Problem A. Grey Area

Input file: `a.in`
Output file: `standard output`

Dr. Grey is a data analyst, who visualizes various aspects of data received from all over the world everyday. He is extremely good at sophisticated visualization tools, but yet his favorite is a simple self-made histogram generator.

Figure 1 is an example of histogram automatically produced by his histogram generator. A histogram is a visual display of frequencies of value occurrences as bars. In this example, values in the interval 0-9 occur five times, those in the interval 10-19 occur three times, and 20-29 and 30-39 once each.

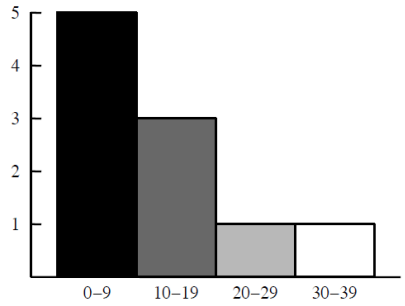

Figure 1: A histogram

Dr. Grey's histogram generator is a simple tool. First, the height of the histogram is fixed, that is, the height of the highest bar is always the same and those of the others are automatically adjusted proportionately. Second, the widths of bars are also fixed. It can only produce a histogram of uniform intervals, that is, each interval of a histogram should have the same width (10 in the above example). Finally, the bar for each interval is painted in a grey color, where the colors of the leftmost and the rightmost intervals are black and white, respectively, and the darkness of bars monotonically decreases at the same rate from left to right. For instance, in Figure 1, the darkness levels of the four bars are 1, 2/3, 1/3, and 0, respectively. In this problem, you are requested to estimate ink consumption when printing a histogram on paper. The amount of ink necessary to draw a bar is proportional to both its area and darkness.

Input

The input consists of multiple datasets, each of which contains integers and specifies a value table and intervals for the histogram generator, in the following format.

n w

v_1

v_2

...

v_n

n is the total number of value occurrences for the histogram, and each of the n lines following the first line contains a single value. Note that the same value may possibly occur multiple times.

w is the interval width. A value v is in the first (i.e. leftmost) interval if $0 \leq v < w$, the second one if $w \leq v < 2w$, and so on. Note that the interval from 0 (inclusive) to w (exclusive) should be regarded as the leftmost even if no values occur in this interval. The last (i.e. rightmost) interval is the one that includes the largest value in the dataset.

You may assume the following.

$1 \leq n \leq 100$

$10 \leq w \leq 50$

$0 \leq v_i \leq 100$ for $1 \leq i \leq n$

You can also assume that the maximum value is no less than w. This means that the histogram has more than one interval.

The end of the input is indicated by a line containing two zeros.

Output

For each dataset, output a line containing the amount of ink consumed in printing the histogram. One unit of ink is necessary to paint one highest bar black. Assume that 0.01 units of ink per histogram is consumed for various purposes except for painting bars such as drawing lines and characters (see Figure 1). For instance, the amount of ink consumed in printing the histogram in Figure 1 is:

$$1 \times 1 + \frac{2}{3} \times \frac{3}{5} + \frac{1}{3} \times \frac{1}{5} + 0 \times \frac{1}{5} + 0.01 = 1 + \frac{2}{5} + \frac{1}{15} + 0.01 = 1.476666$$

Each output value should be in a decimal fraction and may have an error less than 10^{-5}.

Example

a.in	standard output
3 50 100 0 100 3 50 100 100 50 10 10 1 2 3 4 5 16 17 18 29 30 0 0	0.51 0.26 1.4766666666666667

Problem B. Expected Allowance

Input file: `b.in`
Output file: `standard output`

Hideyuki is allowed by his father Ujisato some 1000 yen bills every month for his pocket money. In the first day of every month, the number of bills is decided as follows. Ujisato prepares n pieces of m-sided dice and declares the cutback k. Hideyuki rolls these dice. The number of bills given is the sum of the spots of the rolled dice decreased by the cutback. Fortunately to Hideyuki, Ujisato promises him to give at least one bill, even if the sum of the spots does not exceed the cutback. Each of the dice has spots of 1 through m inclusive on each side, and the probability of each side is the same.

In this problem, you are asked to write a program that finds the expected value of the number of given bills.

Input

The input is a sequence of lines each of which contains three integers n, m and k in this order. They satisfy the following conditions.

$1 \leq n$

$2 \leq m$

$0 \leq k < nm$

$nm \leq m^n < 100000000$ (10^8)

The end of the input is indicated by a line containing three zeros.

