2024 Shanghai Freshman Programming Challenge

Problems	**Time Limit**	**Memory Limit**	**Balloon Color**
A. Rush for a place	1 second	512 megabytes	Red
B. nindeECNUOJgule	1 second	256 megabytes	Orchid
C. sha7dow loves rand function	1 second	128 megabytes	Spring Green
D. 2D Pinball (Easy)	4 seconds	1024 megabytes	Yellow
E. 2D Pinball (Hard)	4 seconds	1024 megabytes	Orange
F. sha7dow loves data structure	1 second	128 megabytes	Pale Green
G. Roll the dice	2 seconds	512 megabytes	Dark Blue
H. 164ovo dislikes triangles	1 second	256 megabytes	Pink
I. sha7dow loves sqrt technology	1 second	128 megabytes	Sky Blue
J. Count S	1 second	512 megabytes	Sandy Brown

There are three problems that we think are the easiest to solve. Only the first to solve these three problems will be given a balloon. The first to solve the remaining problems will be given two identical balloons.

A. Rush for a place

The 2024 ICPC China Shaanxi National Invitational Programming Contest will be held in May. `SpadeZ` is very eager to participate in this contest and get her travel to Xi'an reimbursed so she can visit her friend. The registration channel will open on March 18th, but the number of participants is limited. To avoid missing the registration, she plans to keep checking the *QQ* group every fixed interval. However, her intense studies have left her very tired, so she needs to take a nap at some point, which means she won't be able to check the group during that time. After waking up, she will immediately check the group, and then continue checking with the previous interval. `SpadeZ` wants to know how many different nap schedules will allow her to get a place.

Specifically, assume the checking interval is t:

- If she doesn't sleep, `SpadeZ` will check the group at $t,\ 2t,\ 3t, \cdots$;
- If `SpadeZ` sleeps from time x for y units of time, she will check the group as usual in $[1, x-1]$, not check in $[x, x+y-1]$, and then check again at $x+y,\ x+y+t,\ x+y+2t, \cdots$.

Given the checking interval t and a positive integer n, the registration channel will open at time p (1 ≤ p ≤ n). `SpadeZ` can only register if she check the group at time p. `SpadeZ` will choose a time $x\ (1 \leqslant x \leqslant n)$ to sleep for $y\ (1 \leqslant y \leqslant n)$ units of time (i.e. there are $n \times n$ possible combinations). Please find the number of schedules that will allow `SpadeZ` to get a place for the contest.

Input

Each test case consists of multiple test cases.

The first line contains an integer $t\ (1 \leqslant t \leqslant 2 \times 10^5)$ -- the number of test cases. Then follows the description of the test cases.

The first line of each test case contains 3 integers $n,\ t,\ p\ (1 \leqslant n \leqslant 1 \times 10^9,\ 1 \leqslant t,\ p \leqslant n)$ -- the time range, the time interval of checking the group, and the time when registration opens, respectively.

Output

For each test case, output an integer representing the number of schedules that will allow `SpadeZ` to get a place for the contest.

Samples

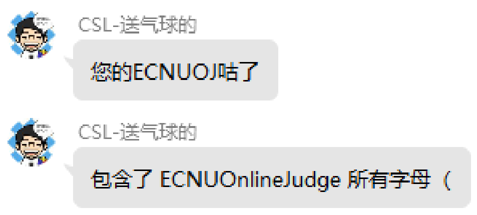
Input #1

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

Output #1

```
0
4
```

B. nindeECNUOJgule


CSL-送气球的

您的ECNUOJ咕了

CSL-送气球的

包含了 ECNUOnlineJudge 所有字母（

Someone noticed that the name of the official QQ Group of ECNU Online Judge `nindeECNUOJgule` is a shuffled version of `ECNUOnlineJudge` ! `Cuber QQ` and his friends want to make more string pairs with this feature, they decided to generate some random string pairs, then delete some characters in these strings to construct string pairs that satisfy the requirement above. `Cuber QQ` already generated some string pairs for you, please help them to complete the second half of the problem.

