Problem A
Amalgamated Artichokes
Time limit: 5 seconds

Fatima Cynara is an analyst at Amalgamated Artichokes (AA). As with any company, AA has had some very good times as well as some bad ones. Fatima does trending analysis of the stock prices for AA, and she wants to determine the largest decline in stock prices over various time spans. For example, if over a span of time the stock prices were 19, 12, 13, 11, 20 and 14, then the largest decline would be 8 between the first and fourth price. If the last price had been 10 instead of 14, then the largest decline would have been 10 between the last two prices.

Picture by Hans Hillewaert via Wikimedia Commons

Fatima has done some previous analyses and has found that the stock price over any period of time can be modelled reasonably accurately with the following equation:

$$\text{price}(k) = p \cdot (\sin(a \cdot k + b) + \cos(c \cdot k + d) + 2)$$

where p, a, b, c and d are constants. Fatima would like you to write a program to determine the largest price decline over a given sequence of prices. Figure A.1 illustrates the price function for Sample Input 1. You have to consider the prices only for integer values of k.

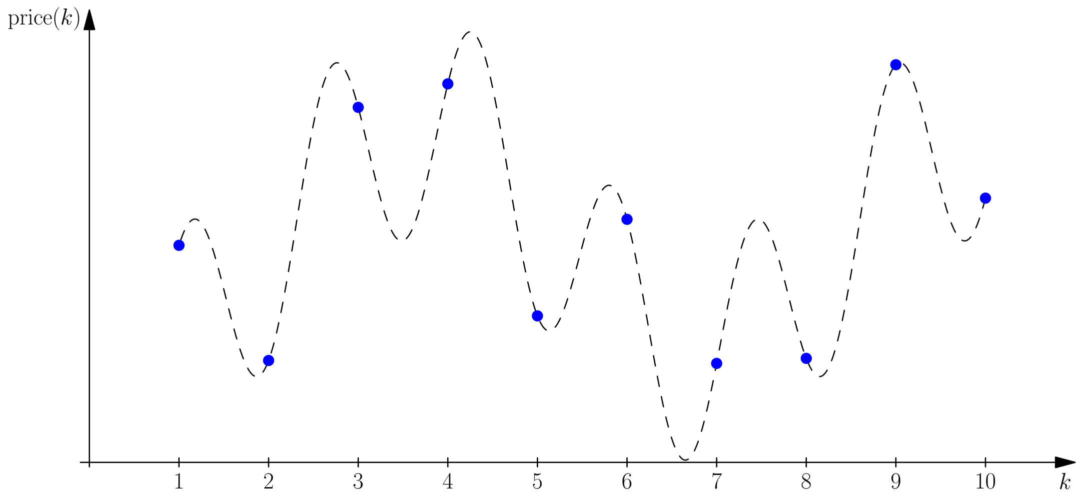

Figure A.1: Sample Input 1. The largest decline occurs from the fourth to the seventh price.

Input

The input consists of a single line containing 6 integers p ($1 \le p \le 1\,000$), a, b, c, d ($0 \le a, b, c, d \le 1\,000$) and n ($1 \le n \le 10^6$). The first 5 integers are described above. The sequence of stock prices to consider is $\text{price}(1), \text{price}(2), \ldots, \text{price}(n)$.

Output

Display the maximum decline in the stock prices. If there is no decline, display the number 0. Your output should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
`42 1 23 4 8 10`	`104.855110477`

Sample Input 2	Sample Output 2
`100 7 615 998 801 3`	`0.00`

Sample Input 3	Sample Output 3
`100 432 406 867 60 1000`	`399.303813`

Problem B
Asteroids
Time limit: 2 seconds

The year is 2115. The asteroid communication relay system was set up a decade ago by the Asteroid Communication Ministry. It is running fine except for one small problem – there are too many asteroids! The smaller ones not only keep interfering with the signals from the relay stations but they are also a danger to all the maintenance aircrafts that fly between the stations. These small asteroids must be destroyed! The Interplanetary Coalition to Prevent Catastrophes (ICPC) has been charged with removing these dangerous asteroids and has hired an elite team of hot-shot pilots for the job. Han Duo is the captain of this team of asteroid destroyers. Armed with his missiles, Han flies through the asteroid belt blowing up any asteroid that the ICPC deems a nuisance.

The ICPC is having some unfortunate budgetary problems. One result of this is that Han and his team do not have as many missiles as they would like, so they cannot blow up all the troublesome asteroids. But the asteroids are small and the missiles are powerful. So if two asteroids are near each other and line up properly, it is possible to take out both with a single missile.

Han's screen displays asteroids as non-rotating two-dimensional simple convex polygons, each of which moves at a fixed velocity. He has decided that the best time to hit two asteroids is when the overlap of the two polygons is at a maximum. For example, Figure B.1, which illustrates Sample Input 1, shows two asteroids and snapshots of their subsequent positions at 1-second intervals. The two asteroids start touching after 3 seconds and the maximum overlap area occurs between 4 and 5 seconds.