Output

The output should be comprised of lines each of which contains a single decimal fraction. It is the expected number of bills and may have an error less than 10^{-7}. No other characters should occur in the output.

Example

b.in	standard output
2 6 0 2 6 3 3 10 9 13 3 27 1 2008 3 0 0 0	7.00000000 4.11111111 7.71000000 1.42902599 1001.50298805

Problem C. Stopped Watches

Input file: c.in
Output file: standard output

In the middle of Tyrrhenian Sea, there is a small volcanic island called Chronus. The island is now uninhabited but it used to be a civilized island. Some historical records imply that the island was annihilated by an eruption of a volcano about 800 years ago and that most of the people in the island were killed by pyroclastic flows caused by the volcanic activity. In 2003, a European team of archaeologists launched an excavation project in Chronus Island. Since then, the project has provided many significant historic insights. In particular the discovery made in the summer of 2008 astonished the world: the project team excavated several mechanical watches worn by the victims of the disaster. This indicates that people in Chronus Island had such a highly advanced manufacturing technology.

Shortly after the excavation of the watches, archaeologists in the team tried to identify what time of the day the disaster happened, but it was not successful due to several difficulties. First, the extraordinary heat of pyroclastic flows severely damaged the watches and took away the letters and numbers printed on them. Second, every watch has a perfect round form and one cannot tell where the top of the watch is. Lastly, though every watch has three hands, they have a completely identical look and therefore one cannot tell which is the hour, the minute, or the second[1]. This means that we cannot decide the time indicated by a watch uniquely; there can be a number of candidates. We have to consider different rotations of the watch. Furthermore, since there are several possible interpretations of hands, we have also to consider all the permutations of hands.

You are an information archaeologist invited to the project team and are asked to induce the most plausible time interval within which the disaster happened, from the set of excavated watches. In what follows, we express a time modulo 12 hours. We write a time by the notation hh:mm:ss, where hh, mm, and ss stand for the hour (hh = 00, 01, 02, ... , 11), the minute (mm = 00, 01, 02, ... , 59), and the second (ss = 00, 01, 02, ... , 59), respectively. The time starts from 00:00:00 and counts up every second 00:00:00, 00:00:01, 00:00:02, ... , but it reverts to 00:00:00 every 12 hours.

The watches in Chronus Island obey the following conventions of modern analog watches.

- A watch has three hands, i.e. the hour hand, the minute hand, and the second hand, though they look identical as mentioned above.
- Every hand ticks 6 degrees clockwise in a discrete manner. That is, no hand stays between ticks, and each hand returns to the same position every 60 ticks.
- The second hand ticks every second.
- The minute hand ticks every 60 seconds.
- The hour hand ticks every 12 minutes.

At the time 00:00:00, all the three hands are located at the same position.

Because people in Chronus Island were reasonably keen to keep their watches correct and pyroclastic flows spread over the island quite rapidly, it can be assumed that all the watches were stopped in a short interval of time. Therefore it is highly expected that the time the disaster happened is in the shortest time interval within which all the excavated watches have at least one candidate time.

You must calculate the shortest time interval and report it to the project team.

[1]It is a mystery how the people in Chronus Island were distinguishing the three hands. Some archaeologists guess that the hands might be painted with different colors, but this is only a hypothesis, as the paint was lost by the heat.

Input

The input consists of multiple datasets, each of which is formatted as follows.

n

$s_1\ t_1\ u_1$

$s_2\ t_2\ u_2$

...

$s_n\ t_n\ u_n$

The first line contains a single integer n ($2 \leq n \leq 10$), representing the number of the watches. The three numbers s_i, t_i, u_i in each line are integers such that 0 . si; ti; ui . 59 and they specify the positions of the three hands by the number of ticks relative to an arbitrarily chosen position. Note that the positions of the hands of a watch can be expressed in many different ways. For example, if a watch was stopped at the time 11:55:03, the positions of hands can be expressed differently by rotating the watch arbitrarily (e.g. 59 55 3, 0 56 4, 1 57 5, etc.) and as well by permuting the hour, minute, and second hands arbitrarily (e.g. 55 59 3, 55 3 59, 3 55 59, etc.). The end of the input is indicated by a line containing a single zero.