Given two strings, please determine whether you can make two strings identical by swapping characters in the **same** string (In other words, select two characters in the same string and swap their places) finite number of times. If not, find the minimum number of characters you can delete in both strings to make the string pair satisfy the conditions above.

Input

In the first line there is one positive integer len $(1 \le len \le 2 \times 10^5)$, denoting the length of strings in the string pair below.

In the next two lines, there is one string of length len in each line, denoting the generated string pair.

It is guaranteed that these strings only contain visible characters excluding white-space.

Output

In the first line you should output one non-negative number num , indicating the **minimum** number of characters you can delete in **EACH** string to make the string pair satisfy the condition above.

If $num > 0$, output two extra lines, each line contains one string of length num , indicating the characters you select to delete in these two strings. After deleting these number of characters in the corresponding string above, the string pair should satisfy the condition.

You can output the characters to be deleted in any order, and the string after deletion can become empty.

Samples

Input #1

```
15
ECNUOnlineJudge
nindeECNUOJgule
```

Output #1

```
0
```

Input #2

```
15
L3]k*sSdh_1ACPX
3BXh_!1]*PsNkAd
```

Output #2

```
3
CSL
NB!
```

Input #3

```
4
BBBB
bbbb
```

Output #3

```
4
BBBB
bbbb
```

C. sha7dow loves rand function

`sha7dow` loves rand function, and there are always people wanting his help in generating arcane random numbers with various periods. `sha7dow` is annoyed with having to avoid everyone's disliked numbers and not being able to simply use `mt19937(steady_clock::now().time_since_epoch().count())`, so all he wants is a perfect random number generator that fulfills all the requirements of everyone. You need to use a linear congruence generator to help him generate an infinite random number sequence x_i such that the given n numbers t_i are all periods of the sequence, and the m numbers s_i are not periods of the sequence. In detail, with $x_0 = 0$ and $x_i = (ax_{i-1} + b) \bmod c$, you need to help sha7dow get a set of **feasible** values for a, b, c, or regrettably tell `sha7dow` that this is not feasible.

Input

The first line of the input contains two integers n and m.

The second line of the input contains n integers t_i — all of which are periods of the random number sequence.

The third line of the input contains m integers s_i — none of which are periods of the random number sequence.

Output

Print three integers a, b, c or `0 0 0` if not feasible.

Samples

Input #1

```
2 1
20 40
66
```

Output #1

```
22 33 10
```

Note

$1 \leqslant n, m \leqslant 10^5$
$1 \leqslant t_i, s_i \leqslant 10^9$
$0 \leqslant a, b, c \leqslant 10^9$

$r = a \bmod b$ if and only if $a = qb + r, r \in [0, b)$

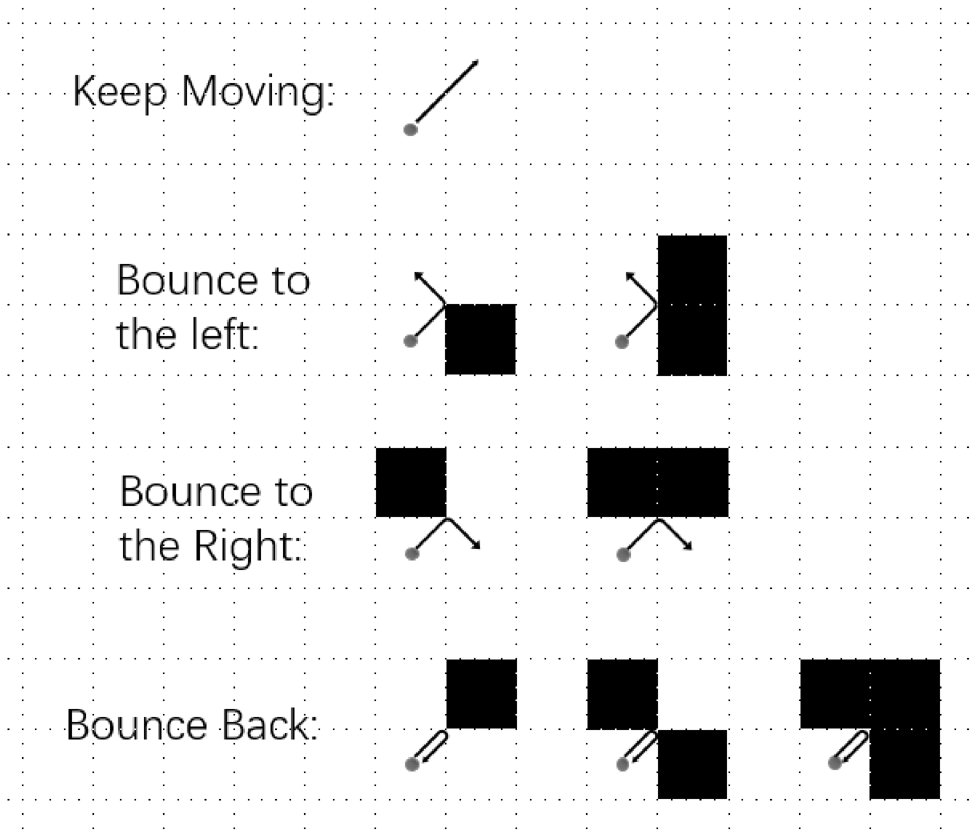
D. 2D Pinball (Easy)

