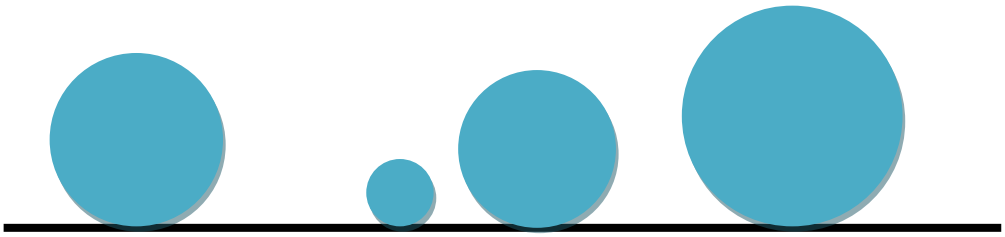
Problem : 2D-Solar System

The 2D-solar system like our solar system comprises Bigsun (its sun) and its planetary system of many circular planets orbiting around Bigsun. Due to the high gravity of Bigsun, all planets have been attracted by Bigsun. Precisely, they orbit around Bigsun while being tangent to it as depicted in the figure (As Bigsun is so huge, its boundary looks like a line.) Surprisingly, up to the current time no two planets have collided with each other, but no one knows whether the system is free of collisions in the future. You are to write a program to verify whether there is a possibility of any collision in the future and if so, compute the time at which the first collision happens. The scientists of NASA have realized that each planet in the 2D-solar system moves with a constant velocity. More precisely, it turned out that the motion equation of a planet can be described by the position of its touching point with the boundary of Bigsun through time by the linear equation $y = at + b$ where a and b are two known parameters and t denotes time.



Input (Standard Input)

There are multiple test cases in the input. Each test case starts with a line containing an integer n ($0 \leq n \leq 50{,}000$) where n is the number of planets. The i^{th} line of the next n lines contains 3 space-separated integers r_i, a_i, and b_i whose absolute values are not exceeding 1,000,000,000. The number r_i which is a positive square number, denotes the radius of Planet i and a_i and b_i specify its motion equation, i.e. the position of the tangent point of the planet on the boundary of Bigsun at time t is $a_i t + b_i$. The input terminates with a line containing "0" which should not be processed.

Output (Standard Output)

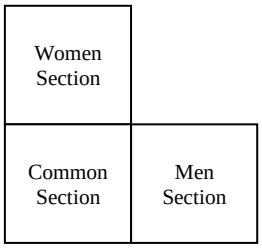
For each test case, output a line containing the time at which the first collision happens under the assumption that the current time is equal to 0 and all planets are disjoint at the current time. If the system is free of collisions you must output "Collision-Free System". The output must be rounded to exactly two digits after the decimal point.

Sample Input and Output

Standard Input	Standard Output
3 1 1 1 4 3 6 9 -7 30 2 4 -1 1 1 1 7 2 1 1 10 1 2 5 0	1.20 Collision-Free System 3.00

Problem : Wedding Hall

Kamran has recently bought a rectangular flat garden in an awesome part of the countryside. His business plan is to construct a Hall to host wedding ceremonies, since the countryside has recently attracted a lot of attention for being a fantastic area to host wedding ceremonies. As women and men sections must be separated based on the nation law, he thinks of designing the hall in three sections: men section, women section and common section (including rest rooms, dinner room and etc). As the common section must be easily accessible to all persons, it must be designated in the middle of the other two sections. Among several proposal designs, Kamran has selected the one depicted below where all three sections are squares of the same size, they are attached to each other like an L shape, their sides are parallel to the garden sides, and the visible sides of the common section from outside face the south and west of the garden. Now the main question is where the hall must be constructed. The garden is full of old trees and cutting the trees is forbidden due to high air pollution. He kindly asks you to help him to find the largest hall that he can construct.



Input (Standard Input)

There are multiple test cases in the input. Each test case starts with a line containing a non-negative integers n ($1 \leq n \leq 50{,}000$) and two positive integers a and b (all not exceeding 1,000,000) where n is the number of distinct trees in the garden and a and b specify the sides of the garden. Precisely, $[0, a] \times [0, b]$ denotes the rectangle modeling the garden. The next n lines, each contains 2 space-separated non-negative integers x_i and y_i ($0 < x_i < a$, $0 < y_i < b$) denoting the x and y coordinates of a tree, respectively. You may assume that trees have distinct coordinates and the south side (i.e. $[0, a]$) and the west side (i.e. $[0, b]$) of the garden lie on the x-axis and the y-axis, respectively. The input terminates with a line containing "0 0 0" which should not be processed.

Output (Standard Output)

For each test case, output the area of the largest hall that Kamran can construct in his garden. Note that the hall can touch the trees or the garden sides but it can't interiorly include them. The output must be rounded to "exactly" two digits after the decimal point.

