Problem A
Ceiling Function
Time limit: 4 seconds

Advanced Ceiling Manufacturers (ACM) is analyzing the properties of its new series of Incredibly Collapse-Proof Ceilings (ICPCs). An ICPC consists of n layers of material, each with a different value of collapse resistance (measured as a positive integer). The analysis ACM wants to run will take the collapse-resistance values of the layers, store them in a binary search tree, and check whether the shape of this tree in any way correlates with the quality of the whole construction. Because, well, why should it not?

To be precise, ACM takes the collapse-resistance values for the layers, ordered from the top layer to the bottom layer, and inserts them one-by-one into a tree. The rules for inserting a value v are:

- If the tree is empty, make v the root of the tree.
- If the tree is not empty, compare v with the root of the tree. If v is smaller, insert v into the left subtree of the root, otherwise insert v into the right subtree.

ACM has a set of ceiling prototypes it wants to analyze by trying to collapse them. It wants to take each group of ceiling prototypes that have trees of the same shape and analyze them together.

For example, assume ACM is considering five ceiling prototypes with three layers each, as described by Sample Input 1 and shown in Figure A.1. Notice that the first prototype's top layer has collapse-resistance value 2, the middle layer has value 7, and the bottom layer has value 1. The second prototype has layers with collapse-resistance values of 3, 1, and 4 – and yet these two prototypes induce the same tree shape, so ACM will analyze them together.

Given a set of prototypes, your task is to determine how many different tree shapes they induce.

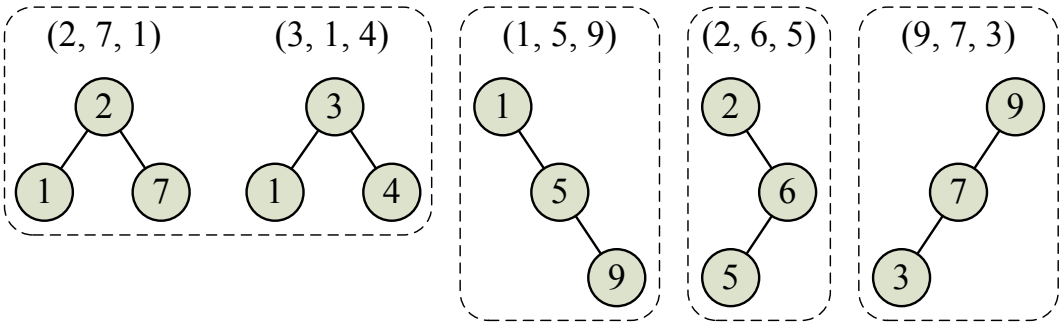


Figure A.1: The four tree shapes induced by the ceiling prototypes in Sample Input 1.

Input

The first line of the input contains two integers n ($1 \leq n \leq 50$), which is the number of ceiling prototypes to analyze, and k ($1 \leq k \leq 20$), which is the number of layers in each of the prototypes.

The next n lines describe the ceiling prototypes. Each of these lines contains k distinct integers (between 1 and 10^6, inclusive), which are the collapse-resistance values of the layers in a ceiling prototype, ordered from top to bottom.

Output

Display the number of different tree shapes.

Sample Input 1	Sample Output 1
5 3 2 7 1 3 1 4 1 5 9 2 6 5 9 7 3	4

Sample Input 2	Sample Output 2
3 4 3 1 2 40000 3 4 2 1 33 42 17 23	2

Problem B
Cutting Cheese
Time limit: 2 seconds

Of course you have all heard of the International Cheese Processing Company. Their machine for cutting a piece of cheese into slices of exactly the same thickness is a classic. Recently they produced a machine able to cut a spherical cheese (such as Edam) into slices – no, not all of the same thickness, but all of the same weight! But new challenges lie ahead: cutting Swiss cheese.

Picture by Jon Sullivan via Wikimedia Commons

Swiss cheese such as Emmentaler has holes in it, and the holes may have different sizes. A slice with holes contains less cheese and has a lower weight than a slice without holes. So here is the challenge: cut a cheese with holes in it into slices of equal weight.

By smart sonar techniques (the same techniques used to scan unborn babies and oil fields), it is possible to locate the holes in the cheese up to micrometer precision. For the present problem you may assume that the holes are perfect spheres.