PLEASE NOTE: There is a harder version of this problem. The only difference between the two versions is the data range. It is guaranteed that if you passed the hard version you can use the same code to pass the easy version. In this version, $T = 1$.

`SpadeZ` have a rectangular table with a grid of n rows and m columns of squares. Each square can be empty or have an obstacle.

We use a string of two characters to represent the movement direction of the ball: `LU` is Left Up, `LD` is Left Down, `RU` is Right Up, `RD` is Right Down. the movement direction can only be one of these four patterns.

The ball will start from the center of a square, move at a speed of $\sqrt{2}\times$ length of the square side per second as the frictional force is ignored. If the ball is not blocked in the front, left and right of its movement direction, it will continue moving in the same direction, otherwise it will collide with the obstacles after 0.5 seconds and change its movement direction shown in the picture below. Because the size of the ball is much smaller than the square grid, the small shifts in its movement is ignored so **we consider the ball is always in the center of one square in integer seconds**.



After `SpadeZ` arranged the pinball table, `sha7dow` immediately came to play it. He threw a ball into the table in second 0, and can't wait to find the position and direction of the ball after moving t seconds. Given the layout of the table, the position and movement direction of `sha7dow`'s pinball, please calculate its position and movement direction after moving t seconds.

Input

In the first line there are two space-separated positive integers n, m $(1 \le n \le 500,\ 1 \le m \le 500)$, denoting the rows and columns of the grid.

In the next n lines, each line consists of one string of length m ,indicating the layout of the grid. The character in the left-up corner is in the first row and first column. In the strings, `0` represents an obstacle, while `.` represents there is an empty space. The positions outside the grid are all obstacles.

In the next line, there is one positive integer T $(T = 1)$,indicating the number of classmates to play the pinball game made by `SpadeZ` .

In the next T lines, the i-th line $(1 \le i \le T)$ consists of three space-separated positive integers t_i, r_i, c_i $(1 \le t_i \le 10^9, 1 \le r_i \le n, 1 \le c_i \le m)$ and a string dir_i , indicating the starting movement status of the ball of the i-th classmate. t_i is the time the classmate waits (in seconds), r_i, c_i are the row position and the column position of the ball, dir_i is the starting movement direction.

It is guaranteed that every ball starts in an empty space.

Output

Output T lines, the i-th line $(1 \le i \le T)$ contains two integers and one string separated by one space, indicating the position in rows and columns and the movement direction of the i-th classmate's ball after t_i seconds.

