Problem A. Maximal Convex Subset

Input file:	`convex.in`
Output file:	`convex.out`
Time limit:	1 second
Memory limit:	64 megabytes

For a given set of n points on the plane, find the maximal subset that can be a set of vertices of a non-strictly convex polygon.

Input

The first line contains an integer n ($3 \le n \le 50$). Then follow n lines with two numbers x_i, y_i — the coordinates of the points. All coordinates are integer and their absolute values are no more than 10 000.

Output

Output the maximal number of points that form a non-strictly convex polygon (three successive vertices may lie on a line; area of polygon can be 0).

Examples

convex.in	convex.out
4 0 0 1 0 0 1 1 1	4
4 0 0 1 1 1 10 10 1	3

Problem B. Death Star

Input file:	`deathstar.in`
Output file:	`deathstar.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

A long time ago in a galaxy far, far away The Merchant Federation fought against the rebels — a small group of defenders of the Republic. For the ultimate victory the Federation has decided to build the Death Star — a huge space ship, a weapon that the Universe had never seen before. They selected several planets and build a factory on each one so that all together they became a big circular conveyor: the details produced at the planet a_i were transported to the planet number a_{i+1} for $i < M$ and from the planet a_M to the planet number a_1 (naturally, all M planets are distinct). To somewhat reduce the cost of the project, the Federation also decided to transport the details only through the existing hypertunnels, each of which connects some two planets in a certain direction.

Having heard of the Death Star, the rebels realized that the war would not last for too long yet: either the Death Star would destroy them or they would destroy it (and the Federation as well). It's not very hard to understand that the Republic defenders consider satisfying only the second alternative...

The Analysis Center took the decision to destroy the Death Star at the construction stage. So they need to find out which planets were selected by the Federation. They already have the hypertunnel map. It's also known that for the years of tyranny, the Federation had managed to build exactly one hypertunnel among each pair of planets. In order to find out whether there is a factory on the planet or not, they had to send there a scout ship (the rebels really had very few of them). For this reason, first of all they needed to consider the matter theoretically, and only after that send scouts to the plantes which, as they thought, contained the factories most probably.

Actually, the most urgent question is to count how many different ways of selecting the locations for the factories and the transportation routes the Federation had. And since you worked in the Analysis Center, this task was assigned exactly to you.

Two ways are considered the same if their canonic forms do not differ. The canonic form is a sequence of M numbers $a_j, a_{j+1}, \ldots, a_M, a_1, \ldots, a_{j-1}$ such that a_j is minimal among all a_k and there are tunnels $a_j \to a_{j+1}, a_{j+1} \to a_{j+2}, \ldots, a_{M-1} \to a_M, a_M \to a_1, \ldots, a_{j-1} \to a_j$.

Input

On the first line of the input file there are two integer numbers: N — the number of planets in the Galaxy, M — the number of the factories the Federation intends to build ($1 \le N \le 1400$, $2 < M < 6$). On the second line of the input file there is a sequence of hexidecimal digits (`0..9`, `A..F`) which specifies the hypertunnel map in the ciphered way. In order to decipher it, one should to replace each digit with its binary representation (keep in mind that if the latter is shorter than 4 digits, it is padded with zeroes to 4 digits at the left end). Then one should take only first $N * (N - 1)/2$ digits and the rest of the sequence must be dropped.

The hypertunnel description is given in the following order: $(1{-}2), (1{-}3), \ldots, (1{-}n), (2{-}3), (2{-}4), \ldots, (2{-}n), \ldots, ((n$ where each pair $(a - b)$ has exactly one corresponding digit which is 1 when the tunnel from a to b exists and 0 otherwise (then there exists a hypertunnel from b to a).

Output

Output one integer number — the number of ways to select the locations for the factories. And may The Force be with you!

Example

deathstar.in	deathstar.out
4 3 B4	2

Comment on the sample:

After decipering the string "B4" we obtain the string "10110100". Dropping off 3 unnecessary trailing digits, we see that there are tunnels $1 \to 2$, $3 \to 1$, $1 \to 4$, $2 \to 3$, $4 \to 2$, $3 \to 4$.