Each uncut block has size $100 \times 100 \times 100$ where each dimension is measured in millimeters. Your task is to cut it into s slices of equal weight. The slices will be 100 mm wide and 100 mm high, and your job is to determine the thickness of each slice.

Input

The first line of the input contains two integers n and s, where $0 \leq n \leq 10\,000$ is the number of holes in the cheese, and $1 \leq s \leq 100$ is the number of slices to cut. The next n lines each contain four positive integers r, x, y, and z that describe a hole, where r is the radius and x, y, and z are the coordinates of the center, all in micrometers.

The cheese block occupies the points (x, y, z) where $0 \leq x, y, z \leq 100\,000$, except for the points that are part of some hole. The cuts are made perpendicular to the z axis.

You may assume that holes do not overlap but may touch, and that the holes are fully contained in the cheese but may touch its boundary.

Output

Display the s slice thicknesses in millimeters, starting from the end of the cheese with $z = 0$. Your output should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
0 4	25.000000000 25.000000000 25.000000000 25.000000000

Sample Input 2	Sample Output 2
2 5 10000 10000 20000 20000 40000 40000 50000 60000	14.611103142 16.269801734 24.092457788 27.002992272 18.023645064

Problem C
Keyboarding
Time limit: 3 seconds

How many keystrokes are necessary to type a text message? You may think that it is equal to the number of characters in the text, but this is correct only if one keystroke generates one character. With pocket-size devices, the possibilities for typing text are often limited. Some devices provide only a few buttons, significantly fewer than the number of letters in the alphabet. For such devices, several strokes may be needed to type a single character. One mechanism to deal with these limitations is a virtual keyboard displayed on a screen, with a cursor that can be moved from key to key to select characters. Four arrow buttons control the movement of the cursor, and when the cursor is positioned over an appropriate key, pressing the fifth button selects the corresponding character and appends it to the end of the text. To terminate the text, the user must navigate to and select the Enter key. This provides users with an arbitrary set of characters and enables them to type text of any length with only five hardware buttons.

In this problem, you are given a virtual keyboard layout and your task is to determine the minimal number of strokes needed to type a given text, where pressing any of the five hardware buttons constitutes a stroke. The keys are arranged in a rectangular grid, such that each virtual key occupies one or more connected unit squares of the grid. The cursor starts in the upper left corner of the keyboard and moves in the four cardinal directions, in such a way that it always skips to the next unit square in that direction that belongs to a different key. If there is no such unit square, the cursor does not move.

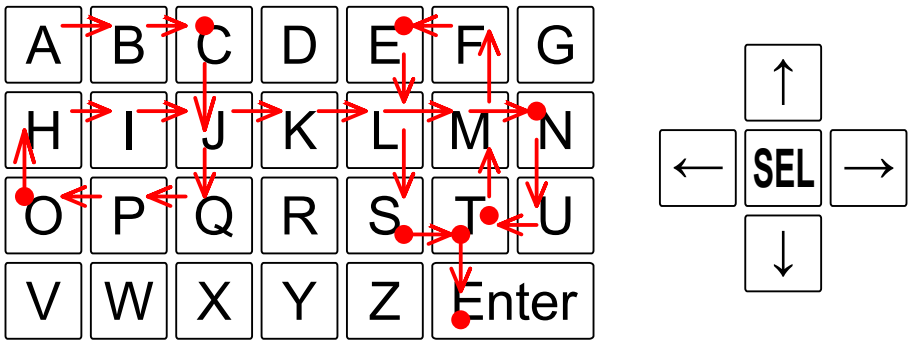


Figure C.1: Sample Input 1. An example virtual keyboard and hardware buttons.

Figure C.1, illustrating Sample Input 1, shows a possible way to type CONTEST using 30 strokes on an example virtual keyboard. The red dots represent the virtual keys where the select button was pressed.

Input

The first line of the input contains two integers r and c ($1 \leq r, c \leq 50$), giving the number of rows and columns of the virtual keyboard grid. The virtual keyboard is specified in the next r lines, each of which contains c characters. The possible values of these characters are uppercase letters, digits, a dash, and an asterisk (representing Enter). There is only one key corresponding to any given character. Each key is made up of one or more grid squares, which will always form a connected region. The last line of the input contains the text to be typed. This text is a non-empty string of at most 10 000 of the available characters other than the asterisk.

