Problem A
Amalgamated Artichokes

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.

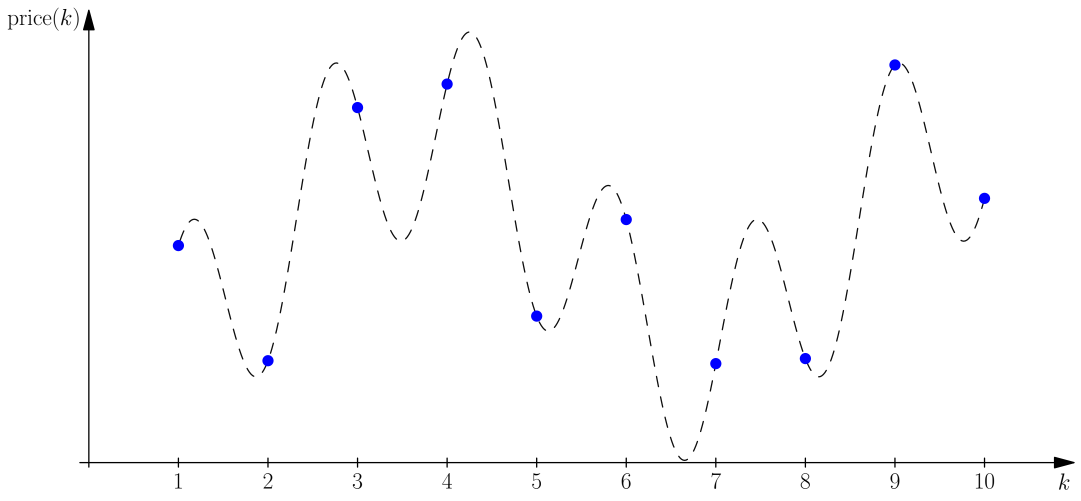

Figure A.1: Sample Input 1. The largest decline occurs from the fourth to the seventh price.

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

```
42 1 23 4 8 10
```

Sample Output 1

```
104.855110477
```

Sample Input 2

```
100 7 615 998 801 3
```

Sample Output 2

```
0.00
```

Sample Input 3

```
100 432 406 867 60 1000
```

Sample Output 3

```
399.303813
```

Problem B
Game Strategy

Alice and Bob are playing a board game. The board is divided into positions labeled $a, b, c, d, \ldots$ and the players use a gamepiece to mark the current position. Each round of the game consists of two steps:

1. Alice makes a choice. Depending on the current position, she has different options, where each option is a set of positions. Alice chooses one set S among the available sets of positions.
2. Bob makes a choice. His choice is one position p from the set S that Alice chose in step 1. Bob moves the gamepiece to position p, which is the position for the start of the next round.

Prior to the first round, each player independently selects one of the positions and reveals it at the start of the game. Bob's position is where the game starts. Alice wins the game if she can force Bob to move the gamepiece to the position she has chosen. To make things interesting, they have decided that Bob will pay Alice a certain amount if he loses, but Alice must pay Bob a certain amount after every round. The game now ends if Alice's position is reached or when Alice runs out of cash.

Both Alice and Bob play optimally: Alice will always choose an option that will lead to her winning the game, if this is possible, and Bob will always try to prevent Alice from winning.

For all possible start and end positions, Alice would like you to determine whether she can win the game and if so, how many rounds it will take.

Input

The input consists of a single test case. The first line contains the number of positions n ($1 \leq n \leq 25$). The n positions are labeled using the first n letters of the English alphabet in lowercase. The rest of the test case consists of n lines, one for each position p, in alphabetical order. The line for position p contains the options available to Alice in position p. It starts with the number of options m ($1 \leq m < 2^n$), which is followed by m distinct strings, one for each option. Each string contains the positions available to Bob if Alice chooses that option. The string has at least 1 character, the characters (which correspond to valid board positions) are in alphabetical order, and no characters are duplicated. The total number of options for the test case is at most 10^6.

Output

For each position p in alphabetical order, display one line. In that line, for each position q in alphabetical order display the minimal number of rounds in which Alice can be guaranteed to arrive at position q when starting the game in position p, or -1 if Alice cannot be guaranteed to reach q from p.

