Problem K. Handmade Billiards Table

Ballon:	
Time limit:	1 seconds
Memory limit:	512 megabytes

Tri is a boy who loves playing billiards. His wish is to become an international champion player like Bao Phuong Vinh, but because he did not have money to go to a billiards club to practice, he decided to make himself a billiards table to practice.

Unfortunately, **Tri**'s skills in making billiard tables are limited, so the billiard tables he creates are not as good as tables that meet international standards. Pointing out the disadvantages of the Billiards table that **Tri** created are countless, but the main disadvantages are:

- The billiard table may not be rectangular, but it is definitely a convex quadrilateral.

- Each time the ball collides with the edge of the table, the ball's velocity after colliding with the edge of the table is only half of the velocity before collision.

However, the billiard table that **Tri** created still satisfies the properties of a billiard table. When the ball collides with the edge of the table, it will reflect like light when it hits a flat mirror. Every point on the table surface has the same friction, meaning the ball always has a uniformly decreasing velocity when rolling on the table surface, called the backward acceleration $a\ cm/s^2$. To prove that the billiard table he made can still be used, **Tri** placed a red ball and a white circular marker on the table. The red ball and white marker both have a radius of r cm, their centers are respectively at coordinates (x_r, y_r) and (x_w, y_w). At first, the ball and the maker do not touch each other and are at least 1 cm away from the table wall. After that, **Tri** uses a billiard cue to create an initial velocity for the red ball so that when the red ball stops, it fits neatly at the marker position. Because **Tri** is not a professional billiards player, he cannot make the initial speed of the red ball greater than the value $v\ cm/s$. However, he wants to cover up the disadvantage as mentioned above, so he tries to hit the red ball as many times as possible on the edge of the table before touching the white maker. Knowing that, the ball and the maker are said to touch each other if the distance between their centers is less than or equal to the sum of their radii.

Question: How many positions can **Tri** create where the red ball collides with the edge of the table before reaching the white maker position?

Input

The first line is the integer coordinates of 4 consecutive adjacent vertices $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$ with units of length is cm of the billiard table. Where, $1 \leq x_i, y_i \leq 1000$.

The second line contains 5 integers which are the integer coordinates of the center of the red ball (x_r, y_r), the coordinates of the center of the white marker (x_w, y_w), and their radius r (cm). The red ball and white marker are always satisfied to be inside the billiards table and at least 1 cm away from the table wall, and $1 \leq r \leq 3$.

The third line is two real numbers representing the backward acceleration a (cm/s^2) of the ball relative to the table surface and the initial velocity v (cm/s) of the red ball. Where, $1 < a \leq 1000$ and $1 \leq v \leq 1000$.

Output

The first line is an integer n representing the maximum number of positions where the red ball has collided with the table wall before reaching the position of the white marker without previously touching the maker. If the red ball cannot be brought to the white marker position, whether it touches the edge of the table at least once or not, print '0'.

The next n lines, each line contains 2 real numbers x, y and 2 integers i, j respectively represent the coordinates of the center of the red ball (x, y) when it collides with the edge of the table, which is the straight line connecting i vertex and vertex j of the billiard table. These n lines are sorted in collision

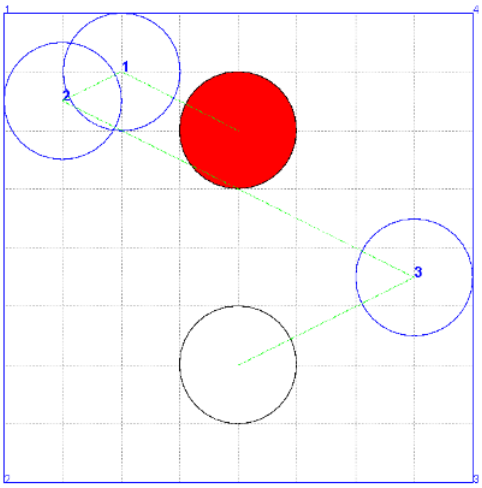
order. The maximum absolute error allowed is 10^{-6}. If there are multiple answers, you can print any one answer.

Examples

standard input	standard output
0 0 0 8 8 8 8 0 4 2 4 6 1 1.0 1000.0	3 2.0 1.0 4 1 1.0 1.5 1 2 7.0 4.5 3 4
3 11 14 15 14 1 1 1 7 5 11 6 2 2.0 500.0	2 4.554961773 3.0 3 4 3.596421722 3.784069582 4 1

Note

Description of input and output of example 1:



Description of input and output of example 2: