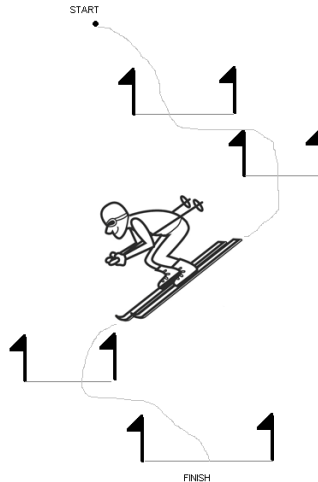


Slalom



In spite of the scarcity of snowfall in Madrid, interest in winter sports is growing in the city, especially with regard to skiing. Many people spend several weekends or even full weeks improving their skills in the mountains.

In this problem we deal with only one of the multiple alpine skiing disciplines: slalom. A course is constructed by laying out a series of gates, which are formed by two poles. The skier must pass between the two poles forming each gate. The winner is the skier who takes the least time to complete the course while not missing any of the gates.

You have recently started to learn to ski, but you have already set yourself the goal of taking part in the Winter Olympic Games of 2018, for which Madrid will presumably present a candidature. As part of the theoretical training, you need to write a program that calculates, given a starting point and a series of gates, the minimum-length path starting from the point given and passing through each gate until you reach the last one, which is the finish line. You may assume that the gates are horizontal and are ordered from highest to lowest, so that you need to pass through them in order. You consider yourself an accomplished skier, so you can make any series of turns, no matter how difficult, and your only concern is minimizing the total length of the path.

Input

The first line of each case gives the number of gates n ($1 \leq n \leq 1000$). The next line contains two floating point numbers, the Cartesian coordinates x and y of the starting position, in that order. Next come n lines with three floating point numbers each, y x_1 x_2 , meaning that the next gate is a horizontal line from (x_1, y) to (x_2, y) . You can safely assume that $x_1 < x_2$. The values of y are strictly decreasing and are always smaller than that of the starting position. The last gate represents the finish line. All coordinates are between $-500\,000$ and $500\,000$, inclusive. A value of 0 for n means the end of the input. A blank line follows each case.

Output

For each test case, output a line with the minimum distance needed to reach the finish line. Your answer should be accurate to within an absolute or relative error of 10^{-7} .

Sample Input

```
2
0 2
1 1 2
0 0.5 3
```

```
3
0 4
3 1 2
2 -1 0
1 1 2
```

```
0
```

Sample Output

```
2.41421356237
4.24264068712
```

Where could we meet?

Introduction

George is the CEO of a chain of stores. From time to time, George wants to arrange meetings with workers from any store. He is not a fan of the new technologies and prefers to talk directly with people.

These meetings always take place at the end of the work day. George realized that due to rush hour, it is impossible to go from some stores to the management building and get there on time.

Since each store also has a meeting room for the store manager to use, George decided that his meetings could also take place at one of the stores. However, workers from any store should be able to get to the meeting place on time during rush hour.

Problem

Consider that you are given a set of directed connections between stores. These connections represent roads that never have rush hour problems at the end of the day. A store s can be used to hold the meeting between George and the workers if there is a path (using one or more connections) from all other stores to s . Your task is to determine how many stores George can use to meet with the workers.

Input

The input is a directed map specified as follows:

- One line with two integers N and M , where N represents the number of stores and M the number of available directed connections between stores.
- A list of M lines where each line contains two integers s_i and s_j (with $1 \leq s_i \leq N$ and $1 \leq s_j \leq N$) that represent an available connection from store s_i to store s_j .

Output

The output is a single integer with the number of stores from which there is a path from all other stores.

Constraints

$$0 < N \leq 10^6$$
$$0 < M \leq 10^6$$

Sample Input 1

```
4 4
1 2
2 1
3 2
4 1
```

Sample Output 1

2

Sample Input 2

4 4
1 2
2 1
2 3
1 4

Sample Output 2

0

Lift

A nursing home has acquired an old building to install a new facility. The building is nicely located and spacious, making it very attractive to prospective clients. There is, however, a slight problem: the only lift, despite being able to carry up to 100 people, is very very slow. So slow that not even people in their latter years have the patience to sometimes wait for it to complete each trip. Because of that, the management decided to install a new AI system in the lift and you will help in its design.

People select their destinations at the ground floor, and given a maximum number of stops per trip, the AI system must determine at which floors the lift should stop. People are not worried that they won't stop at the specified floor (for instance, they may select the 17th floor, but the lift may decide to stop either at the 16th or the 18th floor instead), but they would like the stops to be chosen so that the total walking distance is minimized (exercise is good for you, but there is too much of a good thing, especially if you are elderly). It does not matter if they walk up or down. The walking distance of a single person is measured in the number of floors it has to walk from the nearest lift stop to the desired floor. The AI system must determine how to stop a determined number of times in order to minimize the sum of all persons' walking distance.

Note that people may choose to walk if they think the stops chosen by the lift will make them walk more than walking all the way (i.e., they may "exit" the lift at the ground floor, thus not taking the lift at all). The management has also asked that the AI control stop at the lowest floor when equal cost solutions exist (this way, it also helps to save the planet whilst saving money on electricity). Therefore, for example, and assuming equal costs, stopping on floors 2 and 4 will be preferred to stopping on floors 3 and 4, and stopping on 2, 4 and 6 will be preferred to stopping on 2, 5 and 6.

Can you help in the design of the AI control?

Problem

Given a number of floor requests, and a specific number of stops, you must compute in which floors should the lift make exactly that number of stops in order to minimize the total walking distance of every person.