Output

Display the minimal number of strokes necessary to type the whole text, including the Enter key at the end. It is guaranteed that the text can be typed.

Sample Input 1	Sample Output 1
4 7 ABCDEFG HIJKLMN OPQRSTU VWXYZ** CONTEST	30

Sample Input 2	Sample Output 2
5 20 12233445566778899000 QQWWEERRTTYYUUIIOOPP -AASSDDFFGGHHJJKKLL* --ZZXXCCVVBBNNMM--** ---------- ACM-ICPC-WORLD-FINALS-2015	160

Sample Input 3	Sample Output 3
2 19 ABCDEFGHIJKLMNOPQZY X**********Y AZAZ	19

Sample Input 4	Sample Output 4
6 4 AXYB BBBB KLMB OPQB DEFB GHI* AB	7

Problem D
Map Tiles
Time limit: 11 seconds

Publishing maps is not an easy task. First you need some appropriate transformation to display the earth's spherical shape in a two-dimensional plane. Then another issue arises – most high-quality maps are too large to be printed on a single page of paper. To cope with that, map publishers often split maps into several rectangular tiles, and print each tile on one page. In this problem, you will examine this "tiling" process.

The International Cartographic Publishing Company (ICPC) needs to cut their printing costs by minimizing the number of tiles used for their maps. Even with a fixed tile size (determined by the page size) and map scale, you can still optimize the situation by adjusting the tile grid.

The left side of Figure D.1 shows 14 map tiles covering a region. The right side shows how you can cover the same region with only 10 tiles, without changing the tile sizes or orientation.

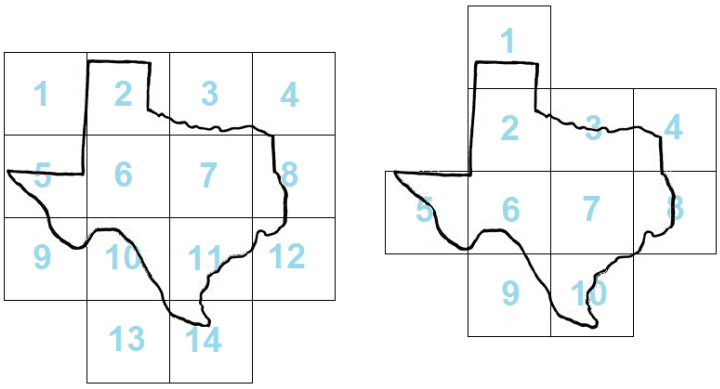


Figure D.1: Two possible ways of tiling Texas.

Your task is to help the ICPC find the minimum number of tiles needed to cover a given region. For simplicity, the region will be given as a closed polygon that does not intersect itself.

Note that the tiles must be part of a rectangular grid aligned with the x-axis and y-axis. That is, they touch each other only with their whole sides and cannot be rotated. Also note that although all input coordinates are integers, tiles may be located at non-integer coordinates.

The polygon may touch the edges of marginal lines (as in Sample Input 2). However, to avoid floating-point issues, you may assume the optimal answer will not change even if the polygon is allowed to go outside the map tiles by a distance of 10^{-6}.

Input

The input consists of a single test case. The first line of a test case contains three integers: n, x_s, and y_s. The number of polygon vertices is n ($3 \leq n \leq 50$), and x_s and y_s ($1 \leq x_s, y_s \leq 100$) are the dimensions of each tile. Each of the next n lines contains two integers x and y ($0 \leq x \leq 10x_s$, $0 \leq y \leq 10y_s$), specifying the vertices of the polygon representing the region (in either clockwise or counter-clockwise order).

Output

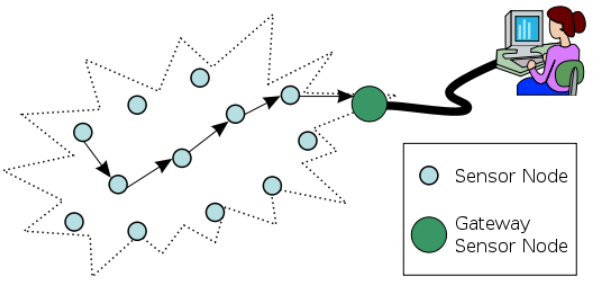
Display the minimal number of tiles necessary to cover the whole interior of the polygon.