Output

For each dataset, output the shortest time interval within which all the watches given in the dataset have at least one candidate time. The output must be written in a single line in the following format for each dataset.

hh:mm:ss h'h':m'm':s's'

Each line contains a pair of times hh:mm:ss and h'h':m'm':s's', indicating that the shortest interval begins at hh:mm:ss and ends at h'h':m'm':s's' inclusive. The beginning time and the ending time are separated by a single space and each of them should consist of hour, minute, and second in two digits separated by colons. No extra characters should appear in the output.

In calculating the shortest interval, you can exploit the facts that every watch has at least one candidate time and that the shortest time interval contains 00:00:00 only if the interval starts from 00:00:00 (i.e. the shortest interval terminates before the time reverts to 00:00:00).

If there is more than one time interval that gives the shortest, output the one that first comes after 00:00:00 inclusive.

Example

c.in	standard output
3 8 8 18 32 32 32 57 2 57 5 49 3 49 7 30 44 27 21 21 33 56 56 21 46 4 3 45 52 28 36 26 36 20 55 50 10 33 8 39 50 57 43 35 21 12 21 17 11 16 21 58 45 40 53 45 30 53 39 1 8 55 48 30 7 48 15 0	00:00:00 00:00:10 06:14:56 06:32:09 07:27:37 07:32:02 05:17:40 05:21:03

Problem D. Digits on the Floor

Input file: `d.in`
Output file: `standard output`

Taro attempts to tell digits to Hanako by putting straight bars on the floor. Taro wants to express each digit by making one of the forms shown in Figure 2. Since Taro may not have bars of desired lengths, Taro cannot always make forms exactly as shown in Figure 2. Fortunately, Hanako can recognize a form as a digit if the connection relation between bars in the form is kept. Neither the lengths of bars nor the directions of forms affect Hanako's perception as long as the connection relation remains the same. For example, Hanako can recognize all the awkward forms in Figure 3 as digits. On the other hand, Hanako cannot recognize the forms in Figure 4 as digits. For clarity, touching bars are slightly separated in Figures 2, 3 and 4. Actually, touching bars overlap exactly at one single point.

Figure 2: Representation of digits

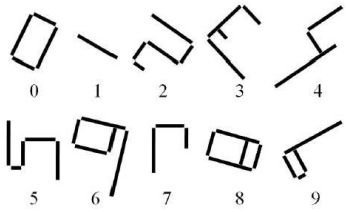

Figure 3: Examples of forms recognized as digits

In the forms, when a bar touches another, the touching point is an end of at least one of them.

That is, bars never cross. In addition, the angle of such two bars is always a right angle.

To enable Taro to represent forms with his limited set of bars, positions and lengths of bars can be changed as far as the connection relations are kept. Also, forms can be rotated.

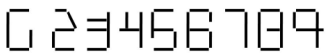

Figure 4: Forms not recognized as digits (these kinds of forms are not contained in the dataset)

Keeping the connection relations means the following.

- Separated bars are not made to touch.
- Touching bars are not made separate.
- When one end of a bar touches another bar, that end still touches the same bar. When it touches a midpoint of the other bar, it remains to touch a midpoint of the same bar on the same side.
- The angle of touching two bars is kept to be the same right angle (90 degrees and -90 degrees are considered different, and forms for 2 and 5 are kept distinguished).

Your task is to find how many times each digit appears on the floor. The forms of some digits always contain the forms of other digits. For example, a form for 9 always contains four forms for 1, one form for 4, and two overlapping forms for 7. In this problem, ignore the forms contained in another form and count only the digit of the "largest" form composed of all mutually connecting bars. If there is one form for 9, it should be interpreted as one appearance of 9 and no appearance of 1, 4, or 7.

Input

The input consists of a number of datasets. Each dataset is formatted as follows.

n

$x_{1a}\ y_{1a}\ x_{1b}\ y_{1b}$

$x_{2a}\ y_{2a}\ x_{2b}\ y_{2b}$

...

$x_{na}\ y_{na}\ x_{nb}\ y_{nb}$

In the first line, n represents the number of bars in the dataset. For the rest of the lines, one line represents one bar. Four integers x_a, y_a, x_b, y_b, delimited by single spaces, are given in each line. x_a and y_a are the x- and y-coordinates of one end of the bar, respectively. x_b and y_b are those of the other end. The coordinate system is as shown in Figure 5. You can assume $1 \leq n \leq 1000$ and $0 \leq x_a, y_a, x_b, y_b \leq 1000$.

