The 15-th BIT Campus Programming Contest

Sponsored by

连山科技
LSSEC TECH

Onsite Round

Problem List

A	IQ difference
B	Residue Problem
C	Simple AniPop
D	Life Game
E	Eigen Substring
F	SVM
G	Nim plus
H	Treasure Hunt
I	Nobody Knows Better Than Me
J	Teacher Long and Machine Learning
K	Number Puzzle
L	Long Long Wanna Buy
M	Camouflage

Do not open before the contest has started.

2020/11/29

Problem A. IQ difference

Input file:	`standard input`
Output file:	`standard output`

The registration for the 15-th BIT Campus Programming Contest has started. Students can't wait to form teams to join this competition. But high intensity and cooperation are required during this match, excluding some specialists teams up with his (her) girlfriend (boyfriend). So you will find that it's hard to communicate between pupil and grandmaster, especially coding in XCPC.

We define the *IQ difference* of a team as the differece between the maximum and minimum IQ of all team members. In other words, if the IQs of three participants are c_i, c_j, c_k, then this team's *IQ difference* is $\max\{c_i, c_j, c_k\} - \min\{c_i, c_j, c_k\}$.

Through the second law of cosmology, we know that the *IQ difference* of one XCPC team cannot exceed a balanced constant C. Now you are given a number of students who want to participant in the 15-th BITCPC, please help them to find out how many teams can be teamed up at most. Note that each team should exactly contains three participants.

Input

The input contains two lines, the first line contains two positive integers n $(1 \le n \le 1\ 000)$ and C $(1 \le C \le 50)$ — indicate the number of students and the equilibrium constant under the second law of the universe.

The second line contains n separated non-negative integer $c_1, c_2, \cdots, c_n$ $(0 \le c_i \le 200)$ — indicate the IQ of these students.

Output

Please output a non-negative integer as the answer — indicating the maximum number of teams that this group of students can form.

Example

standard input	standard output
6 2 0 1 2 4 5 5	2

Note

For the first example, the first 3 students form a team with an *IQ difference* of 2, and the last 3 students form another team with an *IQ difference* of 1, so that all students can participate in the competition!

Problem B. Residue Problem

Input file:	`standard input`
Output file:	`standard output`

Let function $f(i, r, P)$ equal the number of pairs of $(a, b), 0 \le a, b < P$ that satisfy the equation $a^r(b + b^2)^i \equiv b^i (\bmod P)$, where P is a prime. Given m queries, each with Y, N, r, P, answer $\prod_{i=1}^{N} Y^{f(i,r,P)}$ mod $10^9 + 7$.

Input

First, an integer $m(1 \le m \le 10^5)$, indicating the number of queries.

The following m lines, each with four integers Y, N, r, P ($10^5 \le Y < 10^9 + 7$, $1 \le N \le P$, $1 \le r \le 3$, $3 \le P < 10^9 + 7$), indicating the input of this query.

Output

Output m lines, each line with an integer standing for the answer to the corresponding query.

Example

standard input	standard output

4
100000 1234567 1 999999937
100000 7654321 2 999999929
100000 7777777 3 999999893
100000 7777777 3 999999757
	968518472
	236321637
	770733917
	509589974