Sample Input 1	Sample Output 1
12 9 9 1 8 1 16 6 16 9 29 19 31 23 24 30 23 29 18 20 12 22 8 14 0 14 8	10

Sample Input 2	Sample Output 2
4 5 7 10 10 15 10 15 17 10 17	1

Problem E
Sensor Network
Time limit: 3 seconds

A wireless sensor network consists of autonomous sensors scattered in an environment where they monitor conditions such as temperature, sound, and pressure.



Picture from Wikimedia Commons

Samantha is a researcher working on the Amazon Carbon-dioxide Measurement (ACM) project. In this project, a wireless sensor network in the Amazon rainforest gathers environmental information. The Amazon rainforest stores an amount of carbon equivalent to a decade of global fossil fuel emissions, and it plays a crucial role in the world's oxygen-transfer processes. Because of the huge size of this forest, changes in the forest affect not only the local environment but also global climate by altering wind and ocean current patterns. The goal of the ACM project is to help scientists better understand earth's complex ecosystems and the impact of human activities.

Samantha has an important hypothesis and to test her hypothesis, she needs to find a subset of sensors in which each pair of sensors can communicate directly with each other. A sensor can communicate directly with any other sensor having distance at most d from it. In order for her experiments to be as accurate as possible, Samantha wants to choose as many sensors as possible.

As one does not simply walk into the Amazon, Samantha cannot add new sensors or move those that are currently in place. So given the current locations of the sensors, she needs your help to find the largest subset satisfying her criteria. For simplicity, represent the location of each sensor as a point in a two-dimensional plane with the distance between two points being the usual Euclidean distance.

Input

The input consists of a single test case. The first line contains two integers n and d ($1 \leq n \leq 100$ and $1 \leq d \leq 10\,000$), where n is the number of sensors available and d is the maximum distance between sensors that can communicate directly. Sensors are numbered 1 to n. Each of the next n lines contains two integers x and y ($-10\,000 \leq x, y \leq 10\,000$) indicating the sensor coordinates, starting with the first sensor.

Output

Display a maximum subset of sensors in which each pair of sensors can communicate directly. The first line of output should be the size of the subset. The second line of output should be the (one-based) indices of the sensors in the subset. If there are multiple such subsets, any one of them will be accepted.

Sample Input 1	Sample Output 1
4 1 0 0 0 1 1 0 1 1	2 1 2

Sample Input 2	Sample Output 2
5 20 0 0 0 2 100 100 100 110 100 120	3 4 3 5

Problem F
Weather Report
Time limit: 1 second

You have been hired by the Association for Climatological Measurement, a scientific organization interested in tracking global weather trends over a long period of time. Of course, this is no easy task. They have deployed many small devices around the world, designed to take periodic measurements of the local weather conditions. These are cheap devices with somewhat restricted capabilities. Every day they observe which of the four standard kinds of weather occurred: *Sunny*, *Cloudy*, *Rainy*, or *Frogs*. After every n of these observations have been made, the results are reported to the main server for analysis. However, the massive number of devices has caused the available communication bandwidth to be overloaded. The Association needs your help to come up with a method of compressing these reports into fewer bits.

For a particular device's location, you may assume that the weather each day is an independent random event, and you are given the predicted probabilities of the four possible weather types. Each of the 4^n possible weather reports for a device must be encoded as a unique sequence of bits, such that no sequence is a prefix of any other sequence (an important property, or else the server would not know when each sequence ends). The goal is to use an encoding that minimizes the expected number of transmitted bits.

Input

The first line of input contains an integer $1 \leq n \leq 20$, the number of observations that go into each report. The second line contains four positive floating-point numbers, p_{sunny}, p_{cloudy}, p_{rainy}, and p_{frogs}, representing the respective weather probabilities. These probabilities have at most 6 digits after the decimal point and sum to 1.

Output

Display the minimum expected number of bits in the encoding of a report, with an absolute or relative error of at most 10^{-4}.

Sample Input 1	Sample Output 1
2 0.9 0.049999 0.05 0.000001	1.457510

Sample Input 2	Sample Output 2
20 0.25 0.25 0.25 0.25	40.000000

This page is intentionally left blank.