And the required ways are: $1 \to 2 \to 3$, $2 \to 3 \to 4$.

Problem C. Elevator

Input file:	`elevator.in`
Output file:	`elevator.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Let's consider some building with N floors. There is an elevator in it, of course. It works in the following way:

1. if the button is pressed on some floor, the number of this floor is pushed into the calls queue;
2. if at the moment when button is pressed, elevator is not moving and the moving has not taken place or no more than A seconds have passed since the moment elevator stopped, then the elevator starts moving to the floor on which the button was pressed. This call will be the current call;
3. it takes B seconds for the elevator to move between consecutive floors;
4. if the elevator passes a floor, number of which is in the calls queue, it stops immediately (we will consider this moment as the moment when the corresponding call is processed);
5. after the stop one occurence of floor's number is removed from the calls queue, lift stays motionless during A seconds (time for passengers to enter and leave), and then continues moving by the current call;
6. if the elevator stopped at the floor of the current call, then the next earliest call becomes the current one;
7. if at the moment when the elevator starts motion a call from the current floor occurs, the elevator immediately stops again.

You are required to determine the moments of time when the calls will be processed. Initially the elevator is on the first floor.

Input

On the first line of the input file there are 4 integer numbers: N — the number of floors, M — the number of calls, A — the number of seconds for passengers to enter and leave the elevator, B — the number of seconds for the elevator to move between two consecutive floors, ($1 \le N, M \le 10^5$, $1 \le A, B \le 100$). The next M lines contain 2 integer numbers each: T_i — the number of seconds passed since the time moment 0 till the i-th call was made, F_i — the number of floor from which the i-th call was made ($1 \le T_i \le 10^5$, $1 \le F_i \le N$, $T_i \le T_{i+1}$ for all i in range $1..M-1$). In case when $T_i = T_{i+1}$, the i-th call is considered to be made earlier than the $(i+1)$-th call. The numbers on each line are separated by spaces.

Output

Output the moments when the calls were processed (the number of seconds passed since the moment 0 till the moment when the call was processed) in the order the calls were given in the input file, one per line.

Example

elevator.in	elevator.out
5 6 12 13 42 5 86 1 154 4 183 3 234 2 234 4	94 158 221 196 272 234

Problem D. Numbers

Input file:	`numbers.in`
Output file:	`numbers.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Given $2n$ distinct integer numbers in range from 1 to $2n$, find the number of ways to arrange them around the circle so that no two consecutive numbers $2i - 1$, $2i$ $(1 \le i \le n)$ are in the neighbouring positions. The arrangements that can be transformed one to another with rotations and symmetries are considered different.

Input

The input file contains a single integer number n $(1 \le n \le 500)$.

Output

Output the required number on the first line of the output file without leading zeroes.

Example

numbers.in	numbers.out
2	8

Comment:

For the sample test, the required arrangements are:

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

Problem E. Pareto's Domination

Input file:	`pareto.in`
Output file:	`pareto.out`
Time limit:	1 second
Memory limit:	64 megabytes

For any two points A and B, we will say that A dominates B in Pareto's sense if all coordinates of point A are strictly greater than corresponding coordinates of point B.

For a given set of n points in 3-D space find the number of points that are not dominated in Pareto's sense by any other point in this set.

Input

The first line contains an integer n ($1 \le n \le 50\,000$) then follow n lines with two numbers x_i, y_i, z_i — the coordinates of the points. All x_i are different, all y_i are different, and all z_i are also different. All coordinates are integers and their absolute values are no more than 10^9.

Output

Output the number of points that are not dominated in Pareto's sense.

Example

pareto.in	pareto.out
3 1 1 1 5 2 2 2 3 3	2

Problem F. Unexperienced Soldiers

Input file:	`soldiers.in`
Output file:	`soldiers.out`
Time limit:	1 second
Memory limit:	64 megabytes

Some soldiers stand in a column. Some of them are turned to the left end, and some of them are turned to the right end of the column. In the end of each second each pair of soldiers, who are turned face-to-face, turn to the opposite position. (Soldiers are unexperienced, so each soldier, who sees a face in front of him, thinks, that he stands wrong.) For example, if the initial position is:

`llrrlrlrrl`

('`l`' means a soldier turned to the left, and '`r`' means a soldier turned to the right), then the position after the first second is:

`llrlrlrrlr`

Then we get

`lllrlrrlrr`

and so on.