Problem C. Simple AniPop

Input file: `standard input`
Output file: `standard output`

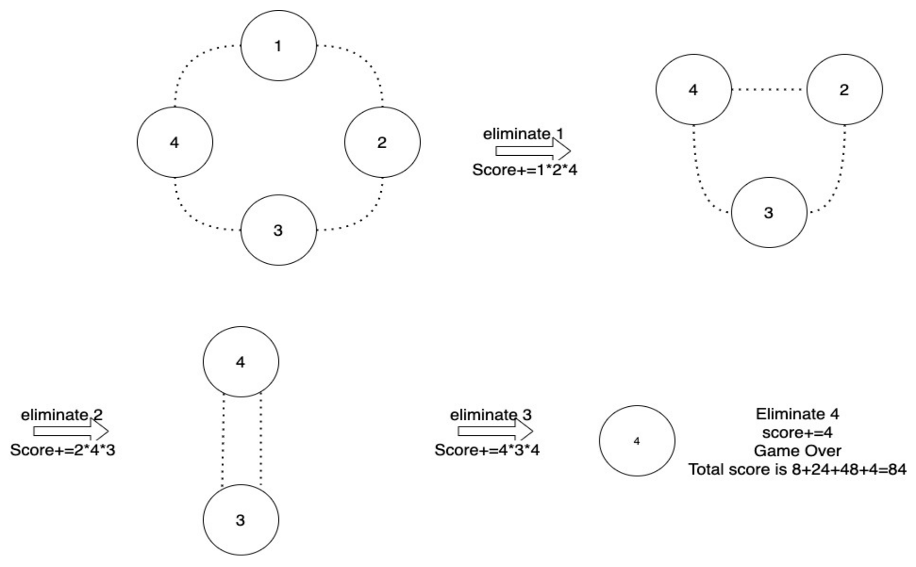
A few years ago, a popular puzzle game named *AniPip* that particularly appealed to older people like *XIRHXQ* was hot. And *XIRHXQ* tries his best to get the highest scores. However, the game is full of randomness and the rules are too complex, so *XIRHXQ* has a hard time finding an absolutely optimal way to eliminate all the blocks. As a result, *XIRHXQ* invented a game with simpler rules as a daily pastime, in which he was able to calculate the highest score he could obtain in this game within the time complexity of $O(1)$.

The rules are as follows.

- There are a number of objects that need to be eliminated, arranged to form a **circle(ring)**.
- The player can choose one object per move to eliminate.
- Each object is given a value, and when you choose to eliminate an object, you get the score whose value equals the **product** value of the values of **that object and the two objects adjacent to it** when it is eliminated. (until there exists only one object left)
- When there is only one object left, the player can eliminate the last object and achieve the score of **the last object's value**, and then the game is over. The final score is the sum of the scores you get during the game.

Though *XIRHXQ* can calculate the highest score which the player can achieve immediately, he still gives you the assignment because he wants to play *AniPop*. When you get the objects and values, you should work out **the highest score** a player can get, and of course, without the limitation of time complexity of $O(1)$.

When there are 4 objects and the values are [1,2,3,4] respectively, players can eliminate objects in the order shown below and end up with 84 points!



Input

Two lines.

The first line exists a number n $(1 \le n \le 500)$ which means there are n objects.

The second line contains $a_i(1 \le i \le n, 1 \le a_i \le 100)$, which are the values given to the objects.

Output

A number which is the highest score the player can get.

Examples

standard input	standard output
4 1 2 3 4	84
10 45 29 8 3 32 54 88 68 70 83	2304371

Problem D. Life Game

Input file:	`standard input`
Output file:	`standard output`

Scientists try to figure out what was the initial biosphere like. So they design an experiment with serval micro-creatures in a matrix grid and observe their behavior.

A grid point can exist at most 1 micro-creature and there are only two states of their life activities, which are `*` for alive and `.` for death.

At each time point, all the micro-creatures will transfer between two states. And the transformation will only depend on the states of the 8 micro-creatures around it.

Let l, r ($0 \le l \le r \le 8$) be the suitable living range of the micro-creatures. Only if the number of alive micro-creatures around it is not greater than r and not less than l, the creature will become or keep alive, otherwise it will keep death or die due to overcrowding or loneliness.

Now, you are given the initial state of the micro-creatures in a matrix grid, please predict what will them be looked like after a while.

Input

The first line gives two integers $n, m (1 \le n, m \le 100)$, which means that there are $n \times m$ grids on the matrix grid.

The second line gives two integers $l, r (0 \le l \le r \le 8)$, which represents that the it is suitable for survival when the number of living creatures around the creature is $[l, r]$.

The third line gives only one integer $t (1 \le t \le 10^3)$, which means the time point you should predict.

In the next n line, each line gives m characters `*` or `.` to represent there is life or no life on the grid in the initial state.

Output

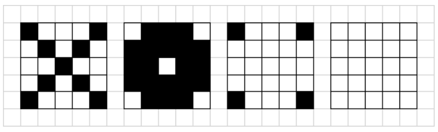
A matrix of $n \times m$ composed of `*` and `.` to represent the state at the time t. Please notice that the initial state is considered as time 1.

Examples

standard input	standard output
5 5 2 3 8 *...* .*.*. ..*.. .*.*. *...*	....
5 5 3 6 8 *...* .*.*. ..*.. .*.*. *...*	***** *...* *...* *...* *****

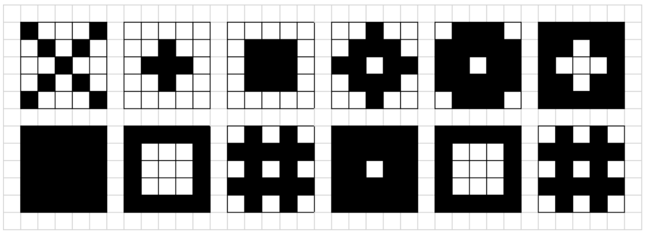
Note

The evolution of case 1 is as follows:



It will finally stop in the last state.

The evolution of case 2 is as follows:



Finally it loops through the three states.

Problem E. Eigen Substring

Input file:	`standard input`
Output file:	`standard output`

For any string s, we define the substring $s[l..r]$ an **eigen** substring of s if and only if $s[l..r]$ only appears once in s.

You are given a string s. Please calculate the length of the shortest eigen substring for each prefix of s.

The notation $s[l..r]$ represents the substring of s that spans from the l-th character in s to the r-th character in s, inclusive.

Input

The input contains two lines.

The first line contains an integer n $\left(1 \le n \le 10^6\right)$ which is the length of string s.

The second line contains the string s. It's guaranteed that all characters in s are lowercase English letters.

Output

Your output should contain n lines. The i-th line of your output should contain one integer which denotes the length of the shortest eigen substring of $s[1..i]$.

Example

standard input	standard output
5 ababb	1 1 1 2 2

Note

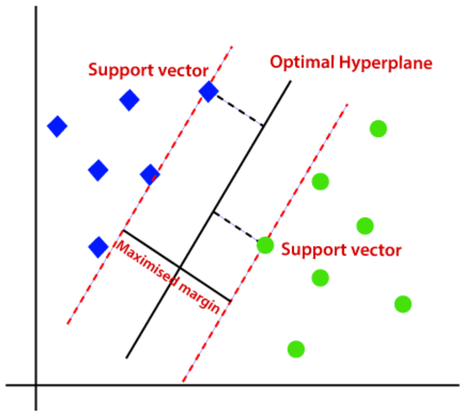
For the example test case:

- The shortest eigen substring of $s[1..1] = \mathrm{a}$ is `a`;
- The shortest eigen substrings of $s[1..2] = \mathrm{ab}$ are `a` and `b`;
- The shortest eigen substring of $s[1..3] = \mathrm{aba}$ is `b`;
- The shortest eigen substring of $s[1..4] = \mathrm{abab}$ is `ba`;
- The shortest eigen substrings of $s[1..5] = \mathrm{ababb}$ are `ba` and `bb`.

Problem F. SVM

Input file:	`standard input`
Output file:	`standard output`

Lancern is learning Support Vector Machine(SVM), a widely known linear classification model, recently. A set of data points(represent as high dimension points) are linear divisible if there exists a separating hyperplane which divides all data points into two parts, data points on the one side of the separating hyperplane all belong to one class, while data points on the other side all belong to the other class, and no data points are on the separating hyperplane. The SVM aims to find a separating hyperplane that satisfies the minimum Euclidean distance from any data point to the separating hyperplane is maximized. Those points with the minimum distance are called support vectors, and the distance between support vectors is called **margin**(also equals double minimum distance from support vector to the optimal separating hyperplane).



Poor Lancern doesn't understand SVM at all, neither do you, but the deadline of his assignment of SVM is coming, so Lancern asks you to help him solve his assignment.

In his assignment, all data points are given in 3-dimension real space, and each point only belongs to one of the two classes. You should help Lancern determine whether the data points are linear divisible? If they are, you have to calculate the margin.

Input

The first line contains one integer n ($2 \le n \le 500$), indicating the number of data points.

The following n lines, each line contains 4 integers x_i, y_i, z_i ($|x_i|, |y_i|, |z_i| \le 1\,000$) and c_i ($c_i \in \{1, 2\}$), denoting the coordinate of data point and its class.

It is guaranteed that no two data points have the same coordinate and each class has at least one data point.

Output

If the data points are linear divisible, output the margin. Otherwise, output `-1` instead.

Your answer is considered correct if and only if $\frac{|a-b|}{\max(1,a)} \le 10^{-6}$, where a is the jury's answer, b is your output.

Examples

standard input	standard output

8
0 0 0 1
1 0 0 1
1 1 0 1
0 1 0 1
0 0 1 2
1 0 1 2
1 1 1 2
0 1 1 2
	1.00

8
0 0 0 1
1 0 0 2
1 1 0 1
0 1 0 2
0 0 1 1
1 0 1 2
1 1 1 1
0 1 1 2
	-1

Note

In the first example, obviously, $z = 0.5$ is the optimal separating plane.

In the second example, the data points are not linear divisible.

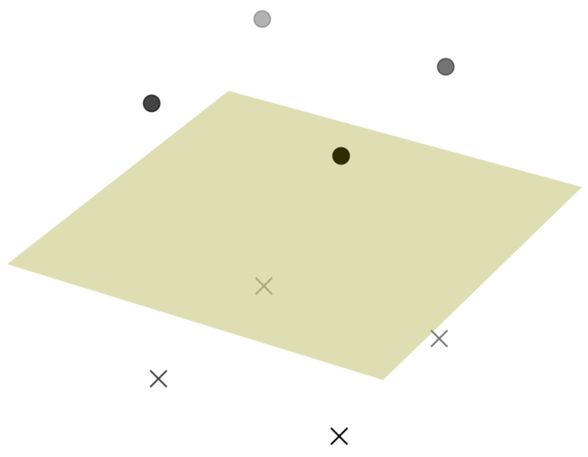


Fig. 1. first example

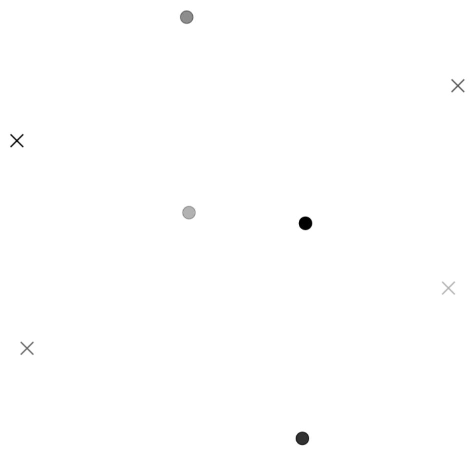


Fig. 2. second example

Problem G. Nim plus

Input file:	`standard input`
Output file:	`standard output`

Long Long and Mao Mao both have AKed the new div.1 round in Codeforces tonight, but it's too early to sleep now. So they decide to play one special Nim game. In this special nim game, there are n paper balls on the table at the beginning of the game. But unlike some simple nim game, they each have m numbers in a set. They could only take away the number of paper balls which occur in their set. In the end, those who cannot take away any paper ball lose this game.

Long Long takes the first, and he would like to know whether he can win this game, please help him.

Input

The first line contains two positive integers n ($1 \le n \le 5\ 000$) and m ($1 \le m \le 100$) — the number of the paper balls and their sets' size.

The second line contains m positive integers $a_1, a_2, \cdots, a_m$ ($1 \le a_i < a_{i+1} \le 500$) — the numbers in Long Long's set.

The third line contains m positive integers $b_1, b_2, \cdots, b_m$ ($1 \le b_i < b_{i+1} \le 500$) — the numbers in Mao Mao's set.

Output

Print `Long Long nb!` if Long Long can secure the victory, and `Mao Mao nb!` otherwise.

Examples

standard input	standard output

5 1
6
7
	Mao Mao nb!

20 3
3 7 10
2 6 7
	Long Long nb!

Note

In the first sample, Long Long can only take 6 paper balls, but there are only 5 remains on the table, so he loses this game.

In the second sample, Long Long could take 10 paper balls first, and 10 remains on the table:

- If Mao Mao takes 2 paper balls, Long Long could take another 7 paper balls. Then there is only one paper ball that remains on the table, the winner is Long Long.
- If Mao Mao takes 6 or 7 paper balls, Long Long can also take 3 paper balls to make one paper ball or empty left on the table. In this case, the winner is Long Long.

Problem H. Treasure Hunt

Input file:	`standard input`
Output file:	`standard output`

The double twelve is upcoming, and little LC has a lot of clothes, cosmetic and skincare products to buy. Since the double eleven just ended, LC is a little tight on money, so she has to make money to satisfy her willingness to purchase.

LC finds a game called Treasure Hunt, a roguelike game, in which players can exchange game currency to real-life money. In this game, LC is given a directed acyclic graph(DAG) with n nodes. LC can play as many turns as she wants. At the beginning of each turn, LC is at node 1, she has to reach node n to end this turn. When LC reaches node i, she will get a_i gold coins as reward. However, every road j has a pass limit c_j, which means if this road has been passed in c_j turns, this road can no longer be used again.

LC wants to know the maximum gold coins she can earn from this game.

Input

The first contains two integers $n\,(2 \le n \le 100)$ and $m\,(1 \le m \le 500)$, representing the number of nodes and the number of roads.

The second line contains n integers $a_1, a_2, \cdots, a_n\,(1 \le a_i \le 100, i = 1, 2, \cdots, n)$.

The next m lines, each line contains 3 integers $u, v\,(1 \le u, v \le n)$ and $c\,(1 \le c \le 100)$.

Output

Output one integer, the maximum gold coins LC can get.

Example

standard input	standard output

4 4
1 4 3 2
1 2 1
1 3 1
2 4 1
3 4 1
	13

Problem I. Nobody Knows Better Than Me

Input file:	`standard input`
Output file:	`standard output`

It's election year in Byteland! This year, Iden and Rump are competing for becoming the next president of Byteland. The election rule of the Byteland is pretty simple. Each citizen in the Byteland has the right to vote for his preferred president. The Byteland consists of several states. Internal to some state, if the total number of citizens in this state voting for Iden is greater than the total number of citizens in this state voting for Rump, then Iden gets one score from this state, and vise versa. Finally, the one that gets more score wins and will become the next president of Byteland.

Rump really wants to win the election. Since nobody knows election better than Rump, Rump has prepared the election data from the last election. The data contains the percentage of citizens in each state that voted for Rump and Illary (the opponent to Rump in the last election) in the last election. With some investigation, Rump finds that he can change citizens' mind in this election by giving impressive speeches. Specifically, for a state Rump lost in the last election (i.e. the total number of citizens in this state voting for Illary was greater than the total number of citizens in this state voting for Rump in the last election), Rump can spend $\lceil \frac{l-w}{2} \rceil$ units of time in this state giving a speech to turn citizens' mind and ensure getting score from this state this time. If Rump doesn't spend enough time in this state, then he will lose this state again this time. l is the percentage of citizens that voted for Illary and w is the percentage of citizens that voted for Rump in the last election. For a state that Rump won in the last election, Rump is very sure that he will also win this state this time and so Rump doesn't have to make efforts on these states this time.

Rump's time is very limited since he is busy tweeting every day. He only has t units of free time which he can spend for traveling around and giving speeches to citizens. He wants to know whether he has the opportunity to win the election this time. Can you help him?

Input

The first line of input contains two space-separated integers n and t $\left(1 \le n \le 2 \times 10^5, 0 \le t < 2^{31}\right)$ which represents the number of states in the Byteland and the total units of free time Rump has, respectively. It's guaranteed that n is odd so there couldn't be a tie between Iden and Rump.

Then there are n lines of input, the i-th line of which contains two space-separated integers l_i and w_i $(0 \le l_i, w_i \le 100, l_i + w_i = 100, l_i \ne w_i)$ which represents the percentage of citizens who voted for Illary and Rump in the last election, respectively.

Output

If Rump has the opportunity to win the election this time, output `Make Byteland Great Again!` on a separate line; otherwise output `Fake Polls!` on a separate line.

Examples

standard input	standard output

5 10
40 60
49 51
55 45
53 47
36 64
	Make Byteland Great Again!

5 4
49 51
53 47
55 45
56 44
62 38
	Fake Polls!

Note

In the first given example test case, Rump doesn't have to make any effort to ensure his winning since he is very sure that he will win the first, second and fifth state;

In the second given example however, Rump has no way to win the election since he does not have enough time.

Problem J. Teacher Long and Machine Learning

Input file:	`standard input`
Output file:	`standard output`

Teacher Long is currently studying machine learning. Today, he focuses on a question about regression.

'Suppose a polynomial F satisfies F(1)=1, F(2)=4, F(3)=9, F(4)=16, F(5)=25, then F should be:'

$F(x) = x^2$ (Correct)

'Suppose a polynomial F satisfies F(1)=1, F(2)=4, F(3)=9, F(4)=16, F(5)=24, then F should be:'

$F(x) = -\frac{1}{24}x^4 + \frac{5}{12}x^3 - \frac{11}{24}x^2 + \frac{25}{12}x - 1$ (Not Exactly Correct)

Correct answer:

$F(x) = x^2$, since there is noise of value 1 in $F(5)$.

Teacher Long felt so bad, and decided to write a code to help him automatically calculate this kind of problem.

Suppose there is a fourth-order polynomial $F(x) = a_4x^4+a_3x^3+a_2x^2+a_1x+a_0$ and a given sequence $\{f_1, f_2, f_3, f_4, f_5\}$, satisfying that the coefficients a_i are all in range $[-100, 100]$ and are all integers, and for $\forall i \in [1, 5], |F(i) - f_i| \leq 1$. Given $\{f_1, f_2, f_3, f_4, f_5\}$, calculate $\{a_0, a_1, a_2, a_3, a_4\}$.

Input

There are multiple test cases in input data.

An integer $T(1 \leq T \leq 10)$ is in the first line, standing for the total number of test cases.

Next is T test cases.

For each test case, the input data is one line with five numbers, with f_1, f_2, f_3, f_4, f_5 respectively.

It is guaranteed that there is at least one valid answer.

Output

Print T lines. Each line is the answer of one test case, with a_0, a_1, a_2, a_3, a_4 respectively.

Example

standard input	standard output

3
1 4 9 16 25
1 4 9 16 24
25 16 9 4 1
	0 0 1 0 0
	0 0 1 0 0
	36 -12 1 0 0

Problem K. Number Puzzle

Input file: `standard input`
Output file: `standard output`

This is an interactive problem!

Lancern is a cryptography master, he never uses name+birthday combination as his password, instead, he uses a randomly generated number. To avoid forgetting his password, he stores his password in a number puzzle, in which, there is a $n \times m$ jigsaw, but with $n \times m + 1$ pieces of blocks, each has a number on it, among them, $n \times m$ pieces can constitute the jigsaw, and the number on the extra piece is Lancern's password. Let's use $a_{i,j}$ to represent the number of the piece on the i-th row and j-th column of the jigsaw, it is satisfied:

- $a_{1,j} < a_{i,j}$, for all $i = 2, \cdots, n$ and $j = 1, \cdots, m$
- $a_{i,j} < a_{k,j+1}$, for all $i, k = 1, \cdots, n$ and $j = 1, \cdots, m - 1$

Today, Lancern forgets his password again, but he is too busy to solve this puzzle. He really needs the password, so could you help him to solve the puzzle?

At the beginning, you will be given $k = n \times m + 1$ **distinct** numbers, represent the jigsaw pieces. Each time, you can use one of the following queries to check:

- $r\ x\ y$: whether the piece with number x and the piece with number y is horizontally adjacency.
- $c\ x\ y$: whether the piece with number x and the piece with number y is vertically adjacency.
- $!\ x$: confirm x is the password.

After each query, you can use `fflush(stdout)` in C/C++, `cout.flush()` in C++, `System.out.flush()` in Java to flush the output stream.

For each `r x y` and `c x y` query, you will get a 'yes' or 'no' as input. The number of `r x y` and `c x y` queries you made can't exceed $2k$. After you confirm the answer using `! x` query, you have to **terminate your program immediately** after you flush the output stream.

Input

Use standard input to read the initial input and the responses to the queries.

The input contains an integer k ($k = n \times m + 1, 2 \le n \le m \le 100$), and k integers a_i $\left(0 \le a_i \le 10^9\right)$ at the beginning.

Interaction Protocol

To make the queries your program must use standard output.

Your program must print the `r x y`, `c x y` and `! x` queries, each query per line. After printing each line your program must perform operation flush.