Sample Input 1	Sample Output 1
2 2 ab b 1 b	0 1 -1 0

Sample Input 2

```
3
1 b
2 b a
2 ab ac
```

Sample Output 2

```
0 1 -1
1 0 -1
2 2 0
```

Problem C
Pirate Chest

Pirate Dick finally had enough of fighting, marauding, theft, and making life miserable for many on the open seas. So he decided to retire, and he found the perfect island to spend the rest of his days on, provided he does not run out of money. He has plenty of gold coins now, and he wants to store them in a chest (he is a pirate after all). Dick can construct a rectangular chest with integer dimensions of any size up to a specified maximum size for the top but with an arbitrary integer height. Now he needs a place to hide the chest. While exploring the island, he found the perfect solution.

Dick will hide his chest by submerging it in a murky pond. The pond has a rectangular surface, and it completely fills the bottom of a valley that has high vertical rocky walls. Dick surveyed the pond and knows its depth for each of the squares of a Cartesian coordinate grid system placed on the pond surface. When Dick submerges the chest, it will sink as far as possible until it touches the bottom. The top of the chest will remain parallel to the pond's surface and the chest will be aligned with the grid squares. The water displaced by the submerged chest will raise the level of the pond's surface (this will occur even if there is no space around the chest for the displaced water to rise). The walls of the valley are high enough that the water can never splash out of the valley. Of course, since the chest must be invisible, its top must be strictly below the surface of the pond. Your job is to find the volume of the largest chest that Pirate Dick can hide this way.

In Figure C.1, the leftmost image shows a pond, the middle image shows a possible placement of a chest of volume 3, and the rightmost image shows a placement of a chest of volume 4, which is the maximum possible volume. Note that if the second chest were made one unit taller, its top would be visible because it would be at exactly the same height as the surface of the water.

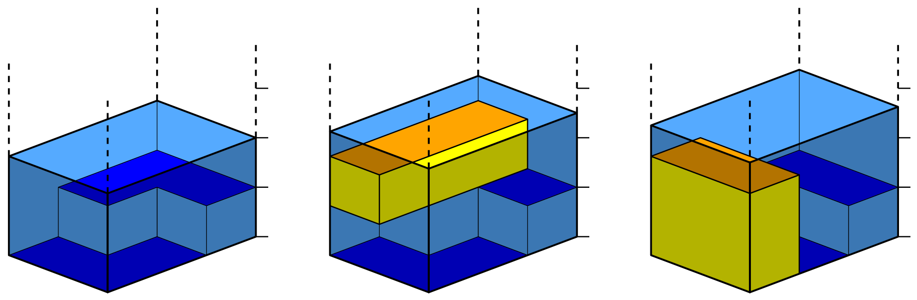
Figure C.1: Illustration of Sample Input 1.

Input

The input consists of a single test case. A test case starts with a line containing four integers a, b, m, and n ($1 \le a, b, m, n \le 500$). The pond's surface dimensions are $m \times n$ and the maximum size of the top (and bottom) of the chest is $a \times b$. In addition, a and b are small enough that it is not possible to cover the entire pond with a chest with top size $a \times b$. Each of the remaining m lines in a test case contains n integers $d_{i,j}$ specifying the pond's depth at grid square (i, j), where $0 \le d_{i,j} \le 10^9$ for each $1 \le i \le m$ and $1 \le j \le n$.

Output

Display the maximum volume of a rectangular chest with integer dimensions (where one of the dimensions of the top is bounded by a and the other is bounded by b) that can be completely submerged below the surface of the pond. If no chest can be hidden in the pond, display `0`.

Sample Input 1

```
3 1 2 3
2 1 1
2 2 1
```

Sample Output 1

```
4
```

Sample Input 2

```
4 1 1 5
2 0 2 2 2
```

Sample Output 2

```
12
```

Sample Input 3

```
2 3 3 5
2 2 2 2 2
2 2 2 2 2
2 2 2 2 2
```

Sample Output 3

```
18
```

Problem D
Bus Tour

Imagine you are a tourist in Warsaw and have booked a bus tour to see some amazing attraction just outside of town. The bus first drives around town for a while (a *long* while, since Warsaw is a big city) picking up people at their respective hotels. It then proceeds to the amazing attraction, and after a few hours goes back into the city, again driving to each hotel, this time to drop people off.

For some reason, whenever you do this, your hotel is always the first to be visited for pickup, and the last to be visited for dropoff, meaning that you have to suffer through two not-so-amazing sightseeing tours of all the local hotels. This is clearly not what you want to do (unless for some reason you are *really* into hotels), so let's fix it. We will develop some software to enable the sightseeing company to route its bus tours more fairly—though it may sometimes mean longer total distance for everyone, but fair is fair, right?