Problem G
Ceiling Function 2
Time limit: 4 seconds

Advanced Ceiling Manufacturers (ACM) is analyzing the properties of its new series of Incredibly Collapse-Proof Ceilings (ICPCs). An ICPC consists of n layers of material, each with a different value of collapse resistance (measured as a positive integer). The analysis ACM wants to run will take the collapse-resistance values of the layers, store them in a binary search tree, and check whether the shape of this tree in any way correlates with the quality of the whole construction. Because, well, why should it not?

To be precise, ACM takes the collapse-resistance values for the layers, ordered from the top layer to the bottom layer, and inserts them one-by-one into a tree. The rules for inserting a value v are:

- If the tree is empty, make v the root of the tree.
- If the tree is not empty, compare v with the root of the tree. If v is smaller, insert v into the left subtree of the root, otherwise insert v into the right subtree.

ACM has a set of ceiling prototypes it wants to analyze by trying to collapse them. It wants to take each group of ceiling prototypes that have trees of the same shape and analyze them together.

For example, assume ACM is considering five ceiling prototypes with three layers each, as described by Sample Input 1 and shown in Figure G.1. Notice that the first prototype's top layer has collapse-resistance value 2, the middle layer has value 7, and the bottom layer has value 1. The second prototype has layers with collapse-resistance values of 3, 1, and 4 – and yet these two prototypes induce the same tree shape, so ACM will analyze them together.

Given a set of prototypes, your task is to determine how many different tree shapes they induce.

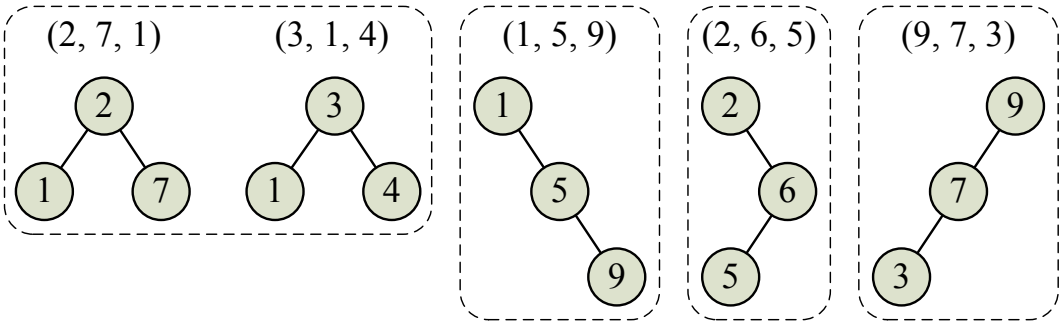


Figure G.1: The four tree shapes induced by the ceiling prototypes in Sample Input 1.

Input

The first line of the input contains two integers n ($1 \leq n \leq 50$), which is the number of ceiling prototypes to analyze, and k ($1 \leq k \leq 20$), which is the number of layers in each of the prototypes.

The next n lines describe the ceiling prototypes. Each of these lines contains k distinct integers (between 1 and 10^6, inclusive), which are the collapse-resistance values of the layers in a ceiling prototype, ordered from top to bottom.

Output

Display the number of different tree shapes.

Sample Input 1	Sample Output 1
5 3 2 7 1 3 1 4 1 5 9 2 6 5 9 7 3	4

Sample Input 2	Sample Output 2
3 4 3 1 2 40000 3 4 2 1 33 42 17 23	2

Problem H
Cutting Cheese 2
Time limit: 2 seconds

Of course you have all heard of the International Cheese Processing Company. Their machine for cutting a piece of cheese into slices of exactly the same thickness is a classic. Recently they produced a machine able to cut a spherical cheese (such as Edam) into slices – no, not all of the same thickness, but all of the same weight! But new challenges lie ahead: cutting Swiss cheese.

Picture by Jon Sullivan via Wikimedia Commons

Swiss cheese such as Emmentaler has holes in it, and the holes may have different sizes. A slice with holes contains less cheese and has a lower weight than a slice without holes. So here is the challenge: cut a cheese with holes in it into slices of equal weight.

By smart sonar techniques (the same techniques used to scan unborn babies and oil fields), it is possible to locate the holes in the cheese up to micrometer precision. For the present problem you may assume that the holes are perfect spheres.