Samples

Input #1

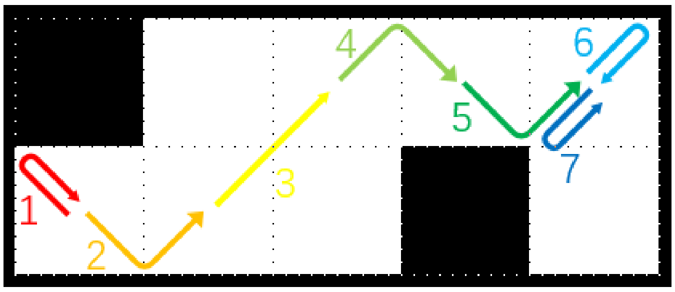
```
2 5
0....
...0.
1
7 2 1 LU
```

Output #1

```
1 5 RU
```

Notes

In the first sample, the movement of the ball is demonstrated in the picture below, in which one second of movement is represented by one labeled arrow.



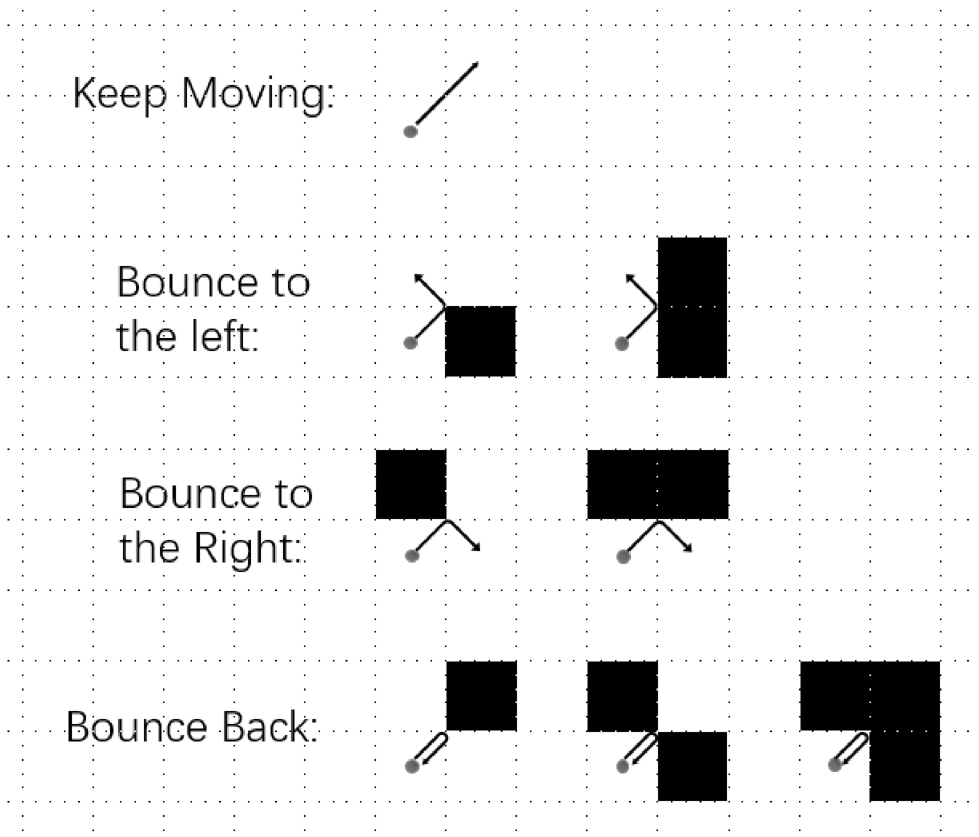
E. 2D Pinball (Hard)

PLEASE NOTE: There is an easier version of this problem. The only difference between the two versions is the data range. It is guaranteed that if you passed the hard version you can use the same code to pass the easy version. In this version, $1 \le T \le 2 \times 10^5$.

`SpadeZ` have a rectangular table with a grid of n rows and m columns of squares. Each square can be empty or have an obstacle.

We use a string of two characters to represent the movement direction of the ball: `LU` is Left Up, `LD` is Left Down, `RU` is Right Up, `RD` is Right Down. the movement direction can only be one of these four patterns.