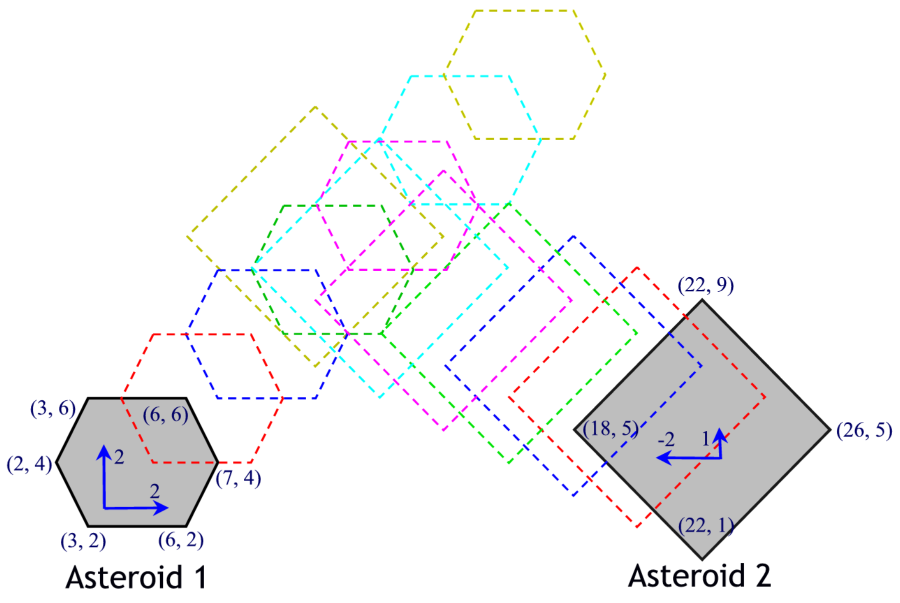

Figure B.1: Sample Input 1. Two asteroids with crossing paths.

Calculating when the maximum overlap occurs for two asteroids requires a bit of programming, but unfortunately Han slept through most of his coding classes at the flight academy. This is where you come in.

Input

The input consists of two asteroid specifications. Each has the form $n\ x_1\ y_1\ x_2\ y_2\ \ldots\ x_n\ y_n\ v_x\ v_y$ where n ($3 \le n \le 10$) is the number of vertices, each x_i, y_i ($-10\,000 \le x_i, y_i \le 10\,000$) are the coordinates of a vertex of the asteroid on Han's screen given in clockwise order, and v_x, v_y ($-100 \le v_x, v_y \le 100$) are the x and y velocities (in units/second) of the asteroid. The x_i, y_i values specify the location of each asteroid at time $t = 0$, and the polygons do not intersect or touch at this time. The maximum length of any side of an asteroid is 500. All numbers in the input are integers.

Output

Display the time in seconds when the two polygons have maximum intersection, using the earliest such time if there is more than one. If the two polygons never overlap but touch each other, treat it as an intersection where the common area is zero and display the earliest such time. If the polygons never overlap or touch, display `never` instead. You should consider positive times only. Your output should have an absolute or relative error of at most 10^{-3}.

Sample Input 1

```
6 3 2 2 4 3 6 6 6 7 4 6 2 2 2
4 18 5 22 9 26 5 22 1 -2 1
```

Sample Output 1

```
4.193518
```

Sample Input 2

```
4 0 0 0 2 2 2 2 0 -1 1
4 10 0 10 2 12 2 12 0 1 1
```

Sample Output 2

```
never
```

Problem C
Catering
Time limit: 4 seconds

Picture from Wikimedia Commons

Paul owns a catering company and business is booming. The company has k catering teams, each in charge of one set of catering equipment. Every week, the company accepts n catering requests for various events. For every request, they send a catering team with their equipment to the event location. The team delivers the food, sets up the equipment, and instructs the host on how to use the equipment and serve the food. After the event, the host is responsible for returning the equipment back to Paul's company.

Unfortunately, in some weeks the number of catering teams is less than the number of requests, so some teams may have to be used for more than one event. In these cases, the company cannot wait for the host to return the equipment and must keep the team on-site to move the equipment to another location. The company has an accurate estimate of the cost to move a set of equipment from any location to any other location. Given these costs, Paul wants to prepare an Advance Catering Map to service the requests while minimizing the total moving cost of equipment (including the cost of the first move), even if that means not using all the available teams. Paul needs your help to write a program to accomplish this task. The requests are sorted in ascending order of their event times and they are chosen in such a way that for any $i < j$, there is enough time to transport the equipment used in the i^{th} request to the location of the j^{th} request.

Input

The first line of input contains two integers n ($1 \leq n \leq 100$) and k ($1 \leq k \leq 100$) which are the number of requests and the number of catering teams, respectively. Following that are n lines, where the i^{th} line contains $n - i + 1$ integers between 0 and $1\,000\,000$ inclusive. The j^{th} number in the i^{th} line is the cost of moving a set of equipment from location i to location $i + j$. The company is at location 1 and the n requests are at locations 2 to $n + 1$.

Output

Display the minimum moving cost to service all requests. (This amount does not include the cost of moving the equipment back to the catering company.)

Sample Input 1	Sample Output 1

3 2
40 30 40
50 10
50
	80

Sample Input 2

```
3 2
10 10 10
20 21
21
```

Sample Output 2

```
40
```