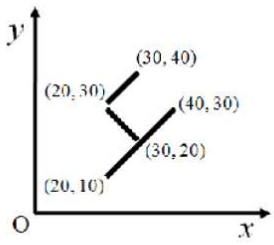

Figure 5: The coordinate system

The end of the input is indicated by a line containing one zero.

Output

For each dataset, output a single line containing ten integers delimited by single spaces. These integers represent how many times 0, 1, 2, . . . , and 9 appear on the floor in this order. Output lines must not contain other characters.

Example

d.in	standard output
9 60 140 200 300 300 105 330 135 330 135 250 215 240 205 250 215 298 167 285 154 30 40 30 90 30 90 150 90 150 90 150 20 30 40 150 40 8 320 20 300 60 320 20 380 50 380 50 240 330 10 50 40 20 10 50 110 150 110 150 180 80 40 20 37 17 37 17 27 27 20 72 222 132 182 204 154 204 54 510 410 520 370 404 54 204 54 530 450 410 450 204 68 404 68 80 110 120 30 130 160 180 60 520 370 320 320 310 360 320 320 120 30 180 60 60 100 80 110 404 154 204 154 80 60 60 100 430 550 590 550 510 410 310 360 430 450 430 550 404 54 404 154 232 202 142 262 142 262 102 202 0	0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 0 2 0 0 0 1 0 12

You can also assume the following conditions.

- More than two bars do not overlap at one point.
- Every bar is used as a part of a digit. Non-digit forms do not exist on the floor.
- A bar that makes up one digit does not touch nor cross any bar that makes up another digit.
- There is no bar whose length is zero.

Problem E. Spherical Mirrors

Input file: `e.in`
Output file: `standard output`

A long time ago in a galaxy, far, far away, there were N spheres with various radii. Spheres were mirrors, that is, they had re ective surfaces ....

You are standing at the origin of the galaxy $(0, 0, 0)$, and emit a laser ray to the direction (u, v, w). The ray travels in a straight line.

When the laser ray from I hits the surface of a sphere at Q, let N be a point outside of the sphere on the line connecting the sphere center and Q. The reflected ray goes to the direction towards R that satisfies the following conditions: (1) R is on the plane formed by the three points I, Q and N, (2) $\angle IQN = \angle NQR$.

After it is reflected several times, finally it goes beyond our observation. Your mission is to write a program that identifies the last reflection point.

Input

The input consists of multiple datasets, each in the following format.

N

$u\ v\ w$

$x_1\ y_1\ z_1\ r_1$

...

$x_N\ y_N\ z_N\ r_N$

The first line of a dataset contains a positive integer N which is the number of spheres. The next line contains three integers u, v and w separated by single spaces, where (u, v, w) is the direction of the laser ray initially emitted from the origin.

Each of the following N lines contains four integers separated by single spaces. The i-th line corresponds to the i-th sphere, and the numbers represent the center position (x_i, y_i, z_i) and the radius r_i.

N, u, v, w, x_i, y_i, z_i and r_i satisfy the following conditions.

$1 \leq N \leq 100$

$-100 \leq u, v, w \leq 100$

$-100 \leq x_i, y_i, z_i \leq 100$

$5 \leq r_i \leq 30$

$u^2 + v^2 + w^2 > 0$

You can assume that the distance between the surfaces of any two spheres is no less than 0.1.

You can also assume that the origin $(0, 0, 0)$ is located outside of any sphere, and is at least 0.1 distant from the surface of any sphere.

The ray is known to be reflected by the sphere surfaces at least once, and at most five times.

You can assume that the angle between the ray and the line connecting the sphere center and the reflection point, which is known as the angle of reflection, is less than 85 degrees for each point of reflection.

The last dataset is followed by a line containing a single zero.

Output

For each dataset in the input, you should print the x-, y- and z-coordinates of the last reflection point separated by single spaces in a line. No output line should contain extra characters. No coordinate values in the output should have an error greater than 0.01.

Example

e.in	standard output
3 -20 -20 -24 100 100 100 30 10 8 3 5 -70 -70 -84 5 4 0 47 84 -23 41 42 8 45 -10 14 19 -5 28 47 12 -27 68 34 14 0	79.0940 79.0940 94.9128 -21.8647 54.9770 34.1761

Problem F. Traveling Cube

Input file: `f.in`
Output file: `standard output`

On a small planet named Bandai, a landing party of the starship Tadamigawa discovered colorful cubes traveling on flat areas of the planet surface, which the landing party named beds. A cube appears at a certain position on a bed, travels on the bed for a while, and then disappears. After a longtime observation, a science officer Lt. Alyssa Ogawa of Tadamigawa found the rule how a cube travels on a bed.