The ball will start from the center of a square, move at a speed of $\sqrt{2}\times$ length of the square side per second as the frictional force is ignored. If the ball is not blocked in the front, left and right of its movement direction, it will continue moving in the same direction, otherwise it will collide with the obstacles after 0.5 seconds and change its movement direction shown in the picture below. Because the size of the ball is much smaller than the square grid, the small shifts in its movement is ignored so **we consider the ball is always in the center of one square in integer seconds**.



After `SpadeZ` arranged the pinball table, `sha7dow` immediately came to play it with his classmates! A classmate can throw a ball into the table only after the previous ball thrown was picked up by the classmate who threw it. Each classmate will throw a ball into the table in second 0, and they can't wait to find the position and direction of the ball after moving t seconds, especially the classmates still waiting to throw their ball. Given the layout of the table, the position and movement direction of `sha7dow`'s classmates' pinballs, please calculate their positions and movement directions after moving t_i seconds.

Input

In the first line there are two space-separated positive integers n, m $(1 \le n \le 500, 1 \le m \le 500)$, denoting the rows and columns of the grid.

In the next n lines, each line consists of one string of length m ,indicating the layout of the grid. The character in the left-up corner is in the first row and first column. In the strings, `0` represents an obstacle, while `.` represents there is an empty space. The positions outside the grid are all obstacles.

In the next line, there is one positive integer T $(T = 1)$,indicating the number of classmates to play the pinball game made by `SpadeZ` .

In the next T lines, the i-th line $(1 \le i \le T)$ consists of three space-separated positive integers t_i, r_i, c_i $(1 \le t_i \le 10^9, 1 \le r_i \le n, 1 \le c_i \le m)$ and a string dir_i , indicating the starting movement status of the ball of the i-th classmate. t_i is the time the classmate waits (in seconds), r_i, c_i are the row position and the column position of the ball, dir_i is the starting movement direction.

It is guaranteed that every ball starts in an empty space.

Output

Output T lines, the i-th line $(1 \le i \le T)$ contains two integers and one string separated by one space, indicating the position in rows and columns and the movement direction of the i-th classmate's ball after t_i seconds.

Samples

Input #1

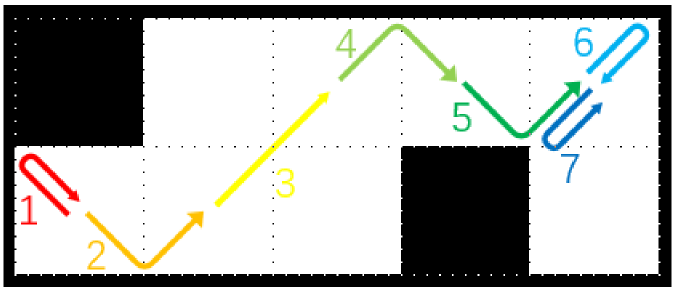
```
2 5
0....
...0.
1
7 2 1 LU
```

Output #1

```
1 5 RU
```

Notes

In the first sample, the movement of the ball is demonstrated in the picture below, in which one second of movement is represented by one labeled arrow.



F. sha7dow loves data structure

`sha7dow` loves data structure, especially all sorts of peculiar trees. `sha7dow` recently obtained a perfect sequential storage binary heap, which is a full binary tree with $n+1$ layers. He hopes that you can help him generate another tree T with this perfect binary heap H.

- `sha7dow` starts by edging every two neighboring leaves x and $x+1$ in H ($2^n \leqslant x < 2^{n+1}-1$).
- You need to determine the number of nodes m in T and choose a node in T to edge for each node in H.
- For each pair of nodes u, v in T, edge u, v if there exists a pair of nodes x, y in H and a chain of the form $u - x - y - v$.
- `sha7dow` ends by deleting all edges connected to nodes in H except those in the original full binary tree.

`sha7dow` prefers not to overcomplicate the graph in this generation process, so you need to guarantee that there should be no more than k edges connected to each of the nodes in H and T throughout the process.

Input

The only line of the input contains two integers n and k.

Output

In the first line, print one integer — the number of nodes m in T.

In the second line, print n integer — the node in T to edge for each node in H.