Sample Input and Output

Standard Input	Standard Output
2 3 5 2 2 1 4 0 0 0	6.75

Problem : Homotopic Paths

Soroush plans to walk through a predefined path in the birds garden from the entrance point to the exit point to enjoy watching birds on trees. His playful dog on a leash does not necessarily follow the same path but he finally reaches the exit point. The garden is full of tall trees, and there is no other obstacle. The leash is constructed in such a way that its length can vary from 0 to any length but it tends to have the smallest length at any time. Soroush and his dog start their journey from the entrance point with the leash length to be zero. Soroush is wondering by knowing his dog's path in advance, if it is possible for the leash length to be zero when they both arrive at the exit point. Note that the leash can not be paased over the trees. If the leash length is zero at the exit point, the Soroush's path and his dog's path are called homotopic. Your task is to write a program to determine whether two given paths are homotopic.

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains three positive integers n, m and k ($0 \leqslant m, k \leqslant 1,000$ and $1 \leqslant n \leqslant 1,000$) where n is the number of trees, and m and k are the the number of the intermediate vertices of the Soroush's path and his dog's path, respectively. The next two lines contain the x and y coordinates of the entrance point s and the exit point t ($s \neq t$) in that order. The next n lines present the x and y coordinates of the trees. Let Soroush's path and his dog's path be $s, v_1, \ldots, v_m, t$ and $s, u_1, \ldots, u_k, t$, respectively. At the end, the x and y coordinates of $v_1, \cdots, v_m$ come in m lines in order and then the x and y coordinates of $u_1, \cdots, u_k$ come in k lines in order. Note that the paths may have self-intersections or intersect each other. There is no tree on the Soroush's path or his dog's path including s and t. You can assume all input coordinates are integers whose absolute values are at most 10^6. The input terminates with a line containing "`0 0 0`" which should not be processed.

Output (Standard Output)

For each test case, output a line containing "`Yes`" if the two given paths are homotopic. Output "`No`", otherwise.

Sample Input and Output

Standard Input	Standard Output
2 4 8 0 2 7 2 2 1 5 1 6 2 6 0 1 0 1 3 3 2 3 0 1 0 1 3 6 3 6 0 4 0 4 2 1 2 0 0 0 10 0 3 1 5 1 1 0 0 0 0	No Yes

Problem : Image Processing Project

You want to be recruited in Cafebazaar, a leading IT company in Iran, as part of an image processing project for detecting celestial bodies. To prove yourself, you should do a pilot project in a matter of hours for the project manager.

The project manager has put some paper sheets on a rectangular table whose color is white. All paper sheets are square and have the same size. Moreover, all paper sheets are white but their boundaries are black. Each side of all paper sheets is parallel to an edge of the table. You must write an image processing program to count the number of paper sheets by receiving a picture of the table taken from above.

Input

The first line of the input contains two positive integers r and c ($3 \leqslant r, c \leqslant 200$) denoting the number of rows and columns of the given picture, respectively. Precisely, the picture is an $r \times c$ table of pixels. The next r lines, each contains exactly c characters. Each character is `+`, `-`, or `.` representing a pixel of the picture. A dash pixel illustrates one unit of a horizontal side of a paper sheets. A plus pixel illustrates either one unit of a vertical side of a paper sheet or a corner of a paper sheet. A dot pixel is a white pixel.

You can assume that in each row there exists at most one horizontal side. Similarly, in each column, there exists at most one vertical side. It is guaranteed that at least two corners of each paper sheet are visible. A corner is visible when along with the pixel corresponding to the corner, at least one immediate pixel from each of its incident edges is also present in the input. Moreover, It is guaranteed that there is at least a dash pixel in the input.

Output

Print the number of paper sheets in the output.

Example

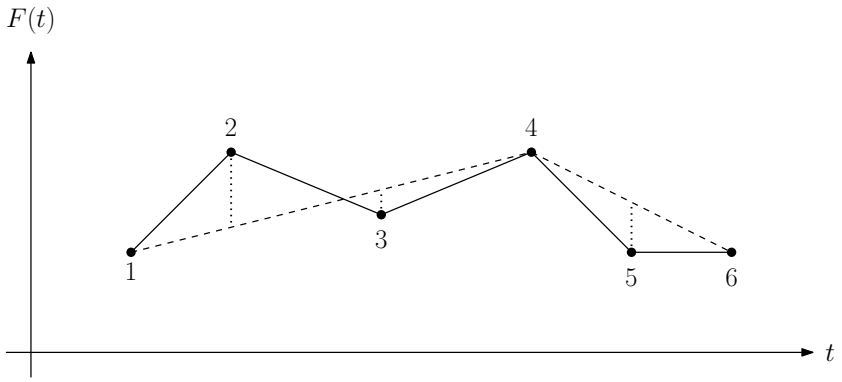
Standard Input	Standard Output
10 13 +---+..... +...+..... +...+...+---+ +...+-+.+...+ +---+.+.+...+ ..+..+--+...+ ..+..+..+---+ ..+--+...+... .....+...+... .....+---+...	4

Problem : Simplification

Amin records the price of his stock every now and then as a data point (t_i, p_i) in his notebook, where p_i is the price of his stock at day t_i. The sequence of these data points represents a piecewise-linear function F displaying the history of prices over a period of time. Indeed, F connects every pair of consecutive points by a straight line segment. If the price is not recorded for some day t, $F(t)$ can be used as an estimate instead.