A bed is a rectangular area tiled with squares of the same size.

- One of the squares is colored red,
- one colored green,
- one colored blue,
- one colored cyan,
- one colored magenta,
- one colored yellow,
- one or more colored white, and
- all others, if any, colored black.

Initially, a cube appears on one of the white squares. The cube's faces are colored as follows — top: red, bottom: cyan, north: green, south: magenta, east: blue, west: yellow.

The cube can roll around a side of the current square at a step and thus rolls on to an adjacent square. When the cube rolls on to a chromatically colored (red, green, blue, cyan, magenta or yellow) square, the top face of the cube after the roll should be colored the same. When the cube rolls on to a white square, there is no such restriction. The cube should never roll on to a black square.

Throughout the travel, the cube can visit each of the chromatically colored squares only once, and any of the white squares arbitrarily many times. As already mentioned, the cube can never visit any of the black squares. On visit to the final chromatically colored square, the cube disappears. Somehow the order of visits to the chromatically colored squares is known to us before the travel starts.

Your mission is to find the least number of steps for the cube to visit all the chromatically colored squares in the given order.

Input

The input is a sequence of datasets. A dataset is formatted as follows:

$w\ d$

$$\begin{matrix} c_{11} & \cdots & c_{w1} \\ \vdots & \ddots & \vdots \\ c_{1d} & \cdots & c_{wd} \end{matrix}$$

$v_1v_2v_3v_4v_5v_6$

The first line is a pair of positive integers w and d separated by a space. The next d lines are w-character-long strings $c_{11} \ldots c_{w1}, c_{1d} \ldots c_{wd}$ with no spaces. Each character c_{ij} is one of the letters `r`, `g`, `b`, `c`, `m`, `y`, `w` and `k`, which stands for red, green, blue, cyan, magenta, yellow, white and black respectively, or a sign `#`. Each of `r`, `g`, `b`, `c`, `m`, `y` and `#` occurs once and only once in a dataset. The last line is a six-character-long string $v_1v_2v_3v_4v_5v_6$ which is a permutation of "rgbcmy".

The integers w and d denote the width (the length from the east end to the west end) and the depth (the length from the north end to the south end) of a bed. The unit is the length of a side of a square. You can assume that neither w nor d is greater than 30.

Each character c_{ij} shows the color of a square in the bed. The characters c_{11}, c_{w1}, c_{1d} and c_{wd} correspond to the north-west corner, the north-east corner, the south-west corner and the southeast corner of the bed respectively. If c_{ij} is a letter, it indicates the color of the corresponding square. If c_{ij} is a `#`, the corresponding square is colored white and is the initial position of the cube.

The string $v_1v_2v_3v_4v_5v_6$ shows the order of colors of squares to visit. The cube should visit the squares colored v_1, v_2, v_3, v_4, v_5 and v_6 in this order.

The end of the input is indicated by a line containing two zeros separated by a space.

Output

For each input dataset, output the least number of steps if there is a solution, or "unreachable" if there is no solution. In either case, print it in one line for each input dataset.

Example

f.in	standard output
10 5 kkkkkwwwww w#wwwrwwww wwwwbgwwww kwwmcwwwkk kkwywwwkkk rgbcmy 10 5 kkkkkkkkkk k#kkkkkkkk kwkkkkwwwk kcmyrgbwwk kwwwwwwwwk cmyrgb 10 5 kkkkkkkkkk k#kkkkkkkk kwkkkkwwkk kcmyrgbwwk kwwwwwwwwk cmyrgb 0 0	9 49 unreachable

Problem G. Search of Concatenated Strings

Input file: `g.in`
Output file: `standard output`

The amount of information on the World Wide Web is growing quite rapidly. In this information explosion age, we must survive by accessing only the Web pages containing information relevant to our own needs. One of the key technologies for this purpose is keyword search. By using well-known search engines, we can easily access those pages containing useful information about the topic we want to know.

There are many variations in keyword search problems. If a single string is searched in a given text, the problem is quite easy. If the pattern to be searched consists of multiple strings, or is given by some powerful notation such as regular expressions, the task requires elaborate algorithms to accomplish efficiently.

