



TASK	STUPOVI	VANDAL	GULIVER
input	standard input		
output	standard output		
time limit	1 second	2 seconds	1 second
memory limit	64 MB		
points	30	50	70
	150		



Due to the big number of cars parked on the sidewalk and problems they cause for the pedestrians, the city government decided to separate some parts of the sidewalk from the road with little pillars.

The sidewalk is divided into parts 1 meter long. The sidewalk is represented as a string where a dot ('.') means that someone can park on that location and an 'X' represents the location of a pillar (which stops the cars from parking there).

All the cars in the town are L meters long. To park on the sidewalk, they need at least L consecutive meters without pillars.

For example, if the string '. . X . . X . .' represents the sidewalk, and the car is 2 meters long, there are 4 different spots where the car can be parked (starting at the first, fourth, fifth and eighth meters).

As the number of available pillars is limited (there are only N available), the government wants to put the pillars so that:

1. The **number of different** available parking spots is the **smallest possible**.
2. If there are multiple ways to meet the first condition, choose one that requires the least number of additional pillars to be placed.

Write a program that, given the number of available pillars N, the length of the cars in meters L and the current layout of the sidewalk, finds any arrangement that satisfies both conditions.

INPUT

The first line of input contains two integers, N and L ($1 \leq N, L \leq 100$), the number of available pillars and the length of the cars in meters.

The second line consists of up to 100 characters that describe the current layout of the sidewalk. Each character represents a meter of the sidewalk. The capital letter 'X' represents a meter that cannot be used for parking because a pillar is already there. The character '.' represents a meter which is currently free and can currently be used for parking.

OUTPUT

Output the layout of the sidewalk after placing additional pillars (if needed), in the same format as in the input.

Note: The solution may not be unique.

EXAMPLE TEST CASES

input

3 4

.

output

. . X . . X . .

input

2 3

X XX XX . . X X

output

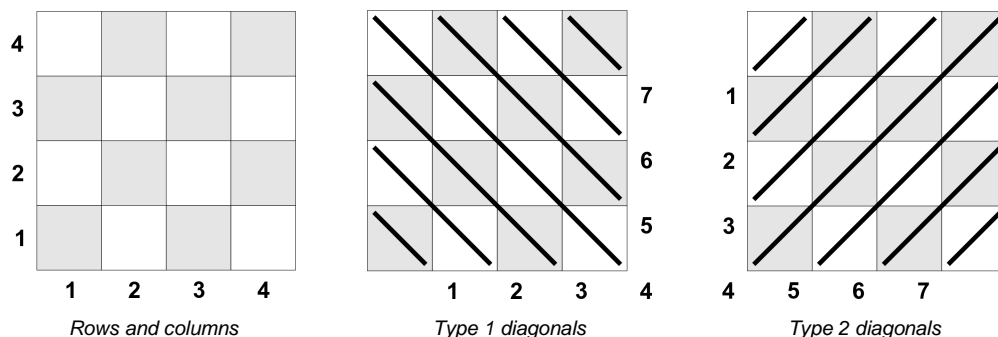
X . . X XX . . X . . XX . . X X

Clarification for first sample: although we have three pillars available, we can place two so that no car can park.

Clarification for second sample: if we choose this final layout, there are four different locations for a car to park. There is no better arrangement.

In an effort to make his village a great tourist attraction, Mirko has drawn a big ($N \times N$) white-gray chessboard on his village hall.

The rows and columns on the chessboard are numbered 1 to N . The bottom-left cell has coordinates (1, 1) and is colored grey. The diagonals of types 1 and 2 are numbered 1 through $2N-1$, like in the following figure (with $N=4$).



The chessboard became a great attraction, but what Mirko didn't expect was that other people would become jealous of his creation. He was unpleasantly surprised when he saw that during the night a hooligan (vandal) ruined all the work he had done on the chessboard with just four brush strokes – he **painted black** all the cells in one row, one column, one diagonal of type 1 and one of type 2!

Before the workday begins, and the tourists rush to the village, he has to repaint all the cells the vandal painted, so that the chessboard looks exactly the same as before the incident.

Write a program that calculates **the total number of cells** that need to be repainted, and how many of them need to be painted white and grey.

INPUT

The first line of input contains the integer N ($1 \leq N \leq 10\,000\,000$), the dimension of the chessboard.

The second line of input contains four integers. Those numbers represent, in order, the row, column, diagonal of type 1, and the diagonal of type 2 which the vandal painted in black.

OUTPUT

On the first line of output, output the total number of cells that need to be repainted.

On the second line, output how many of the cells need to be painted grey and how many white.

Note: If your program outputs the total number of the cells correctly, and it doesn't output the second line of the output correctly or at all, you will get 50% of the points for that case.

EXAMPLE TEST CASES

input

4

1 1 4 4

output

12

6 6

input

7

1 4 11 5

output

19

13 6

input

8

2 6 7 12

output

24

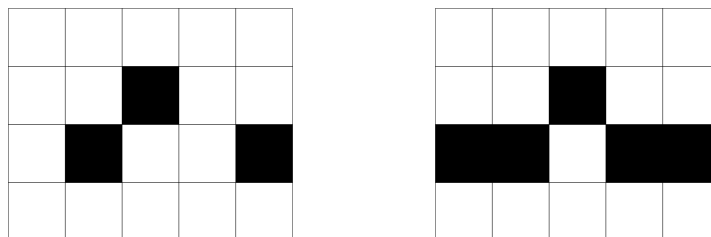
16 8

Once upon a time, on the island inhabited by Lilliput, a great traveler Gulliver was woken up by the cheering of the above mentioned people. After the initial struggle, Lilliputans started telling him about their war with the inhabitants of the neighboring island, Blefuscu.

It is little known that Blefuscu and Lilliput once lived on the same island. After someone had baked an egg, the two nations couldn't agree which side they should start eating the egg on. The war escalated so much that they decided to split the island in two. The South side was taken by Blefuscu, and the North side was taken by Lilliput.

Miraculously, the island's shape was rectangular, composed of $R \times C$ square parcels. Some of the parcels were already underwater, and the citizens decided to flood some other parcels. The island was to be divided into **at least two parts**, one of which would contain the **entire top row** (the North side) and the other part the **entire bottom row** (the South side). Two parcels that aren't underwater belong to the same part of the island if one can get from one to the other by moving up, down, left and right.

For example, if the left image below shows the island before the incident (the black parcels were already underwater), then to separate the island, only two more parcels need to be divided, as in the right image.



The historic problem that people have been talking about for generations is: what was the least number of parcels that had to be flooded to completely separate the two parts of the island? Your task is to write the program which computes that number.

INPUT

The first line of input consists of two numbers, R and C ($3 \leq R, C \leq 1\,000$), the number of rows and columns the island consists of.

Each of the following R lines contains C characters, either '.' or '#'. The '#' sign represent underwater parcels. The input will be such that all of the parcels that aren't underwater are connected and no parcels are flooded in the top and bottom rows.

OUTPUT

Output the number of parcels that need to be flooded to separate the two parts of the island.



EXAMPLE TEST CASES

input

4 5

.....

..#..

.#..#

.....

output

2

input

10 15

.....

#.....#.....

.....

##.....##...

..#.....

.....#.....#...

.....#...##

#.....#...

.....#.....

.....

output

6