Samples

Input #1

```
2 2000
```

Output #1

```
2
2 1 1 2 1 1 2
```

Note

$1 \leqslant n \leqslant 20 \leqslant k \leqslant 10^6$
$1 \leqslant t_i \leqslant m < 2^{n+1}$

A binary heap is a tree-based data structure, in which the tree is a complete binary tree. Heaps are typically constructed in-place in the same array where the elements are stored, with their structure being implicit in the access pattern of the operations, which is called sequential storage, i.e., for all non-leaf nodes x, it is edged with $2x$ and $2x+1$, respectively (notably, in the case of a full binary tree with $n+1$ layers, $1 \leqslant x < 2^n$).

G. Roll the dice

There is an $n \times m$ matrix, where the rows and columns are numbered with integers. The rows are numbered from top to bottom from 1 to n, and the columns are numbered from left to right from 1 to m. The cell (i, j) is the cell at the intersection of the i-th row and the j-th column $(1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant m)$. On this matrix, there is a dice[1]. The dice is initially positioned at cell (sx, sy), with the number `1` facing upwards and `5` facing left. There are four possible operations:

1. Roll the dice one cell to the right and rotate it $90°$ to the right.
   If the dice was initially at (i, j) with `1` facing up and `5` facing left, after this operation it will be at $(i, j+1)$ with `5` facing up and `6` facing left.
2. Roll the dice one cell to the left and rotate it $90°$ to the left.
   If the dice was initially at (i, j) with `1` facing up and `5` facing left, after this operation it will be at $(i, j-1)$ with `2` facing up and `1` facing left.
3. Roll the dice one cell forward and rotate it $90°$ forward.
   If the dice was initially at (i, j) with `1` facing up and `5` facing left, after this operation it will be at $(i+1, j)$ with `4` facing up and `5` facing left.
4. Roll the dice one cell backward and rotate it $90°$ backward.
   If the dice was initially at (i, j) with `1` facing up and `5` facing left, after this operation it will be at $(i-1, j)$ with `3` facing up and `5` facing left.

Note that the position of the dice after an operation must still be within the boundaries of the matrix, i.e., $1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant m$ must always hold. The goal is to determine the minimum number of operations required to move the dice to cell (tx, ty) with `1` still facing upwards. If it is impossible to reach (tx, ty) while satisfying this condition, the output should be -1.

Input

Each test consists of several test cases. The first line contains a single integer t $(1 \leqslant t \leqslant 2 \times 10^5)$ — the number of test cases. Then follow the descriptions of the test cases.

The first and only line of each test case contains six integers $n,\ m,\ sx,\ sy,\ tx,\ ty$ $(1 \leqslant n, m \leqslant 10^{18},\ 1 \leqslant sx, tx \leqslant n,\ 1 \leqslant sy, ty \leqslant m)$ as described in the problem statement.

Output

For each test case, output an integer. If it is impossible to reach (tx, ty) while satisfying the conditions, output -1; otherwise, output the minimum number of operations.

Samples

Input #1

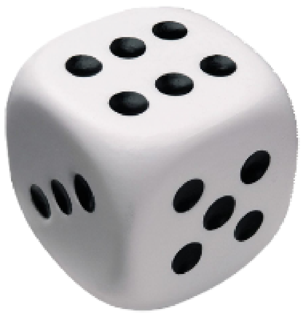
```
4
1 2 1 1 1 2
5 5 1 1 5 5
5 5 3 3 5 5
3 4 1 1 3 4
```