The question is to determine the position after m^{th} second, obtained from given initial position.

Input

The first line contains numbers n and m ($0 \le m < n \le 1\,000\,000$). n is the number of soldiers and m is the number of seconds during which soldiers turn. The second line describes the soldiers from left to right. Each soldier is desribed by '`l`', if he is turned to the left, or by '`r`' if he is turned to the right.

Output

You should evaluate the position of the soldiers after m^{th} second, obtained from the given initial position. To make the checking easy, you should output not the position itself, but the special function of it. Let's regard the standing as a binary number, where '`l`' means 1 and '`r`' means 0. For example, '`rllrl`' means 01101 in binary system, or 13 in decimal system. Output the residue from dividing this number by 2007. Residue should be given in decimal notation.

Example

soldiers.in	soldiers.out
10 2 llrrlrlrrl	932

Problem G. Stocks Dynamics

Input file:	`stocks.in`
Output file:	`stocks.out`
Time limit:	1 second
Memory limit:	64 megabytes

Do you want to get money for nothing? It's really easy. It's enough only to buy stocks of some company in one day and then sell them another day with higher price. Everything is fine, but you have to be sure that price for stocks you are going to buy increases. There is a pretty simple method to do this. First, you should make a record of prices for N days. Then you should calculate $N-1$ difference between every pair of consecutive records. Let's take a look on last $M > 0$ differences. If the same sequence of numbers exists somewhere in your records (excluding last one), then the difference you are going to estimate can be assigned to the difference that is located directly next to the last difference in the sequence you have found. For example, let the price records look like 1 2 2 4 3 4 4. The differences will be 1 0 2 -1 1 0. If $M = 2$ then you are looking for a sequence 1 0. It can be found at the beginning of the records. So we estimate next difference by value 2, and the price of stocks the next day by value 6. Of course, estimations are more precise when M is greater. So you have to take the maximal possible M such that there will be at least one sequence of M differences. If there are several sequences with the maximal M, you should calculate an average.

Input

First line contains an integer number N, $2 \le N \le 100\,000$ — the number of price records. Second line contains N integers X, $|X| \le 10^9$ — recorded price.

Output

Output one fraction — estimated price of the stocks the next day, or the word "`impossible`" (quotes for clarity only) if estimation can't be made. Fraction should be simplified. Denominator should be printed in any case.

Examples

stocks.in	stocks.out
9 1 2 2 4 5 5 8 9 9	23/2
3 1 2 4	impossible

Problem H. Vader's trouble

Input file:	`vader.in`
Output file:	`vader.out`
Time limit:	1 second
Memory limit:	64 megabytes

A long time ago in a far far galaxy... Darth Vader attacks some country by shooting missiles. Country lays on some rectangle $m \times n$. For one shot Vader chooses any two distinct rows and any two distinct columns: each of the 4 squares in the intersection of chosen rows and columns is hit by a missile. Some unit squares are empty, there is no reason to shoot at them. Other squares are important — it is sufficient to hit an important square by one missile to make it empty. (Yeah... Vader's missiles are powerful!) Vader likes shooting, so he wants to make the MAXIMUM possible number of shots. But he is reasonable and he chooses squares for each shot in such a way that among 4 chosen squares, there is at least one square that is important and not hit before.

Input

First line of the input file contains two integer numbers: m and n ($2 \le m, n \le 50$), separated by a single space. Then go m lines with n symbols each which represent the map of the country. Each symbol will be either '`e`' (to represent an empty square) or '`i`' (to represent an important square).

Output

Output one number — the maximum possible number of shots that Vader can make, such that a new important square is hit by each shot.