His collected data is getting larger and larger as he has been tracking the price of his stock over a long period of time. Therefore, he has decided to reduce his data by throwing away some of his recoded data points and constructing a new piecewise-linear function F' with the remaining points. F' is a so-called "simplification" of F. Amin wants to create the simplification in such a way that F' is a good approximation for F. To this end, he has defined an error measure as follows.

Let F be defined over a strictly increasing sequence $L = \langle t_1, \ldots, t_n \rangle$ of days, and F' be defined over a subsequence $L' = \langle t'_1, \ldots, t'_m \rangle$ of L, where $t'_1 = t_1$, $t'_m = t_n$, and $F'(t'_i) = F(t'_i)$ for $1 \leqslant i \leqslant m$. (We call m the size of F'.) The error of F' is defined as the maximum of $|F'(t_k) - F(t_k)|$ for all $1 \leqslant k \leqslant n$. For example, in the following figure, we have 6 data points, labeled 1 through 6, whose coordinates are the same as those presented in the second sample input, and F' is a simplification of F of size 3 with data points 1, 4 and 6. In this figure, F is depicted by solid lines, and F' by dashed lines. The error measure for F' is realized by the vertical distance of point 2 to F'.



Amin's goal is to minimize the size of F', while the error of F' is bounded by a given value δ.

Input

The first line of input contains a positive integer n ($2 \leqslant n \leqslant 2000$) that shows the size of F. Each of the next n lines contains two integers t_i, p_i ($1 \leqslant t_i, p_i \leqslant 10^6$), where p_i is the price of the stock at day t_i. The last line contains the error limit δ which is a non-negative integer at most 10^6.

Output

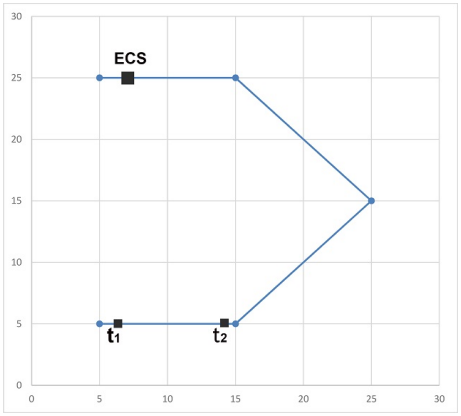
In the output, print the minimum possible size of F'.

Example

Standard Input	Standard Output
3 1 10 3 100 10 20 90	2

Standard Input	Standard Output
6 10 10 20 20 35 14 50 20 60 10 70 10 8	3

Problem : Robot Race



The International competition of Robot Race 2014 will be held in Tehran. In the competition, a path is specified by the scientific committee, and each robot has to move along the path, from the beginning to the end.

There is an electronic charge station (ECS) on the path, at which robots charge their batteries. Every robot have a device which tells its distance to the ECS. Unfortunately, the devices are not good enough, so, each device shows the Euclidean distance of the robot to the ECS, not the remaining distance on the path to the ECS.

Kamran is a member of the scientific committee of the competition. He knows that there is a common bug in the control software of some robots. A buggy robot imagines that its device shows the remaining distance on the path to the ECS, not the the Euclidean distance to the ECS. As a consequence, from the buggy-robot point of view, its device must show a decreasing sequence of numbers before reaching to the ECS. If this is not the case, the buggy robot crashes since it thinks that it has already passed the ECS without getting charged. Kamran considers a given competition path as unfair, if he can choose a position for ECS on the path such that the buggy robots crash in some time. In other words, a path is unfair if an ECS position can be chosen and there exist three times $t_1 < t_2 < t_3$ such that a robot is at ECS at time t_3 and $|p_{t_1}p_{t_3}| < |p_{t_2}p_{t_3}|$ where $|ab|$ denotes the Euclidean distanceb between a and b and p_t is the position of the robot at time t. The scientific committee has proposed a list of possible paths for the competition, and Kamran wants to know which path is fair (i.e, the path is not unfair).

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains a positive integer n ($n \leqslant 10,000$), which is the number of points on the path. The next n lines contain n pair of integers x and y ($-10^6 \leqslant x, y \leqslant 10^6$). The i-th pair specifies the coordinate of the i-point in the path. Robots have to start from the first point, and pass through the segments joining consecutive points each after other, and stop when they reach the last point. It is guaranteed that the path does not intersect itself. The input terminates with a line containing `0` which should not be processed.

Output (Standard Output)

For each test case, output a line containing either `Fair` or `Unfair` depending on whether the given path is fair or unfair, respectively.

Sample Input and Output

Standard Input	Standard Output
5 5 5 15 5 25 15 15 25 5 25 4 0 0 1 0 2 1 3 0 0	Unfair Fair