## A. Aztec Diamond

time limit per test 1 second memory limit per test 128 megabytes
Jaemin found some ancient diamond-shaped symbols composed of $1 \times 2$ bricks.


Noticing that some bricks are horizontal and some are vertical, Jaemin felt uncomfortable and decided to turn all bricks to vertical. Jaemin cannot lift a brick with one hand because his hands are small, but he can lift two bricks with two hands. Therefore, the only thing he can do is to choose any $2 \times 2$ square composed of two bricks and rotate it 90 degrees.


A programming contest is starting soon, so he can't spend too much time here. Let's help him!

## Input

On the first line, the size of symbol N is given( $1<=\mathrm{N}<=100$ ). On the next 2 N lines, the string with length 2 N representing row of symbol is given. "." is a blank, "L" and "R" are the left and right squares of the horizontal brick, and "U" and "D" are the upper and lower squares of the vertical brick.

## Output

On the first line, print the number of rotations D needed to make all the bricks vertically. This value should be same or less than $N \wedge 4$. The next $D$ line shows the row number and column number in the upper left corner of the $2 \times 2$ square to rotate. The top row and leftmost column are numbered 1. The existence of an answer is guaranteed for all inputs satisfying the input condition.

```
Example
input
3
    . .UU. .
    .UDDU.
UDLRDU
DUULRD
.DOLR.
. .LR.
output
4
4
4
3
5
```


## B. Break Oven, Run Cookie!

time limit per test 1 second memory limit per test 256 megabytes
The witch locked the brave cookies in the oven! The floor of the oven is a grid, and each second, each cookie can move to one of the 4 adjacent cells or stay there. The witch will find it suspicious when the cookies are close together, so there can be at most one cookie per cell.

$\square$

$\nu$


2

You are a cookie lover and want to rescue the cookies. You found that some cells are weak, so you plan to direct the cookies there. Fortunately, there are exactly the same number of weak cells as the cookies. Once a cookie step on the cell, it breaks and the cookie escapes through the hole. However, once a cookie escapes through a hole, the witch spawns an obstacle on that cell, preventing other cookies from stepping it.

To avoid the witch's suspicion, each cookie can move at most 1 cell per second. Find the least amount of time needed for all cookies to escape.


The above image shows a solution for the sample input 1. The gray cells are weak cells, and the black cells are impassable cells because of the obstacles.

## Input

The first line of the input consists of three integers $\mathrm{H}, \mathrm{W}$, and N , where H and W are the height and the width of the grid, and N is the number of the cookies (and weak cells). $1<=\mathrm{H}, \mathrm{W}, \mathrm{H}^{*} \mathrm{~W}<=100$, and $1<=\mathrm{N}<=\mathrm{H}^{*} \mathrm{~W} / 2$. Each of the next N lines contains two integers $r$ and $c$, the position of each cookie. Each of the next N lines contains two integers $r$ and $c$, the position of weak cells. $1<=r<=H, 1<=c<=$ W , and all positions are distinct.

## Output

Print the least amount of time needed for all cookies to escape, or -1 if it's not possible that all cookies escape.

## Examples

## input

444
12
13
21
31
24
34
42
43
output
4
input
142
11
12
13
14
output
-1

## C. Coke Challenge

time limit per test 1 second memory limit per test 512 megabytes
Mr. Jeong really loves coke. He loves so much that he drinks coke everyday without exception. One day, he decided to open a coke contest in Daejeon. To the winner, a box of cokes will be given!
$N$ people participate in the contest. Each participant is given $K \mathrm{~mL}$ of coke, and the one who finishes his/her coke earliest wins the contest. But it is painful to drink the whole coke in one time, so each person repeats drinking and taking a rest. More specifically, as soon as the contest starts, the $i$ th participant starts to drink for $t_{i}$ seconds, then takes a rest for $s_{i}$ seconds, and repeats this process until no more coke remains. Moreover, everyone drinks $A \mathrm{~mL}$ per second. The contest is over if one of the participants finished his/her coke.

Given the infomation of $N$ participants, determine after how many seconds the contest is finished, since the contest begins.

## Input

The input starts with the first line containing three integers $N(2 \leq N \leq 1000)$, $K(1 \leq K \leq 10000)$, and $A(1 \leq A \leq 100)$. The $i$ th of the next $N$ lines contains two integers $t_{i}\left(1 \leq t_{i} \leq 100\right)$ and $s_{i}\left(1 \leq s_{i} \leq 100\right)$, the information of $i$ th participant.

## Output

Write a single integer, the answer to the question.