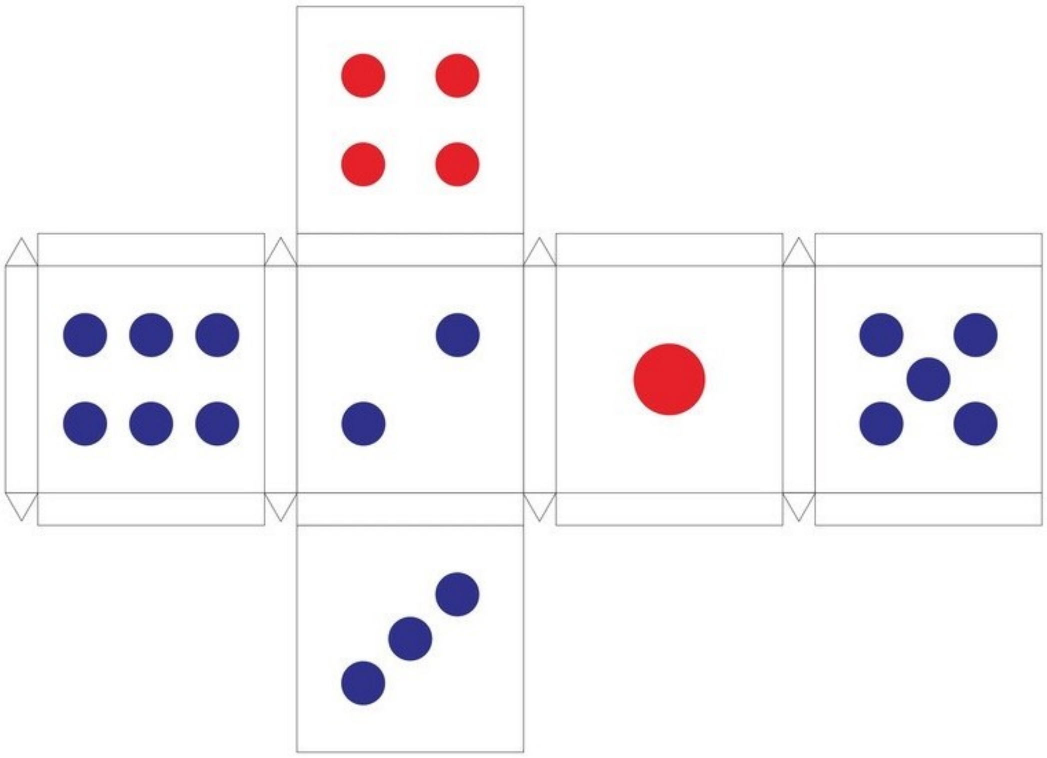
Output #1

```
-1
8
4
5
```

Note

The dice is a three-dimensional cube with six faces marked `1`, `2`, `3`, `4`, `5`, `6`, where `1` and `6` are opposite, `2` and `5` are opposite, and `3` and `4` are opposite. The specific spatial form is as shown in the figures:





H. 164ovo dislikes triangles

After `sha7dow` found that `164ovo` didn't want to see any triangle anymore since 2022 ICPC Nanjing Regional Contest, he decided to help `164ovo` to solve all triangle-related problems he encounter. Look, here comes another problem of triangles:

There is a big regular triangle in a 2-dimensional plane. You can draw any amount of segments in this triangle as you wish in order to separate this large regular triangle into several smaller regular triangles. These small triangles can be different in size, but any pair of the smaller triangles may not intersect except sharing one common point or edge. The sum of these smaller regular triangles must equal to the large triangle in the beginning.

Because `sha7dow` thought this problem is too naïve, he gave this problem back to `164ovo` . Unfortunately `164ovo` didn't want to see any triangle anymore, he gave this problem to you and asked that whether this large regular triangle can be separated into a fixed amount of smaller regular triangles.

Input

There is one line consists of one positive integer n $(2 \le n \le 10^{2\times 10^5})$ in the input, denoting the count of smaller regular triangles you should separate the large regular triangle into.

Output

Output one line consists of one string `Yes` or `No` , indicating whether there is a valid method to separate this large regular triangle into n smaller regular triangles.

Samples

Input #1

```
4
```

Output #1

```
Yes
```

I. sha7dow loves sqrt technology

`sha7dow` loves sqrt technology. He found the template problem for interval inversion when learning *Sweepline Mo / Offline Again*。

You are given a sequence a of length n and m queries, each querying the number of inversion of an interval.
$1 \leqslant n,m \leqslant 10^5$, $0 \leqslant a_i \leqslant 10^9$。

`sha7dow` studied for an entire day to pass this template problem, so he wants to test you too. Without data, however, he intends to generate the intervals for all queries in a simple way.

