

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 mebibytes

The organizers of some open sport programming tournament hired a well-known graphic designer to make a logo styled like ASCII art. The logo consisted of a rectangle sized $h \times w$, made up of the characters "`#`" and "`.`". The designer finished the job, but then the unexpected happened — during the tournament qualification round, the testing server «went down».

In order to make the server's state immediately clear when logging on to the site, the designer suggested «knocking down» the logo when the server is down, by rotating it 90 degrees to the right.

Your task is to use the original given logo and make a «down» version for it.

Input

Two integers h and w are given in the first line of the input file — the dimensions of the logo ($1 \le h, w \le 1000$). The logo itself comes next - h lines of the length w, each consisting of the characters "`.`" and "`#`".

Output

Print w lines, each containing h characters — the logo, rotated 90 degrees to the right.

Example

<table><tr><th>standard input</th><th>standard output</th></tr><tr><td><pre>4 5
.#...
.##..
.###.
.....</pre></td><td><pre>....
.###
.##.
.#..
....</pre></td></tr></table>

Problem B. Battle of Giants

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds (*3 seconds for Java*)
Memory limit:	256 mebibytes

Since the first Battle of Giants at the Urals Championship, this new format has been gaining popularity. The tournament has gradually gained a status similar to the Davis Cup in world tennis, and winning it has become as prestigious for a country as victory in the ACM ICPC World Finals.

...The training camp in Petrozavodsk welcomed T World Championship finalist teams. In order to train for two competitions at once, the teams decided to divide into two groups for each contest —- «home» and «visitors» — with $T/2$ of the teams in each. These groups play a match against each other. The entire training session includes C contests.

The assembly organizers had a lottery drawing and offered the teams a way to divide into groups before each contest. The grouping is considered *successful* if a contest can be found for each pair of teams that they both write when they are in different groups.

For the given lottery results, determine whether the grouping is successful.

Input

Two integers T and C are set in the first line of the input file - the number of finalist teams at the assembly, and the number of contests ($1 \le T \le 4 \cdot 10^4$, $1 \le C \le 50$). It is guaranteed that T is an even number. Teams are assigned numbers in order from 1 to T. In the following C lines, the team groupings are set: the first $T/2$ teams in this round play for the «home team» and the next $T/2$ play for the «visitors».

Output

Output "`YES`" if the lottery can be considered *successful*, and "`NO`" otherwise.

Examples

standard input	standard output

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

`6 2`
`1 2 3 4 5 6`
`1 2 4 3 5 6`
	`NO`

Problem C. Carts

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 mebibytes

For the participants in the World Championship finals in the city of E., a new supermarket was built with fee-based parking for shopping carts. The parking area consists of an area $H \times W$. The projection of each cart on the floor's surface is a square with a side K. A cart is considered parked if it is completely inside the parking area, and its handle (one of the sides of the square) is projected on the border of the area (i.e. a side of the square completely lies on one of the sides of the parking rectangle). In addition, squares corresponding to two different parked carts cannot share any internal points.

What is the maximum number of carts that can be parked?

Input

The first line of the input file contains three integers H, W and K ($1 \le H, W, K \le 5 \cdot 10^8$) dimensions of the parking area and the length of a side of the square that represents a cart, respectively.

Output

Output a single integer — the maximal number of carts that can be parked.

Examples

standard input	standard output
`17 22 5`	`10`
`4 4 2`	`4`
`2 3 4`	`0`

Problem D. Dragon

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 mebibytes

This winter, a lot of snow fell in Kitezhgrad. To solve the problem of clearing the snow from the streets, the researchers from Research Institution of Magic and Wizardry suggested using a fire-breathing dragon.

Kitezhgrad has n intersections connected by $n-1$ two-way streets in such a way that there exists unique path between every two intersections. The length of each street is equal to 1.

For each street, we know the minimal number of times the dragon must go down it in order to completely clear it of snow.

The dragon wants to minimize its efforts, and asked you to make a plan for how it should walk the streets, so that it can completely clear the city of snow while walking a minimum distance. You can choose the starting and finishing intersections.

Input

The first line of the input file contains one integer N — the number of intersections in Kitezhgrad ($2 \leq N \leq 5 \cdot 10^5$). The following $N-1$ lines describe the streets. Each street is defined by three numbers: the indices of the intersections that it connects (a_i and b_i), and the number of times d_i the dragon must walk down this street to clear the snow ($1 \leq a_i, b_i \leq N$, $a_i \neq b_i$, $1 \leq d_i \leq 10^5$).

Output

Print a single integer — the minimum distance the dragon must walk in order to completely clear the snow in Kitezhgrad.

Example

standard input	standard output

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

Problem E. El Clasico

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 mebibytes

From year to year, the Byteland football championship is a face-off between the two top teams: «Integer» and «Bytelona».

Given this, the Byteland Football League decided to perform a draft before each season in order to distribute strong new players between these teams.

The procedure works like this: players line up on the stage in a row from left to right in random order. Also for each player is given his transfer value.

Team managers take turns picking players. For each turn, they can choose either the player in the far right position, the player in the far left position, or both at once. Selected players leave the stage and do not participate in the rest of the selection process.