## Examples

## input

21001
105
510

## output

145

## input

41002
3030
492

5050
2010
output
50

## D. Dev, Please Add This!

time limit per test 1 second memory limit per test 256 megabytes
Jaemin developed a puzzle game app.
A ball is placed on a cell in a grid. You can roll the ball upwards, downwards, to the left, or to the right. The ball rolls in that direction until it hits a wall or the boundary. Some cells have stars on them, and the player obtains the star when the ball stops there or passes through it. The objective of the game is to obtain all stars in the grid.

This app has a level editor where players can make and share their own levels. One day, someone suggested: "Please add a feature to check if my level is possible to solve!" Easier said than done, right?

## Input

The first line of the input consists of two integers, $H$ (height) and $W$ (width). $(1 \leq H, W \leq 50)$ Each of the next $H$ lines contains a string of legnth $W$ which describes each row of the grid. "\#" means a wall, "." means a blank space, "O" means a ball, and "*" means a star. There is exactly one ball and at least one star on the grid.

## Output

Print "YES" if it is possible to obtain all stars, otherwise "NO".

## Examples

## input

37
\#. .0.. \#
\#.\#\#\#.\#
*..\#...
output
NO

## input

66
*..*\#\#
. 0...
*..*\#.
\#\#\#\#*.
...... \#
output
YES

## E. Expectation of Games

time limit per test 1.5 seconds memory limit per test 128 megabytes
Snakes and Ladders is a famous dice game. Initially, a token is placed outside the board. Each turn, you roll a die with $M$ sides. There is $1 / M$ probability that each integer from 1 to $M$ appears. Then you read the number $x$ on the die, and move forward your token $x$ times. (If the token is outside the board, place it on the $x$-th cell.) If the resulting cell has a snake or a ladder, you move to the cell it connects to. The game is over if the token reaches the $N$-th cell or beyond.

It's theoretically possible to get in the infinite loop, but if a game is correctly designed, it must be possible to reach the end regardless of where the token is placed at. If not, let's call it a "badly designed game."

Find the expected number of dice rolls to finish the game.

## Input

The first line of the input consists of three integers: $N, M$, and $S$, where $S$ is the number of snakes and ladders. $(1 \leq N \leq 100,1 \leq M \leq 20,0 \leq S \leq N / 2)$ Each of the next $S$ lines contains two integers: the index of the starting cell and the ending cell. (If the index of the ending cell is smaller than that of the starting cell, it is a snake; otherwise, it is a ladder.) The index of the starting cell is at least 1 and at most $N$ -
1 . The index of the ending cell is at least 1 and at most $N$. There are no duplicates among all starting and ending cells.

## Output

Print the expected number of dice rolls to finish the game. Any answer within the absolute or relative error of at most $10^{-4}$ will be accepted. If the game is badly designed, print - 1 .

## Examples

## input

421
24

## output

1.75

## input

```
6 2
4
5
16
output
```

-1

## F. Faster Sorting

time limit per test 1 second memory limit per test 128 megabytes
From this year, you can use Python in the ACM-ICPC World Final. To celebrate this, we introduce a fun Python fact.

Python uses a sorting algorithm called Timsort. In short, the algorithm splits the list into mostly sorted subarrays, sorts each subarray using insertion sort, and merges the sorted subarrays. If the list is already mostly sorted, this algorithm runs very quickly.

In this problem, let's focus on the splitting part:

1. Take the longest non-decreasing ( $a_{0} \leq a_{1} \leq a_{2} \leq \ldots$ ) or increasing ( $a_{0}>a_{1}>a_{2}>\ldots$ ) subarray that starts from the current index.
2. If the length of the subarray is less than MINRUN, take more elements until the length equals MINRUN or all elements are taken. Let's call these additional elements "bad elements".
MINRUN=3

| 2 | 4 | 4 | 3 | -1 | -2 | -2 | 5 | 6 | 5 | 4 | 3 | 2 | 3 | 4 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

MINRUN=4
BAD BAD BAD

MINRUN=5 BAD BAD BAD BAD BAD

| 2 | 4 | 4 | 3 | -1 | -2 | -2 | 5 | 6 | 5 | 4 | 3 | 2 | 3 | 4 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |

If MINRUN is too small, there are too many subarrays to merge; if MINRUN is too large, it takes too much time to apply insertion sort. For this reason, it is important to set an appropriate value of MINRUN. Also, N/MINRUN should be close to a power of 2 to ensure a balanced merging, but we will not consider this in this problem. For each $M I N R U N$ value, find the number of subarrays and bad elements.