- Generate a queue with only one interval $[1, n]$.
- Keep taking out the first interval of the queue until the queue is empty.
- If this interval is not of length 1, it will be used as an interval for a query, and somehow split into several intervals and put them all back into the queue.

`sha7dow` finds that this generation seems too simple. So for each interval S_i, if the intervals formed by splitting it with the left endpoints ordered from smallest to largest are referred to as $T_{i,j}$, you also need to calculate the sum of the number of inversion of the sequences formed by splicing T_{i,j_1} and T_{i,j_2} which satisfy $j_1 < j_2$ and that the parities of j_1 and j_2 are different.

Input

The first line of the input contains two integers n and m.

The second line of the input contains n integers — given sequence a_i.

Each of the next m lines contains two integers l_i, r_i — queried interval $S_i = [l_i, r_i]$. It is guaranteed that all queried intervals are pairwise distinct.

Output

For each query print two integer — the number of inversion of the queried interval S_i, and the sum of the number of inversion of the sequences formed by splicing T_{i,j_1} and T_{i,j_2}.

Samples

Input #1

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

Output #1

```
1 1
3 3
1 1
10 10
```

Note

$1 \leqslant n, m \leqslant 10^5$
$0 \leqslant a_i \leqslant 10^9$
$1 \leqslant l_i \leqslant r_i \leqslant n$

Inversion in sequence a is a pair of elements that are out of their natural order, indicated by an ordered pair (i, j) if $i < j$ and $a_i > a_j$.

J. Count S

Given an $n \times n$ grid graph, where some points are black and the rest are white, please count the number of S-shaped subgraphs.

An S-shaped subgraph of size k is defined as follows:

1. It consists of 5 line segments of length k $(k \geqslant 1)$
2. The 5 line segments are connected end-to-end and cover only black points (Note that it is all grid points on the line segment that are black, not just the endpoints).
3. The shape is like an "S", with the endpoint coordinates as follows:

$(a, b), (a, b-k)$
$(a, b-k), (a+k, b-k)$
$(a+k, b-k), (a+k, b)$
$(a+k, b), (a+2k, b)$
$(a+2k, b), (a+2k, b-k)$

Note that the grid graph is numbered from top to bottom as 1 to n, and from left to right as 1 to n.

Input

The first line of the input contains an integer n $(1 \leqslant n \leqslant 50)$, representing the size of the grid graph.

The following n lines, each contains a string s_i $(1 \leqslant i \leqslant n)$ of length n.

Let $s_{i,j}$ denote the j^{th} $(1 \leqslant j \leqslant n)$ character of s_i, where $(s_{i,j} \in \{$ `o`, `x` $\})$. If $s_{i,j} =$ \$\o\, *itindicatesthatthepointinthe*\$$i^{th}$ row from top to bottom and the j^{th} column from left to right of the grid graph is white, while \$s_{i, j} = \$ `x` indicates that the point is black.

Output

Output a single integer, representing the number of S-shaped subgraphs in the grid graph.

Samples

Input #1

```
3
xxo
xxo
xxo
```

Output #1

```
1
```

Input #2

```
4
ooxx
ooxx
ooxx
ooxx
```

Output #2

```
2
```

Input #3

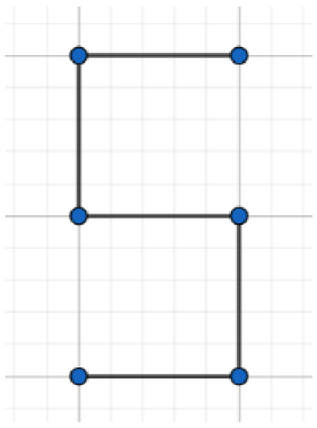
```
3
xxx
xox
xxx
```

Output #3

```
0
```

Note

For example, the figure below contains an S-shaped subgraph of size 5.



Note that the figure below does not contain an S-shaped subgraph.