In our problem, a number of strings (element strings) are given, but they are not directly searched for. Concatenations of all the element strings in any order are the targets of the search here.

For example, consider three element strings aa, b and ccc are given. In this case, the following six concatenated strings are the targets of the search, i.e. they should be searched in the text.

```
aabccc
aacccb
baaccc
bcccaa
cccaab
cccbaa
```

The text may contain several occurrences of these strings. You are requested to count the number of occurrences of these strings, or speaking more precisely, the number of positions of occurrences in the text.

Two or more concatenated strings may be identical. In such cases, it is necessary to consider subtle aspects of the above problem statement. For example, if two element strings are x and xx, the string xxx is an occurrence of both the concatenation of x and xx and that of xx and x. Since the number of positions of occurrences should be counted, this case is counted as one, not two.

Two occurrences may overlap. For example, the string xxxx has occurrences of the concatenation xxx in two different positions. This case is counted as two.

Input

The input consists of a number of datasets, each giving a set of element strings and a text. The format of a dataset is as follows.

n m

e_1

e_2

...

e_n

t_1

t_2

...

t_m

The first line contains two integers separated by a space. n is the number of element strings. m is the number of lines used to represent the text. n is between 1 and 12, inclusive. Each of the following n lines gives an element string. The length (number of characters) of an element string is between 1 and 20, inclusive.

The last m lines as a whole give the text. Since it is not desirable to have a very long line, the text is separated into m lines by newlines, but these newlines should be ignored. They are not parts of the text. The length of each of these lines (not including the newline) is between 1 and 100, inclusive. The length of the text is between 1 and 5000, inclusive.

The element strings and the text do not contain characters other than lowercase letters.

The end of the input is indicated by a line containing two zeros separated by a space.

Output

For each dataset in the input, one line containing the number of matched positions should be output. An output line should not contain extra characters.

Example

g.in
3 1 aa b ccc aabccczbaacccbaazaabbcccaa 3 1 a b c cbbcbcbabaacabccaccbaacbccbcaaaccccbcbcbbcacbaacccaccbbcaacbbabbabaccc 0 0
standard output
5 12

Problem H. Top Spinning

Input file: `h.in`
Output file: `standard output`

Spinning tops are one of the most popular and the most traditional toys. Not only spinning them, but also making one's own is a popular enjoyment.

One of the easiest way to make a top is to cut out a certain shape from a cardboard and pierce an axis stick through its center of mass. Professionally made tops usually have three dimensional shapes, but in this problem we consider only two dimensional ones.

Usually, tops have rotationally symmetric shapes, such as a circle, a rectangle (with 2-fold rotational symmetry) or a regular triangle (with 3-fold symmetry). Although such symmetries are useful in determining their centers of mass, they are not definitely required; an asymmetric top also spins quite well if its axis is properly pierced at the center of mass. When a shape of a top is given as a path to cut it out from a cardboard of uniform thickness, your task is to find its center of mass to make it spin well. Also, you have to determine whether the center of mass is on the part of the cardboard cut out. If not, you cannot pierce the axis stick, of course.

Input

The input consists of multiple datasets, each of which describes a counterclockwise path on a cardboard to cut out a top. A path is indicated by a sequence of command lines, each of which specifies a line segment or an arc.

In the description of commands below, the current position is the position to start the next cut, if any. After executing the cut specified by a command, the current position is moved to the end position of the cut made.

The commands given are one of those listed below. The command name starts from the first column of a line and the command and its arguments are separated by a space. All the command arguments are integers.

`start x y`

Specifies the start position of a path. This command itself does not specify any cutting; it only sets the current position to be (x, y).

`line x y`

Specifies a linear cut along a straight line from the current position to the position (x, y), which is not identical to the current position.

`arc x y r`

Specifies a round cut along a circular arc. The arc starts from the current position and ends at (x, y), which is not identical to the current position. The arc has a radius of $|r|$. When r is negative, the center of the circle is to the left side of the direction of this round cut; when it is positive, it is to the right side (Figure 7). The absolute value of r is greater than the half distance of the two ends of the arc. Among two arcs connecting the start and the end positions with the specified radius, the arc specified is one with its central angle less than 180 degrees.

`close`

Closes a path by making a linear cut to the initial start position and terminates a dataset. If the current position is already at the start position, this command simply indicates the end of a dataset.