## Input

The first line consists of a single integer $N(5 \leq N \leq 100,000)$, the length of the array. The second line consists of $N$ integers, the elements of the array. The absolute value of each element is at most $10^{9}$. The third line consists of a single integer $Q(1 \leq Q \leq 100,000)$, the number of queries. Each of the next $Q$ lines contains a single integer MINRUN $(2 \leq M I N R U N \leq N)$.

## Output

For each query, print the number of subarrays and bad elements in one line.

## Example

## input

15
$2443-1-2-256543234$
3
3
4
5
output
50
43
35

## G. God Game

time limit per test 1 second memory limit per test 512 megabytes
Taeyoung is a game addict. He plays "Legend of League" everyday and does not study at all. His friend Joon though that Taeyoung was a pathetic person, so he decided to make "God Game" to take Taeyoung out of "Legend of League".

The game Joon made is the following. The game takes place in a rectangle of size $N \times M$ placed in the first quadrant of the $(x, y)$-coordinate plane, where one of the corner is the origin of the plane. The goal of the game is to move a ball from the start position (startX, start $Y$ ) to the end position (end $X$, end $Y$ ). These are all integer coordinates. Now for each integer time $t$, Taeyoung does one of the following.

1. Leave the ball in place for 1 second.
2. Let $(x, y)$ be the current position of the ball. Move the ball to the new position for 1 second at a speed of 1 unit per second. The new position should be one of $(x+1, y),(x, y+1),(x-1, y)$, and $(x, y-1)$, provided that it is allowed to place the ball there.

At every moment, the coordinate $(x, y)$ should satisfy $0 \leq x \leq N$ and $0 \leq y \leq M$. Even if this condition is satisfied, some positions are forbidden to place the ball. Note that at least one of $x$-coordinate and $y$-coordinate is an integer at each time $t$.

However, even with these constraints, the game is still too easy. So Joon made $K$ moving obstacle on several levels. The $i$ th obstacle starts from $\left(x_{i}, y_{i}\right)$ at time $t=0$, and follows the borderline of a square with one side of $a_{i}\left(1 \leq a_{i} \leq 5\right)$, in a clockwise direction at a speed of 1 unit per second. $\left(x_{i}, y_{i}\right)$ is the bottom left corner of the square. For example, if $a_{i}=2$, then the obstacle moves along the following in a cycle of 8 seconds.

$$
\begin{gathered}
\left(x_{i}, y_{i}\right) \rightarrow\left(x_{i}, y_{i}+1\right) \rightarrow\left(x_{i}, y_{i}+2\right) \rightarrow\left(x_{i}+1, y_{i}+2\right) \rightarrow\left(x_{i}+2, y_{i}+2\right) \rightarrow\left(x_{i}+2, y_{i}+1\right) \rightarrow( \\
\left.x_{i}+2, y_{i}\right) \rightarrow\left(x_{i}+1, y_{i}\right) \rightarrow\left(x_{i}, y_{i}\right) \rightarrow \ldots
\end{gathered}
$$

The paths of obstacles do not leave the $N \times M$ rectangle, but it may move to a place that the ball is not allowed to move, and several obstacles may share a common path. If the ball and an obstacle is in the same position at some time $t$, the game is over and we are lost. Here, $t$ may not be an integer, and so is the coordinate where the ball and the obstacle met.

After Joon finished developing the game, he called the game "God Game" and introduced it to Taeyoung. But the game was way too difficult, and he couldn't beat the third level. We cannot play the game for him, but maybe we can help him by telling how long it will take to beat the level.

It will be annoying if some obstacles passes through the start position and the end position, and Joon didn't want to be so mean to Taeyoung. So you may assume that such case does not exist.

## Input

The first line contains the size of the rectangle $N$ and $M(1 \leq N, M \leq 50)$. The following $N+1$ lines contain the information of the game. If $j$ th character of the $i$ th line is '.', it means that the ball can move to $(i-1, j-1)$. If it is '\#', then it means that the ball is not allowed to move to $(i-1, j-1)$. If it is 'S' or ' E ', then $(i-1, j-1)$ is the start position and the end position, respectively. The following line contains a single integer $K(0 \leq K \leq(N+1)(M+1)-2)$, the number of obstacles in the level. The $i$ th of the following $K$ lines contains the integers $a_{i}, x_{i}$ and $y_{i}\left(1 \leq a_{i} \leq 5,0 \leq x_{i} \leq N-a_{i}, 0 \leq y_{i} \leq M-a_{i}\right)$, the information of $i$ th obstacle as in the statement.

## Output

Print a single integer, the earliest time to beat the game. If it is impossible to win, then print 'INF'.