Each uncut block has size $100 \times 100 \times 100$ where each dimension is measured in millimeters. Your task is to cut it into s slices of equal weight. The slices will be 100 mm wide and 100 mm high, and your job is to determine the thickness of each slice.

Input

The first line of the input contains two integers n and s, where $0 \leq n \leq 10\,000$ is the number of holes in the cheese, and $1 \leq s \leq 100$ is the number of slices to cut. The next n lines each contain four positive integers r, x, y, and z that describe a hole, where r is the radius and x, y, and z are the coordinates of the center, all in micrometers.

The cheese block occupies the points (x, y, z) where $0 \leq x, y, z \leq 100\,000$, except for the points that are part of some hole. The cuts are made perpendicular to the z axis.

You may assume that holes do not overlap but may touch, and that the holes are fully contained in the cheese but may touch its boundary.

Output

Display the s slice thicknesses in millimeters, starting from the end of the cheese with $z = 0$. Your output should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
0 4	25.000000000 25.000000000 25.000000000 25.000000000

Sample Input 2	Sample Output 2
2 5 10000 10000 20000 20000 40000 40000 50000 60000	14.611103142 16.269801734 24.092457788 27.002992272 18.023645064

Problem I
Keyboarding 2
Time limit: 3 seconds

How many keystrokes are necessary to type a text message? You may think that it is equal to the number of characters in the text, but this is correct only if one keystroke generates one character. With pocket-size devices, the possibilities for typing text are often limited. Some devices provide only a few buttons, significantly fewer than the number of letters in the alphabet. For such devices, several strokes may be needed to type a single character. One mechanism to deal with these limitations is a virtual keyboard displayed on a screen, with a cursor that can be moved from key to key to select characters. Four arrow buttons control the movement of the cursor, and when the cursor is positioned over an appropriate key, pressing the fifth button selects the corresponding character and appends it to the end of the text. To terminate the text, the user must navigate to and select the Enter key. This provides users with an arbitrary set of characters and enables them to type text of any length with only five hardware buttons.

In this problem, you are given a virtual keyboard layout and your task is to determine the minimal number of strokes needed to type a given text, where pressing any of the five hardware buttons constitutes a stroke. The keys are arranged in a rectangular grid, such that each virtual key occupies one or more connected unit squares of the grid. The cursor starts in the upper left corner of the keyboard and moves in the four cardinal directions, in such a way that it always skips to the next unit square in that direction that belongs to a different key. If there is no such unit square, the cursor does not move.

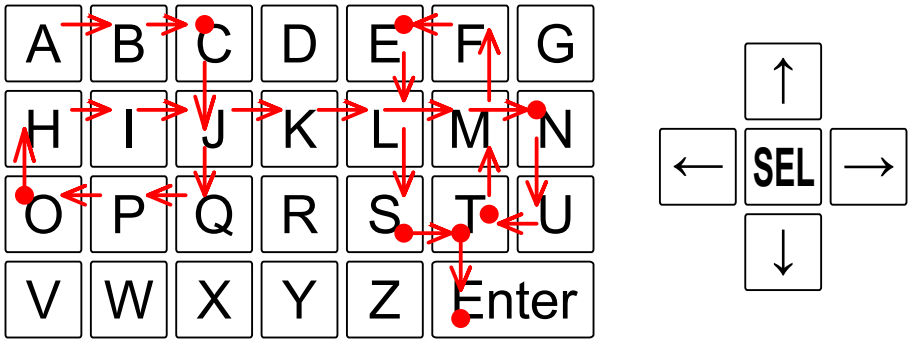


Figure I.1: Sample Input 1. An example virtual keyboard and hardware buttons.

Figure I.1, illustrating Sample Input 1, shows a possible way to type CONTEST using 30 strokes on an example virtual keyboard. The red dots represent the virtual keys where the select button was pressed.

Input

The first line of the input contains two integers r and c ($1 \leq r, c \leq 50$), giving the number of rows and columns of the virtual keyboard grid. The virtual keyboard is specified in the next r lines, each of which contains c characters. The possible values of these characters are uppercase letters, digits, a dash, and an asterisk (representing Enter). There is only one key corresponding to any given character. Each key is made up of one or more grid squares, which will always form a connected region. The last line of the input contains the text to be typed. This text is a non-empty string of at most 10 000 of the available characters other than the asterisk.