For this problem, there is a starting location (the sightseeing company headquarters), h hotels that need to be visited for pickups and dropoffs, and a destination location (the amazing attraction). We need to find a route that goes from the headquarters, through all the hotels, to the attraction, then back through all the hotels again (possibly in a different order), and finally back to the headquarters. In order to guarantee that none of the tourists (and, in particular, *you*) are forced to suffer through two full tours of the hotels, we require that every hotel that is visited among the first $\lfloor h/2 \rfloor$ hotels on the way to the attraction is also visited among the first $\lfloor h/2 \rfloor$ hotels on the way back. Subject to these restrictions, we would like to make the complete bus tour as short as possible. Note that these restrictions may force the bus to drive past a hotel without stopping there (this is not considered visiting) and then visit it later, as illustrated in the first sample input.

Input

The first line of each test case consists of two integers n and m satisfying $3 \le n \le 20$ and $2 \le m$, where n is the number of locations (hotels, headquarters, attraction) and m is the number of pairs of locations between which the bus can travel.

The n different locations are numbered from 0 to $n-1$, where 0 is the headquarters, 1 through $n-2$ are the hotels, and $n-1$ is the attraction. Assume that there is at most one direct connection between any pair of locations and it is possible to travel from any location to any other location (but not necessarily directly).

Following the first line are m lines, each containing three integers u, v, and t such that $0 \le u, v \le n-1$, $u \ne v$, $1 \le t \le 3600$, indicating that the bus can go directly between locations u and v in t seconds (in either direction).

Output

For each test case, display the case number and the time in seconds of the shortest possible tour.

Sample Input 1

```
5 4
0 1 10
1 2 20
2 3 30
3 4 40
4 6
0 1 1
0 2 1
0 3 1
1 2 1
1 3 1
2 3 1
```

Sample Output 1

```
Case 1: 300
Case 2: 6
```

Problem E
Crane Balancing

Wherever there is large-scale construction, you will find cranes that do the lifting. One hardly ever thinks about what marvelous examples of engineering cranes are: a structure of (relatively) little weight that can lift much heavier loads. But even the best-built cranes may have a limit on how much weight they can lift.

The Association of Crane Manufacturers (ACM) needs a program to compute the range of weights that a crane can lift. Since cranes are symmetric, ACM engineers have decided to consider only a cross section of each crane, which can be viewed as a polygon resting on the x-axis.

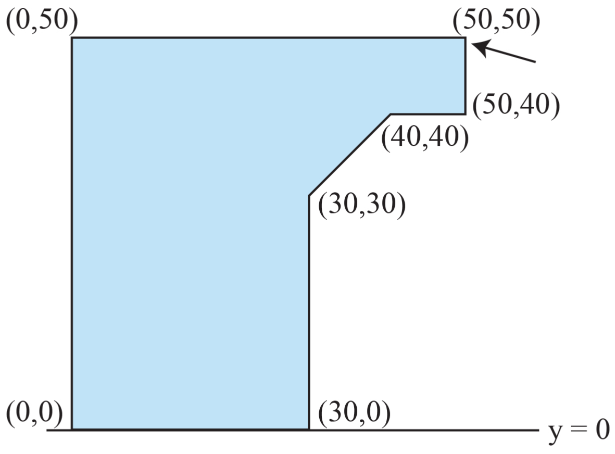

Figure E.1: Crane cross section

Figure E.1 shows a cross section of the crane in the first sample input. Assume that every 1×1 unit of crane cross section weighs 1 kilogram and that the weight to be lifted will be attached at one of the polygon vertices (indicated by the arrow in Figure E.1). Write a program that determines the weight range for which the crane will not topple to the left or to the right.

Input

The input consists of a single test case. The test case starts with a single integer n ($3 \leq n \leq 100$), the number of points of the polygon used to describe the crane's shape. The following n pairs of integers x_i, y_i ($-2\,000 \leq x_i \leq 2\,000, 0 \leq y_i \leq 2\,000$) are the coordinates of the polygon points in order. The weight is attached at the first polygon point and at least two polygon points are lying on the x-axis.

Output

Display the weight range (in kilograms) that can be attached to the crane without the crane toppling over. If the range is $[a, b]$, display $\lfloor a \rfloor$ `..` $\lceil b \rceil$. For example, if the range is $[1.5, 13.3]$, display `1 .. 14`. If the range is $[a, \infty)$, display $\lfloor a \rfloor$ `.. inf`. If the crane cannot carry any weight, display `unstable` instead.