The figure below gives an example of a command sequence and its corresponding path. Note that, in this case, the given radius -r is negative and thus the center of the arc is to the left of the arc. The arc command should be interpreted as shown in this figure and, not the other way around on the same circle.

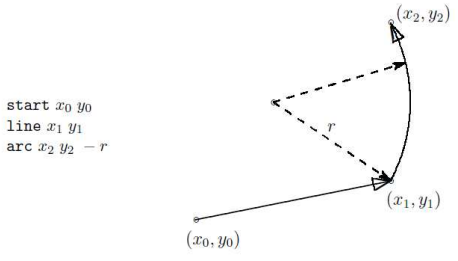

Figure 7: A partial command sequence and the path specified so far

A dataset starts with a start command and ends with a close command. The end of the input is specified by a line with a command end.

There are at most 100 commands in a dataset and at most 100 datasets are in the input.

Absolute values of all the coordinates and radii are less than or equal to 100.

You may assume that the path does not cross nor touch itself. You may also assume that paths will never expand beyond edges of the cardboard, or, in other words, the cardboard is virtually infinitely large.

Output

For each of the dataset, output a line containing x- and y-coordinates of the center of mass of the top cut out by the path specified, and then a character '+' or '-' indicating whether this center is on the top or not, respectively. Two coordinates should be in decimal fractions. There should be a space between two coordinates and between the y-coordinate and the character '+' or '-'. No other characters should be output. The coordinates may have errors less than 10^{-3}. You may assume that the center of mass is at least 10^{-3} distant from the path.

Example

h.in	standard output
start 0 0 arc 2 2 -2 line 2 5 arc 0 3 -2 close start -1 1 line 2 1 line 2 2 line -2 2 arc -3 1 -1 line -3 -2 arc -2 -3 -1 line 2 -3 line 2 -2 line -1 -2 line -1 -1 arc -1 0 2 close start 0 0 line 3 0 line 5 -1 arc 4 -2 -1 line 6 -2 line 6 1 line 7 3 arc 8 2 -1 line 8 4 line 5 4 line 3 5 arc 4 6 -1 line 2 6 line 2 3 line 1 1 arc 0 2 -1 close end	1.00000 2.50000 + -1.01522 -0.50000 - 4.00000 2.00000 +

Problem I. Common Polynomial

Input file: `i.in`
Output file: `standard output`

Math teacher Mr. Masdura is teaching expansion and factoring of polynomials to his students. Last week he instructed the students to write two polynomials (with a single variable x), and to report GCM (greatest common measure) of them as a homework, but he found it boring to check their answers manually. So you are asked to write a program to check the answers. Hereinafter, only those polynomials with integral coefficients, called integral polynomials, are considered.

When two integral polynomials A and B are given, an integral polynomial C is a common factor of A and B if there are some integral polynomials X and Y such that $A = CX$ and $B = CY$. GCM of two integral polynomials

is a common factor which has the highest degree (for x, here); you have to write a program which calculates the GCM of two polynomials. It is known that GCM of given two polynomials is unique when constant multiplication factor is ignored. That is, when C and D are both GCM of some two polynomials A and B, $p \times C = q \times D$ for some nonzero integers p and q.

Input

The input consists of multiple datasets. Each dataset constitutes a pair of input lines, each representing a polynomial as an expression defined below.

1. A primary is a variable x, a sequence of digits 0 - 9, or an expression enclosed within ( ). Examples: x, 99, (x+1).
2. A factor is a primary by itself or a primary followed by an exponent. An exponent consists of a symbol ^followed by a sequence of digits 0 - 9. Examples: x^05, 1^15, (x+1)^3.
3. A term consists of one or more adjoining factors. Examples: 4x, (x+1)(x-2), 3(x+1)^2.
4. An expression is one or more terms connected by either + or -. Additionally, the first term of an expression may optionally be preceded with a minus sign -. Examples: -x+1, 3(x+1)^2-x(x-1)^2.

Integer constants, exponents, multiplications (adjoining), additions (+) and subtractions/negations (-) have their ordinary meanings. A sequence of digits is always interpreted as an integer constant. For example, 99 means 99, not 9×9.

Any subexpressions of the input, when fully expanded normalized, have coefficients less than 100 and degrees of x less than 10. Digit sequences in exponents represent non-zero values. All the datasets are designed so that a standard algorithm with 32-bit two's complement integers can solve the problem without overflows.

The end of the input is indicated by a line containing a period.