## Examples

## input

17
S...... .
. . . . . . . E
4
101
102
103
104

## output

INF

## input

17
S.......

```
........E
3
101
102
103
output
8
input
56
......S
.######
..##E.#
    .####.#
    .##.#.#
......#
1
50
output
39
```


## Note

In the input format of this problem, the downside direction is the positive $x$ direction, and the rightward direction is the positive $y$ direction. It is also possible that there is no obstacle $(K=0)$.


In the above example, it is impossible to go through 4 obstacles in a row. But it is possible if there are 3 obstacles.

## H. Highway Track

time limit per test 1 second memory limit per test 512 megabytes<br>input standard input<br>output<br>standard output

There is a circular highway track in Mr. Jeong's hometown. There are $N$ gas stations in this track, and the stations are numbered from 1 to $N$. One day, Mr. Jeong decided to drive one lab around this track, starting from a particular gas station. But his hometown is a bit strange, so the amount of gas that can be purchased at each gas station is fixed. Moreover, the sum of them is exactly equal to the amount of gas needed to drive one cycle around the track. Mr. Jeong is poor, so initially he has no gas at all. Therefore, if he chooses a bad gas station, his car may run out of gas before he can arrive to another gas station.


In this example, if he starts from the first station, he can safely turn around the track. No other gas station satisfies the condition.

Write a program to find the number of gas stations, that if Mr. Jeong starts from the station, he can safely drive around the track.

## Input

The input starts with a line containing an integer $N(1 \leq N \leq 500,000)$, the number of gas stations. The second line contains $N$ integers, $i$ th of which is an integer $o_{i}\left(1 \leq o_{i} \leq 1,000,000\right)$, the amount of gas that can be purchased from $i$ th gas
station. The third line contains $N$ integers, $i$ th of which is an integer $d_{i}\left(1 \leq d_{i} \leq 1,000,000\right)$, the amount of gas needed to drive from $i$ th station to $(i+1)$ th station. The sum of $o_{i}$ and the sum of $d_{i}$ are equal.

## Output

Write a single integer, the number of gas stations that satisfy the condition.

## Examples

## input

3
222
114
output
1

## input

4
2222
2222
output
4

## I. Impossible Design

time limit per test 1 second memory limit per test 128 megabytes
There are $N$ pilars around a circle. An integer from 0 to $N-1$ is written on each pillar. Each integer is written exactly once. For every pair of integers $(x, y)$ such that $0 \leq x<y \leq N-1$, we are going to connect the pillar with $x$ written and the pillar with $y$ written, using a stick that is parallel to the ground and placed at the height $x+y$. Of course, we can't do that if some sticks intersect, so we have to first check if there is any intersection before actually connecting the pillars. You may assume that the pillars are tall enough.

## Input

The first line consists of a single integer $N$, the number of pillars.
( $2 \leq N \leq 1,000,000$ ) The next line consists of a permutation of $0,1, \ldots, N-1$.

## Output

Print TAK if there is an intersection, otherwise NIE.

## Examples

## input

4
0123

## output

NIE
input
4
0132

## output

TAK

## J. Jeong Lab

time limit per test 1 second memory limit per test 512 megabytes
Mr. Jeong is a geeky chemist. He does a weird experiments with his chemical solutions. There are $N$ weird 1 L solutions in his laboratory, and each of them contains two kinds of chemical substances, A and B. Let $S$ denote the collection of his solutions. One of his experiments requires special 1 L solution containing $A$ and $B$, and he can make the special solution by one of following "mixing strategy".

1. Choose one of the solutions in $S$ and use it with a few treatments.
2. Choose two of the solutions in $S$, extract some portion from each solution so that the sum of the liters of them is 1 L , and mix them.

You can assume that each of $A$ and $B$ is evenly mixed in every solution. For example, if we extract 0.3 L from a 1 L solution with 5 units of $A$, it will contain 1.5 units of $A$.

According to the weird theory of Mr. Jeong, it is always better to have more A's and B's in a solution. Unfortunately, he couldn't figure out whether A is better than B or not. Anyway, he decided to call, the 1 L solution $K$ with the following property, as "bad solution".

Property: Using the solutions of $S$ and the above "mixing strategy", one can make a solution that, both the amount of $A$ and the amount of $B$ contained is greater than or equal to that of $K$.

However, Mr. Jeong thought that the solutions in his lab are not enough for the experiments. So we decided to get some new solutions for Mdays. On the $i$ th day, he does the following.

1. He carefully observes the new 1 L solution $K_{i}$. If it is a "bad solution", then he does not buy anything on that day.
2. Otherwise, he buys the solution and adds it to his collection $S$ of the solutions.

It is quite tedious for him to check whether each solution is "bad" or not, so he needs you help!