Output

Display the minimal number of strokes necessary to type the whole text, including the Enter key at the end. It is guaranteed that the text can be typed.

Sample Input 1	Sample Output 1
4 7 ABCDEFG HIJKLMN OPQRSTU VWXYZ** CONTEST	30

Sample Input 2	Sample Output 2
5 20 12233445566778899000 QQWWEERRTTYYUUIIOOPP -AASSDDFFGGHHJJKKLL* --ZZXXCCVVBBNNMM--** ---------- ACM-ICPC-WORLD-FINALS-2015	160

Sample Input 3	Sample Output 3
2 19 ABCDEFGHIJKLMNOPQZY X**********Y AZAZ	19

Sample Input 4	Sample Output 4
6 4 AXYB BBBB KLMB OPQB DEFB GHI* AB	7

Problem J
Map Tiles 2
Time limit: 11 seconds

Publishing maps is not an easy task. First you need some appropriate transformation to display the earth's spherical shape in a two-dimensional plane. Then another issue arises – most high-quality maps are too large to be printed on a single page of paper. To cope with that, map publishers often split maps into several rectangular tiles, and print each tile on one page. In this problem, you will examine this "tiling" process.

The International Cartographic Publishing Company (ICPC) needs to cut their printing costs by minimizing the number of tiles used for their maps. Even with a fixed tile size (determined by the page size) and map scale, you can still optimize the situation by adjusting the tile grid.

The left side of Figure J.1 shows 14 map tiles covering a region. The right side shows how you can cover the same region with only 10 tiles, without changing the tile sizes or orientation.

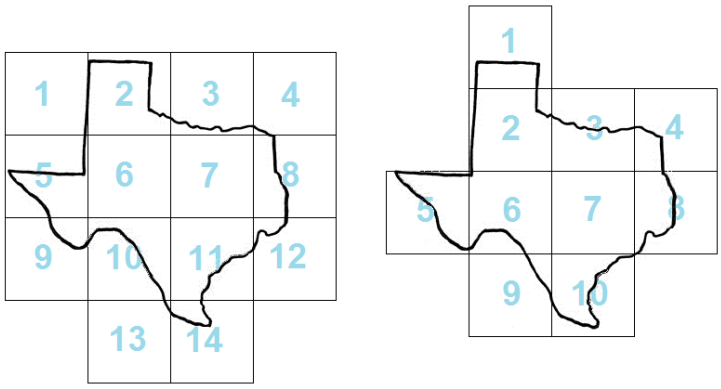


Figure J.1: Two possible ways of tiling Texas.

Your task is to help the ICPC find the minimum number of tiles needed to cover a given region. For simplicity, the region will be given as a closed polygon that does not intersect itself.

Note that the tiles must be part of a rectangular grid aligned with the x-axis and y-axis. That is, they touch each other only with their whole sides and cannot be rotated. Also note that although all input coordinates are integers, tiles may be located at non-integer coordinates.

The polygon may touch the edges of marginal lines (as in Sample Input 2). However, to avoid floating-point issues, you may assume the optimal answer will not change even if the polygon is allowed to go outside the map tiles by a distance of 10^{-6}.

Input

The input consists of a single test case. The first line of a test case contains three integers: n, x_s, and y_s. The number of polygon vertices is n ($3 \leq n \leq 50$), and x_s and y_s ($1 \leq x_s, y_s \leq 100$) are the dimensions of each tile. Each of the next n lines contains two integers x and y ($0 \leq x \leq 10x_s$, $0 \leq y \leq 10y_s$), specifying the vertices of the polygon representing the region (in either clockwise or counter-clockwise order).

Output

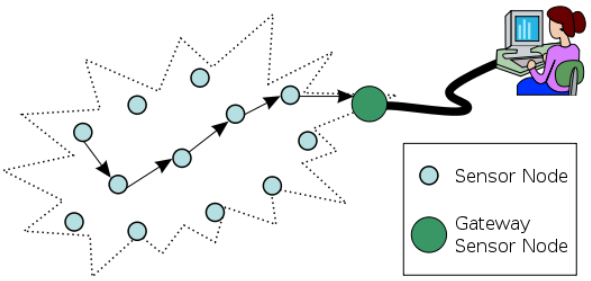
Display the minimal number of tiles necessary to cover the whole interior of the polygon.