Sample Input 1	Sample Output 1

```
7
50 50
0 50
0 0
30 0
30 30
40 40
50 40
```
	`0 .. 1017`

Sample Input 2	Sample Output 2

```
7
50 50
0 50
0 0
10 0
10 30
20 40
50 40
```
	`unstable`

Problem F
Ship Traffic

Ferries crossing the Strait of Gibraltar from Morocco to Spain must carefully navigate to avoid the heavy ship traffic along the strait. Write a program to help ferry captains find the largest gaps in strait traffic for a safe crossing.

Your program will use a simple model as follows. The strait has several parallel shipping lanes in east-west direction. Ships run with the same constant speed either eastbound or westbound. All ships in the same lane run in the same direction. Satellite data provides the positions of the ships in each lane. The ships may have different lengths. Ships do not change lanes and do not change speed for the crossing ferry.

The ferry waits for an appropriate time when there is an adequate gap in the ship traffic. It then crosses the strait heading northbound along a north-south line at a constant speed. From the moment a ferry enters a lane until the moment it leaves the lane, no ship in that lane may touch the crossing line. Ferries are so small you can neglect their size. Figure F.1 illustrates the lanes and ships for Sample Input 1. Your task is to find the largest time interval within which the ferry can safely cross the strait.

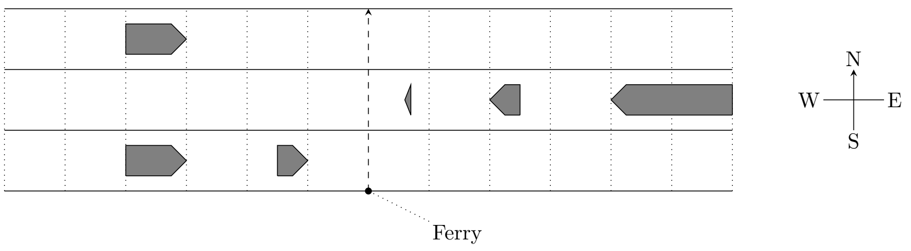

Figure F.1: Sample Input 1.

Input

The first line of input contains six integers: the number of lanes n ($1 \leq n \leq 10^5$), the width w of each lane ($1 \leq w \leq 1\,000$), the speed u of ships and the speed v of the ferry ($1 \leq u, v \leq 100$), the ferry's earliest start time t_1 and the ferry's latest start time t_2 ($0 \leq t_1 < t_2 \leq 10^6$). All lengths are given in meters, all speeds are given in meters/second, and all times are given in seconds.

Each of the next n lines contains the data for one lane. Each line starts with either `E` or `W`, where `E` indicates that ships in this lane are eastbound and `W` indicates that ships in this lane are westbound. Next in the line is an integer m_i, the number of ships in this lane ($0 \leq m_i \leq 10^5$ for each $1 \leq i \leq n$). It is followed by m_i pairs of integers l_{ij} and p_{ij} ($1 \leq l_{ij} \leq 1\,000$ and $-10^6 \leq p_{ij} \leq 10^6$). The length of ship j in lane i is l_{ij}, and p_{ij} is the position at time 0 of its forward end, that is, its front in the direction it moves.

Ship positions within each lane are relative to the ferry's crossing line. Negative positions are west of the crossing line and positive positions are east of it. Ships do not overlap or touch, and are sorted in increasing order of their positions. Lanes are ordered by increasing distance from the ferry's starting point, which is just south of the first lane. There is no space between lanes. The total number of ships is at least 1 and at most 10^5.

Output

Display the maximal value d for which there is a time s such that the ferry can start a crossing at any time t with $s \leq t \leq s + d$. Additionally the crossing must not start before time t_1 and must start no later than time t_2. The output must have an absolute or relative error of at most 10^{-3}. You may assume that there is a time interval with $d > 0.1$ seconds for the ferry to cross.

Sample Input 1

```
3 100 5 10 0 100
E 2 100 -300 50 -100
W 3 10 60 50 200 200 400
E 1 100 -300
```

Sample Output 1

```
6.00000000
```

Sample Input 2

```
1 100 5 10 0 200
W 4 100 100 100 300 100 700 100 900
```

Sample Output 2

```
50.00000000
```