Given that most new players are immediately leased to other European clubs, the management of each team tries to assure that the total transfer price of all the players selected by his team is as maximal as possible.

This year, «Bytelona» gets to pick first.

Using the data for the transfer value of the players listed in the order in which they are standing on the stage, calculate what the total transfer price of the players selected by «Bytelona» will be, and what the total transfer price of the players selected by «Integer» will be.

Input

The first line of the input file contains one integer N — the number of players participating in the draft ($1 \le N \le 10^6$). The next line lists N integers p_i ($1 \le p_i \le 10^9$) — the transfer price of each player. Players are listed from left to right in the order in which they are arranged on the stage.

Output

Output two integers — the total transfer price of the players selected by «Bytelona» and the total transfer price of the players selected by «Integer».

Examples

standard input	standard output
4 3 1 5 2	8 3

Problem F. Fans

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 mebibytes

In the capital of Byteland, there are several football clubs in addition to «Integer» — «Float», «Double», «Cardinal»... the list could go on. There are a total of N teams in the city that play in various division championships of the country, but all games are played on Sundays, and the fans of any two different teams are rather uptight on game days... And if fans of several different teams happen to meet at once, well...

The capital of Byteland is divided into blocks sized 1×1. For historical reasons, each team's fans get together before a game on the territory of a single section of blocks. A section of blocks consists of a rectangle made up of a whole number of blocks. Sections of blocks that correspond to different teams can overlap.

The city council made arrangements to send an extra patrol car to each block that may have fans of no less than $N - 1$ teams.

Based on the given arrangement of sections of blocks corresponding to different teams, calculate the required number of extra patrol cars.

Input

The first line of the input file specifies an integer N ($2 \leq N \leq 5 \cdot 10^5$) — the number of teams in the capital of Byteland.

Each of the following N lines defines a section of blocks corresponding to a single team, and contains four integers $x_1,\ y_1,\ x_2,\ y_2$ ($0 \leq x_1 < x_2 \leq 10^9$, $0 \leq y_1 < y_2 \leq 10^9$) — the coordinates of the lower-left and upper-right corners of the corresponding section. The borders of the blocks are parallel to the axes of the coordinates.

Output

Output a single integer — the number of extra patrol cars.

Example

standard input	standard output

`3`
`0 0 6 6`
`2 1 3 6`
`1 2 7 4`
	`13`

Problem G. Game with Numbers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 mebibytes

Alice and Bob didn't want to wait for the problem designers and testing system authors to come to an agreement about the best way to implement interactive problems, so instead of a game with boxes from the first division problemset, they decided to play a numbers game.

The game has $2 \cdot n$ cards, each of which has a positive integer written on it. The numbers written on different cards can be the same. Before the game begins, the cards are randomly divided into two decks of n cards in each. Each player puts his deck face down, without disrupting the order they were dealt in. In addition, each player has complete knowledge of the number and order of the cards both in his own deck and in his opponent's.

The players take turns, and Alice's turn is first. On her turn, a player picks up the two cards at the top of her deck and chooses one of them to discard (this card will no longer be used in the game), and one of them to give to her opponent. The other player puts the card he is given at the bottom of his deck. The game is finished when each of the players only has one card left. Let's say that Alice has a card with the number a, and Bob has the number b. This means that Alice gets $a - b$ points, and Bob gets $b - a$ points. The goal of each player is to maximize the amount of points he gets.

Knowing the arrangement of the cards in both decks before the game starts, determine how many points Alice can get if both opponents play optimally well.

Input

The first line of the input file contains a single integer n — the number of cards in each deck ($1 \leq n \leq 10^6$). The second line defines n positive integers, each of which does not exceed 10^6, these are the cards in Alice's s tarting deck, listed sequentially starting from the topmost card. The third line defines Bob's cards in the similar format.

Output

Output a single integer — the number of points that Alice gets when the game is played optimally.

Example

standard input	standard output

4
4 2 6 1
1 7 2 3
	1

Problem H. Hall

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 mebibytes

The organizers of the ACM ICPC Finals in the city of E. found a hall for hosting a press center. The hall is arranged as a polygon bordered by a polyline that is not self-intersecting or self-tangent, with each link parallel to one of the coordinate axes. However, the press representatives demanded to install temporary walls that divide the space into several «normal» rooms (shown on the plan as rectangles), since the journalists are more accustomed to working in these conditions. The installation of each temporary wall requires individual approval from the hall's owner and specific financial expenditures (regardless of the length of the wall), so the organizers would like to install as few walls as possible.

On the plan of the press center, a wall is represented by a segment that is parallel to one of the coordinate axes, that has a beginning and end located on the border of the polygon, for which all internal points are located strictly inside the polygon. In addition, partitions can intersect each other; for example, a 2×2 room can be divided into four 1×1 squares using two partitions (although for the purposes of the task set by the organizers, this particular division is useless).

You must find the minimum number of partitions that can be installed for the organizers to divide the hall into «rectangular» rooms.