If you print the `r x y` or `c x y`, Jury will response ‘yes’ or ‘no’ to your queries.

The testing system will allow you to read the response on the query only after your program print the query for the system and perform a flush operation.

Each x and y in your queries must exist in a_i in the initial input, or you may not get any response.

Example

standard input	**standard output**
`10`	
`0 1 2 3 4 5 6 7 8 9`	
	`r 0 1`
`no`	
	`c 0 2`
`yes`	
	`r 0 3`
`yes`	
	`r 3 4`
`no`	
	`c 3 5`
`no`	
	`r 3 6`
`no`	
	`r 3 7`
`yes`	
	`c 3 4`
`yes`	
	`r 2 4`
`yes`	
	`c 1 2`
`yes`	
	`r 1 5`
`no`	
	`r 1 6`
`yes`	
	`! 5`

Note

In the example, the origin jigsaw is

$$\begin{array}{ccc} 0 & 3 & 7 \\ 2 & 4 & 9 \\ 1 & 6 & 8 \end{array}$$

The extra number is 5.

Problem L. Long Long Wanna Buy

Input file:	`standard input`
Output file:	`standard output`

Long Long wanna buy a new computer. Long Long thought he was a master, so he decided to buy accessories and assemble his own computer. But actually Long Long is a rookie, so he only cares about the price and service life of accessories. There are N kinds of accessory and M accessories for each kind. To assemble a computer, you need all the kinds of accessory and each kind only need one. When any one of the accessories reaches the service life, the computer is damaged. Long Long defines the **price per unit time** of his computer is the total price he spent divided by the life of the computer.

Now we will give you all the accessories, and you need to assemble a computer that the **price per unit time** is the lowest.

Input

The first line contains two integers $N, M(1 \le N, M \le 1\,000)$, which means the number of accessory kinds and the accessories number in each kind.

The next N line(s) contain $2 \times M$ numbers, $S_1, P_1, S_2, P_2 \cdots S_M, P_M$ $(1 \le S_i \le S_{i+1} \le 1\,000$, $1 \le P_i \le 1\,000\,000)$, S_i means the i-th accessory service time, and P_i means the price.

Output

Please print only one number, which means the total price of your computer. If there is more than one combination make the **price per unit time** is lowest, print the minimum total price.

Example

standard input	standard output

`3 3`
`1 2 2 2 3 5`
`3 5 6 2 7 8`
`2 4 3 4 3 5`
	`11`

Problem M. Camouflage

Input file:	`standard input`
Output file:	`standard output`

Long Long always dreamed of being a cowboy, so he made a robotic cow to ride it for daily driving.

But unfortunately, this robotic cow has not been painted with cow camouflage. Long Long bought paper and paints from the grocery store to make his own cow camouflage.

However, the vicious Master Yi splashed some white paint on the camouflage while Long Long was away, causing many spots on Long Long's cow camouflage.

Poor Long Long has little black paint left. He can only choose to blacken some of the smaller white spot(s) to make his camouflage looking as natural. And Long Long also has a obsession that he will only repaint the white pixel(s) wrapped by black pixel(s), which means after repainting, all the repainted pixels are surrounded by black pixels, while the white pixels outside including the sides and angles will never be repainted. The pixel surrounded is defined as four direction including left, right, up and down.

Input

The first line contains three integers n, m ($1 \le n, m \le 1\ 000$) and d ($1 \le d \le n \times m$) — the height and weight of Long Long's cow camouflage and the spots' area limit that is chosen to paint.

The next n line(s) contain m character(s) each, where the j-th character of the i-th line is $c_{i,j}$ ($c_{i,j}$ is either `.` if the pixel in (i, j) is white or `#` if the pixel in (i, j) is black).

It is guaranteed that each spot does not cross with any borders.

Output

Plase print n line(s) with m character(s) each, which shows the cow camouflage you painted.

Examples

standard input	standard output
7 6 5 ....##.. .#.#.. .#.#.. .#..#. ..##..	....##.. .###.. .###.. .####. ..##.. ......
10 10 3 ....##..... ..####.... .##..##... .######... .####..... .#..#.###. .##.#.#.#. ..###.###.	....##..... ..####.... .######... .######... .####..... .#..#.###. .##.#.###. ..###.###. ....