## Input

The input starts with the first line containing the number of solutions $N\left(1 \leq N \leq 10^{5}\right)$ that were initially in his lab. The $i$ th of the next $N$ lines contains two integers $a_{i}, b_{i}\left(0 \leq a_{i}, b_{i} \leq 10^{9}\right)$, the amount of the substances A and B , respectively, contained in the $i$ th solution. In the following line, the integer $M\left(1 \leq M \leq 10^{5}\right)$, the number of solutions he wants to check, is given. The $i$ th of the following $M$ lines contains two integers $c_{i}, d_{i}\left(0 \leq c_{i}, d_{i} \leq 10^{9}\right)$, the amount of A and B , respectively, contained in the solution $K_{i}$.

## Output

Print exactly $M$ lines, $i$ th of which contains a string "Yes" if he should buy the solution $K_{i}$, and "No" otherwise.

## Examples

input
1
11
3
13
12
21
output
Yes
No
Yes

## input

2
010
100
3
44
55
66

## output

No
No
Yes

## K. Kimino Ichi Wa

time limit per test 2 seconds
memory limit per test 256 megabytes
Feeling emptiness as usual, Taki and Mitsuha on their train to office saw each other through the window. They recognized each other at once, but the routes of the train were different. Now you are interested in the closest station to the starting point of the train line, among the stations that Taki and Mitsuha can meet.

Since they use special train tickets, there are the following features on the train route.

* Railways connecting each station are directed, and there is no loop connecting the same station.
* All stations on the route except the start and end points have the same number of incoming railways and outgoing railways.
* The starting point has only one outgoing railway, and the end point has only one incoming railway.
* For each station, there is at most one route starting from that station and visiting other stations at most once for each, then return to start point.
* There is always a path from an origin to an arbitrary station, and there is always a path from an arbitrary station to an end point.

If one reaches the end point, one must leave the train unconditionally and go to work. Because both want to meet as soon as possible, they keep moving on the train. The time required to travel between stations are same for every single railway. As Taki and Mitsuha are connected by special destiny, they can meet each other if they are on the same station. When the train route, Taki's initial position, and Mitshua's initial position are given, your job is to find the closest station to the start point of the train route that Taki and Mitsuha can meet each other.

## Input

Each station is represented by a single integer.

On the first line, the number of station N and the number of railways M are given(2 $<=N<=1,000,000)$. On the next $M$ lines, two integers $a$ and $b$ are given $(1<=a, b$
$<=\mathrm{N}$ ) which means there is a railway connecting station a to station b . On the last line, two different integers s and t are given ( $1<=\mathrm{s}, \mathrm{t}<=\mathrm{N}$ ) which are the number of Taki's initial station and Mitsuha's initial station.

## Output

Print the representing number of the station that is closest to the start point of the train route among the stations Taki and Mitsuha can meet each other. If they cannot meet, print "MUSUBI".

## Example

input
32
12
23
21
output
MUSUBI

## L. Labor

time limit per test 1 second memory limit per test 128 megabytes
Problem setter Onionpringles made his final contest problem on September 10, and then go to serve a military service on September 11. Unfortunately, there were so many strange predecessors in his troops. One of the predecessors found out that Onionpringles came from KAIST math-sci department, so he ordered Onionpringles a labor.
"We need a tent. Do you see the two parallel ropes and stakes supporting them? Now you select 4 stakes to build the tent. In our troops, there is a tradition for building tent; You sholud select stakes so they form a trapezoid with both basic angles to be acute or both basic angles to be obtuse. Keep building any trandition following tents. You can stop when I'm satisfied."

Feeling despair, Onionpringles started to calculating the number of different tents he need to make in the worst situation. Two tents are different if and only if the set of stakes that supoorting bottom of tents are different.

$\times$




The picture shows simple examples. Two examples on above are not satisfying the conditions while two examples on below are satisfying.

## Input

On the first line, the number of stakes $\mathrm{N}, \mathrm{M}$ on the two straight ropes and the y coordinates y1, y2 of the rope are given with spaces between them. ( $2<=\mathrm{N}, \mathrm{M}<=$ $100,000,-1,000,000,000<=y 1, y 2<=1,000,000,000, y 1 \neq y 2)$ The next $N$ lines are an integer representing the $x$-coordinate of the stake on the upper rope, and the
$M$ lines that follow are an integer representing the $x$-coordinate of the stake on the lower rope, one per line. The $x$ coordinate range is from $-1,000,000,000$ to $1,000,000,000$. Every stake has different position.

## Output

Print the remainder of the number of possible trapezoids divided by 1,000,000,007.

## Example

## input

233312
-1
1
$-2$
0
2
output
1