Input

The first line of the input file contains a single integer n ($4 \leq n \leq 10^5$) — the number of vertices of the polygon that represents the press center's plan. The following n lines set the traversal of vertices (either clockwise or counter-clockwise). Each vertex is defined by two integers x_i and y_i ($-10^9 \leq x_i, y_i \leq 10^9$). It is guaranteed that the vertices are distinct and that if two sides of the polygon have a shared point, these sides are adjacent, and this point is their shared vertex.

Output

Output a single integer — the minimal number of partitions required in order to divide the press center into «rectangular» rooms.

Example

standard input	standard output

8
-3 -3
3 -3
3 4
1 4
1 0
-1 0
-1 2
-3 2
	2

Problem I. Investigation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 mebibytes

Detective Elijah Bailey is investigating a case of corruption among robots. This case was opened as the result of a statement from one of the guests of a recently colonized planet about how a robot Registrar asked for additional fee in order to accelerate the inspection process. The colonist paid the amount of $X - 1$ space thalers, which the robot did not respond to. However, the robot was perfectly satisfied when the amount of Y space thalers was offered.

Elijah discovered that the problem is that some unknown evildoers have modified the robot's programming so that it only begins to work in normal mode when a fee has been paid that is greater than or equal to some integer amount of P thalers. Now Elijah wants to find out what P is equal to — perhaps the exact value of this amount will lead to the trail of the evildoers.

The main problem is that so far, there is no legal basis for dismantling the robot and checking its software; so Bailey, who doesn't want to waste any time, assigned his agents to go through registration on a flight leaving Earth dressed as rich tourists, and give the robot different amounts when the bribe is requested. If the robot continues to «drag out» the process for an agent, the amount is not enough, but if registration is processed quickly and normally, then the amount is either equal to or in excess of P. Additionally, the money are not returned in any case. Each agent can only give exactly one bribe (in other words, accepting additional payment is not provided for in the program).

With X and Y defined, calculate the minimal sum of money the agents will spend in the worst case scenario in order to find out the exact value of P.

Input

The input file consists of no more than 30 test cases. Each test case is defined on a separate line and consists of two integers X and Y ($20 \le X \le Y \le 1000$) — the minimum and maximum possible values of P based on the information currently available to Bailey. The file ends with the example $X = Y = 0$, which does not need to be processed.

Output

For each test example, output the required optimal amount on a separate line.

Example

standard input	standard output

40 41
20 50
31 100
0 0
	40
	197
	473

Problem J. Journey and Advertising

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 mebibytes

While traveling along the highway that connects the capital of Byteland to a major sea port, a car passes through the zones of different roadside broadcasting towers. The reception area of each tower is a segment with integral coordinates as endpoints. Tower reception areas can overlap.

In addition, the following speed limit is enforced on this highway: all vehicles must travel at a constant speed of one Bytelandian mile per minute.

A well-known Bytelandian advertising agency decided to take advantage of this situation and announced the following promo act: any driver traveling from the capital to the sea can install a device that transmits a one-minute advertisement in all broadcasting frequencies.

During the journey, the device can be activated exactly two times (after each activation, it operates for one minute continuously, transmitting the advertisement). A broadcasting tower receives the advertisement if the vehicle was located within the reception area of this tower during the entire time the advertisement was being transmitted.

The company will award a monetary prize to the driver who can transmit the advertisement to the most broadcasting towers during this journey.

To simplify the tabulation of results, the company's management has asked you to calculate the maximum number of broadcasting towers that a one-minute advertisement can be transmitted to under the given conditions (i.e. transmitting it exactly two times while traveling on this highway).

Input

The first line of the input file contains a single integer n — the number of broadcasting towers ($1 \le n \le 5 \cdot 10^5$). Each of the following n lines contains two integers b_i and e_i — the distance in Bytelandian miles from the capital of Byteland to the starting point and ending point of the reception area for the corresponding broadcasting tower, respectively ($1 \le b_i \le e_i \le 10^9$).

Output

Output a single integer — the maximum number of different broadcasting towers that can receive the advertisement from a vehicle.

Example

standard input	standard output

```
7
1 16
9 13
4 6
7 9
2 2
9 10
3 6
```
	5

Problem K. Kickout

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 mebibytes

The board game «Kickout» goes like this. An n number of identical playing tokens are arranged on n squares, which are numbered by sequential integers from 1 to n. Players take turns. A player's turn consists of moving a token from a square with the number i to a square with the number $2^k \cdot i$ (if such a square exists), where k is a positive integer (for example, a player can choose a token that is on square 3 and move it to one of the squares 6, 12, 24, 48, and so on). If the corresponding square already has a token on it, the two tokens kick each other out (self-destruct) and the square is left empty. The the player who cannot make a move will lose.

Depending on the value of n, in an optimal game, either the first or the second player can win. We'll define the sequence S like this: the j-th element is equal to the j-th value of n in ascending order in which the second player wins. Your task is to use the defined j to calculate S_j.

Input

The first line of the input file contains a single integer j ($1 \le j \le 10^9$) — the index of the required element in the S sequence.

Output

Output a single integer - the j-th element in the S.

Example

standard input	standard output
`3`	`11`