Sample Input 1	Sample Output 1
12 9 9 1 8 1 16 6 16 9 29 19 31 23 24 30 23 29 18 20 12 22 8 14 0 14 8	10

Sample Input 2	Sample Output 2
4 5 7 10 10 15 10 15 17 10 17	1

Problem K
Sensor Network 2
Time limit: 3 seconds

A wireless sensor network consists of autonomous sensors scattered in an environment where they monitor conditions such as temperature, sound, and pressure.



Picture from Wikimedia Commons

Samantha is a researcher working on the Amazon Carbon-dioxide Measurement (ACM) project. In this project, a wireless sensor network in the Amazon rainforest gathers environmental information. The Amazon rainforest stores an amount of carbon equivalent to a decade of global fossil fuel emissions, and it plays a crucial role in the world's oxygen-transfer processes. Because of the huge size of this forest, changes in the forest affect not only the local environment but also global climate by altering wind and ocean current patterns. The goal of the ACM project is to help scientists better understand earth's complex ecosystems and the impact of human activities.

Samantha has an important hypothesis and to test her hypothesis, she needs to find a subset of sensors in which each pair of sensors can communicate directly with each other. A sensor can communicate directly with any other sensor having distance at most d from it. In order for her experiments to be as accurate as possible, Samantha wants to choose as many sensors as possible.

As one does not simply walk into the Amazon, Samantha cannot add new sensors or move those that are currently in place. So given the current locations of the sensors, she needs your help to find the largest subset satisfying her criteria. For simplicity, represent the location of each sensor as a point in a two-dimensional plane with the distance between two points being the usual Euclidean distance.

Input

The input consists of a single test case. The first line contains two integers n and d ($1 \leq n \leq 100$ and $1 \leq d \leq 10\,000$), where n is the number of sensors available and d is the maximum distance between sensors that can communicate directly. Sensors are numbered 1 to n. Each of the next n lines contains two integers x and y ($-10\,000 \leq x, y \leq 10\,000$) indicating the sensor coordinates, starting with the first sensor.

Output

Display a maximum subset of sensors in which each pair of sensors can communicate directly. The first line of output should be the size of the subset. The second line of output should be the (one-based) indices of the sensors in the subset. If there are multiple such subsets, any one of them will be accepted.

Sample Input 1

```
4 1
0 0
0 1
1 0
1 1
```

Sample Output 1

```
2
1 2
```

Sample Input 2

```
5 20
0 0
0 2
100 100
100 110
100 120
```

Sample Output 2

```
3
4 3 5
```

Problem L
Weather Report 2
Time limit: 1 second

You have been hired by the Association for Climatological Measurement, a scientific organization interested in tracking global weather trends over a long period of time. Of course, this is no easy task. They have deployed many small devices around the world, designed to take periodic measurements of the local weather conditions. These are cheap devices with somewhat restricted capabilities. Every day they observe which of the four standard kinds of weather occurred: *Sunny*, *Cloudy*, *Rainy*, or *Frogs*. After every n of these observations have been made, the results are reported to the main server for analysis. However, the massive number of devices has caused the available communication bandwidth to be overloaded. The Association needs your help to come up with a method of compressing these reports into fewer bits.

For a particular device's location, you may assume that the weather each day is an independent random event, and you are given the predicted probabilities of the four possible weather types. Each of the 4^n possible weather reports for a device must be encoded as a unique sequence of bits, such that no sequence is a prefix of any other sequence (an important property, or else the server would not know when each sequence ends). The goal is to use an encoding that minimizes the expected number of transmitted bits.

Input

The first line of input contains an integer $1 \leq n \leq 20$, the number of observations that go into each report. The second line contains four positive floating-point numbers, p_{sunny}, p_{cloudy}, p_{rainy}, and p_{frogs}, representing the respective weather probabilities. These probabilities have at most 6 digits after the decimal point and sum to 1.

Output

Display the minimum expected number of bits in the encoding of a report, with an absolute or relative error of at most 10^{-4}.

Sample Input 1	Sample Output 1
2 0.9 0.049999 0.05 0.000001	1.457510

Sample Input 2	Sample Output 2
20 0.25 0.25 0.25 0.25	40.000000

This page is intentionally left blank.