Output

For each of the dataset, output GCM polynomial expression in a line, in the format below. $c_0x\hat{}p_0+c_1x\hat{}p_1 \ldots +c_nx\hat{}p_n$ Where c_i and p_i $(i = 0, \ldots, n)$ are positive integers with $p_0 > p_1 > \ldots > p_n$, and the greatest common divisor of $\{c_i | i = 0, \ldots, n\}$ is 1.

Additionally:

- When c_i is equal to 1, it should be omitted unless corresponding p_i is 0,
- $x\hat{}0$ should be omitted as a whole, and
- $x\hat{}1$ should be written as x.

Example

i.in	standard output
-(x^3-3x^2+3x-1) (x-1)^2 x^2+10x+25 x^2+6x+5 x^3+1 x-1 .	x^2-2x+1 x+5 1

Problem J. Zigzag

Input file: j.in
Output file: standard output

Given several points on a plane, let's try to solve a puzzle connecting them with a zigzag line. The puzzle is to find the zigzag line that passes through all the given points with the minimum number of turns. Moreover, when there are several zigzag lines with the minimum number of turns, the shortest one among them should be found.

For example, consider nine points given in Figure 10.

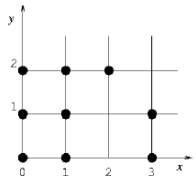

Figure 10: Given nine points

A zigzag line is composed of several straight line segments. Here, the rule requests that each line segment should pass through two or more given points.

A zigzag line may turn at some of the given points or anywhere else. There may be some given points passed more than once.

(a) length= $8 + 3\sqrt{2} \simeq 12.242641$ (b) length = $2\sqrt{2} + (6 + 1/2) + 5\sqrt{5}/2 \simeq 14.918597$

Figure 11: Zigzag lines with three turning points.

Two zigzag lines with three turning points are depicted in Figure 11 (a) and (b) for the same set of given points shown in Figure 10. The length of the zigzag line in Figure 11 (a) is shorter than that in Figure 11 (b). In fact, the length of the zigzag line in Figure 11 (a) is the shortest so that it is the solution for the nine points given in Figure 10. Another zigzag line with four turning points is depicted in Figure 12. Its length is shorter than those in Figure 11, however, the number of turning points is greater than those in Figure 11, and thus, it is not the solution.

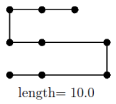

Figure 12: Zigzag line with four turning points.

There are two zigzag lines that passes another set of given points depicted in Figure 13 (a) and (b).

Both have the same number of turning points, and the line in (a) is longer than that in (b). However, the solution is (a), because one of the segments of the zigzag line in (b) passes only one given point, violating the rule.

(a) length= $6 + 2\sqrt{2} \simeq 8.828427$ (b) length= 8.0

Figure 13: Zigzag line with two turning points (a), and not a zigzag line concerned (b).

Your job is to write a program that solves this puzzle.

Input

The input consists of multiple datasets, followed by a line containing one zero. Each dataset has the following format.

n
x_1 y_1
⋮
x_n y_n

Every input item in a dataset is a non-negative integer. Items in a line are separated by a single space.

n is the number of the given points. x_k and y_k $(k = 1, \ldots, n)$ indicate the position of the k-th point. The order of the points is meaningless. You can assume that $2 \leq n \leq 10$, $0 \leq x_k \leq 10$, and $0 \leq y_k \leq 10$.

Output

For each dataset, the minimum number of turning points and the length of the shortest zigzag line with that number of turning points should be printed, separated by a space in a line. The length should be in a decimal fraction with an error less than 0.0001.

You may assume that the minimum number of turning points is at most four, that is, the number of line segments is at most five.

Example

j.in	standard output
2 0 0 10 9 4 0 0 3 1 0 3 3 3 10 2 2 4 2 6 2 2 4 4 4 6 4 2 6 4 6 6 6 3 3 10 0 0 2 0 4 0 0 2 2 2 4 2 0 4 2 4 4 4 6 8 9 0 0 1 0 3 0 0 1 1 1 3 1 0 2 1 2 2 2 10 0 0 1 0 0 1 1 1 9 9 9 10 10 9 10 10 0 2 10 8 10 0 0 0 10 2 0 2 1 2 7 2 10 5 1 6 7 9 2 10 9 0	0 13.45362405 1 18.48683298 3 24.14213562 4 24.94813673 3 12.24264069 3 60.78289622 3 502.7804353