Examples

vader.in	vader.out
2 2 ii ii	1
4 5 eiiie iiiii iiiii eiiie	11

Problem I. War of the Empires

Input file:	war.in
Output file:	war.out
Time limit:	1 second
Memory limit:	64 megabytes

Each of the two sons of the Galaxy Emperor has got a part of the Empire. And of course they start to fight each other as soon as they get control over their territory. Wise Galaxy Emperor doesn't want them to act in such way, so the punishment must be blazing and ruthless. But how to determine, who has to be punished more and who less? An easy solution was proposed by court astronomers. It is just required to calculate the quantity of the proton bomb explosions on the territory of the first prince an on the territory of the second one. Location of explosions is a simple task and it has been already done for you. But how to determine whose territory they belong to?

Input

First line contains an integer number N, $0 < N \leq 100$ — the number of stars the first prince owns. Then follow N lines, each of which contains 3 numbers X, Y, Z, $-200 \leq X, Y, Z \leq 200$ parsecs — coordinates of the star. Then follows block in the same format that describes the stars owned by the second prince (with the same restrictions). And at last number of explosions Q, $Q \leq 10\,000$ is given, followed by Q lines with explosion coordinates in the same format. All numbers in the input are integer.

Princes own the high-tech weapons and they do not want to waste very expensive proton bombs, so you may assume that any explosion is always inside of the convex hull (including border!) of the stars one of the princes control (we called it territory before). Also you may assume that distance between territories of the princes (between convex hulls) is not less than 1 parsec.

Output

Output one line containing Q digits. i^{th} digit should be 1 if explosion was on the territory of the first prince, or 2 otherwise.

Example

war.in	war.out
4 0 0 0 3 0 0 0 3 0 5 5 -5 4 0 0 1 3 0 1 0 3 1 1 1 6 4 0 0 0 0 0 1 1 1 0 1 1 2	1212

Problem J. One Way

Input file:	`way.in`
Output file:	`way.out`
Time limit:	1 second
Memory limit:	64 megabytes

Each of the two sons of the Galaxy Emperor has got a part of the Empire. And of course they start to fight each other as soon as they get control over their territory. Galaxy Emperor must deal with it (see problem "War of the Empires"), and has no time to deal with easier problems. One of them is that he often moves between his native planet (he spends most of time there thinking about the meaning of life) and the planet where the council of war resides. Now he has got the notification that one of his sons wants to destroy the way between these two places (sure, with the help of the proton bombs). And he wants to know the probability that he will get to the council from home after the bombs explode.

There are many ways for the Emperor to travel from home to the council, they are described as follows:

- From home he can move to any of the nearby planets.
- From each of these planets he can travel to any one planet near the council.
- From any of those planets he can travel to the council planet.

After the bombs explode, some of the connections between planets will be lost. So, will there be any way after it all?

Input

First line contain two numbers N and M, $1 \le N, M \le 11$ — the number of planets near home and near the council. Then follows the line with N real numbers between 0 and 1, i^{th} number indicating the probability that Emperor can move from home to the i^{th} planet near home. Then follows the line with M real numbers between 0 and 1, i^{th} number indicating the probability that Emperor can move from the i^{th} planet near the council to the council planet itself. Then N lines with M numbers each follow; j^{th} number in i^{th} of these lines is the probability that he can get from i^{th} planet near home to j^{th} planet near the council.

Note that the Emperor can not move from planets near council to planets near home.

Output

Output the probability of the existence of the way from home to the council. The answer must be accurate up to 10^{-4}.

Example

way.in	way.out
2 1 0.5 0.5 0.5 0.7 0.3	0.22375

Problem K. Wheels

Input file:	`wheels.in`
Output file:	`wheels.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

We have 2 equal sized wheels with spokes. There are N spokes in the first wheel and M ones in the second. For the purposes of this task we consider that the distances between consecutive spokes of a wheel are the same. Wheels are located on the same axle, so their projections on the plane of view coincide and their common border looks like a single circle.

First wheel makes a full turn clockwise on 360 degrees in A seconds, second wheel — in B seconds. You need to count the number of moments, when whole picture looks like a single wheel, which took place during T seconds since motion has started. Remember that the distances between consecutive spokes of a wheel should be the same.

The motion begins from the position, when at least two spoke's projections coincide.

Input

On the first line of the input file you'll find integer numbers N, M, A, B, T ($1 \le N, M, A, B, T \le 10^6$).

Output

Output only one number — the answer for the task.

Example

wheels.in	wheels.out
2 2 5 10 10	5