Input

The first line of input contains two numbers P and S , representing respectively the number of persons taking the lift and the numbers of stops the lift should make ($1 \leq P, S \leq 100$).

Then come exactly P lines, each one with one integer P_i indicating the floor requested by person i ($1 \leq P_i \leq 100$). You can assume that these request will come in ascending order of floor and that the maximum floor requested by someone will always be bigger or equal to the number of lift stops.

Output

The first line of output should contain a single integer, indicating the minimum total walking distance achievable, as described before. Then should come exactly S lines indicating the stops the lift should make (all in different floors), in ascending order (and not including the ground

floor). Remember that if there is more than one solution giving the same total walking distance, you should choose the one that prefers lowest floors.

Sample Input

```
10 3
1
2
3
4
6
6
8
8
10
10
```

Sample Output

```
7
3
6
8
```

BLACK AND WHITE GRAPH COLORING

Problem

Given an undirected graph and a set of k different colors, the graph coloring problem can be defined as assigning each vertex a color such that no two adjacent vertices are assigned the same color.

For $k > 2$, this problem is known to be NP-Complete. However, when considering just 2 colors, efficient solutions exist to color the graph. Suppose you only have two colors (black and white), develop a solution to color a given graph such that the number of black vertices is maximized.

Input

The input defines an undirected graph:

- One line with an integer N that represents the number of vertices in the graph.
- One line with an integer M that represents the number of edges in the graph.
- A list of M lines where in each line there are two integers representing an edge between two vertices. All vertex identifiers are between 1 and N .

Output

The output is a single integer. If the graph can be colored with just two colors, the output should be the maximum number of nodes to be colored as black. Otherwise, the output should be -1 if it is not possible to color the graph using just two colors.

Sample Input 1

```
5
5
1 2
1 3
```

4 2
3 4
5 4

Sample Output 1

3

Sample Input 2

5
6
1 2
1 3
4 2
3 4
5 4
3 5

Sample Output 2

-1

Sample Input 3

7
6
1 2
1 3
3 6
2 6
4 5
6 7

Sample Output 3

4

DELIVERIES

Problem

A small company performs deliveries of packages using just two trucks and the goal is to make deliveries to N cities.

Due to some constraints, the company has defined an order for those N cities that must be respected by the routes of the delivery trucks. Consider that each city is identified by an index from 1 to N that defines its place in the order. As a result, a truck cannot visit city 3 after visiting city 5 because that would violate the ordering defined by the company.

The company has the information of the distance between all pairs of N cities. The objective is to determine the routes of both trucks such that the total distance covered is minimized. Remember that the route of a truck must respect the ordering of the cities. Additionally, assume that the city where each truck starts its route is the first city of the route. Hence, each truck starts its route in a different city.

Input

The input is as follows:

- One line with one integer N that denotes the number of cities.
- A list of N lines where line i contains the distance from city i to all others. Each line is composed of N non-negative integers that represent the distances between cities.

Limits

$$0 < N \leq 1000$$

Output

The output is a single integer with the minimum total distance covered by both trucks.

Sample Input 1

```
5
0 1 2 3 4
1 0 1 2 3
2 1 0 2 2
3 2 2 0 1
4 3 2 1 0
```

Sample Output 1

```
3
```

Sample Input 2

```
8
0 3 3 3 3 6 6 6
3 0 5 5 5 4 4 9
3 5 0 5 5 9 4 4
3 5 5 0 5 4 9 9
3 5 5 5 0 9 9 4
6 4 9 4 9 0 7 8
6 4 4 9 9 7 0 7
6 9 4 9 4 8 7 0
```

Sample Output 2

```
15
```

Trick or Treat

Johnny and his friends have decided to spend Halloween night doing the usual candy collection from the households of their village. As the village is too big for a single group to collect the candy from all houses sequentially, Johnny and his friends have decided to split up so that each of them goes to a different house, collects the candy (or wreaks havoc if the residents don't give out candy), and returns to a meeting point arranged in advance.

There are n houses in the village, the positions of which can be identified with their Cartesian coordinates on the Euclidean plane. Johnny's gang is also made up of n people (including Johnny himself). They have decided to distribute the candy after everybody comes back with their booty. The houses might be far away, but Johnny's interest is in eating the candy as soon as possible.

Keeping in mind that, because of their response to the hospitality of some villagers, some children might be wanted by the local authorities, they have agreed to fix the meeting point by the river running through the village, which is the line $y = 0$. Note that there may be houses on both sides of the river, and some of the houses may be houseboats ($y = 0$). The walking speed of every child is 1 meter per second, and they can move along any direction on the plane.

At exactly midnight, each child will knock on the door of the house he has chosen, collect the candy instantaneously, and walk back along the shortest route to the meeting point. Tell Johnny at what time he will be able to start eating the candy.

Input

Each test case starts with a line indicating the number n of houses ($1 \leq n \leq 50\,000$). The next n lines describe the positions of the houses; each of these lines contains two floating point numbers x and y ($-200\,000 \leq x, y \leq 200\,000$), the coordinates of a house in meters. All houses are at different positions.

A blank line follows each case. A line with $n = 0$ indicates the end of the input; do not write any output for this case.

Output

For each test case, print two numbers in a line separated by a space: the coordinate x of the meeting point on the line $y = 0$ that minimizes the time the last child arrives, and this time itself (measured in seconds after midnight). Your answer should be accurate to within an absolute or relative error of 10^{-5} .

Sample Input

```
2
1.5 1.5
3 0

1
0 0

4
1 4
4 4
-3 3
2 4

5
4 7
-4 0
7 -6
-2 4
8 -5

0
```

Sample Output

```
1.500000000 1.500000000
0.000000000 0.000000000
1.000000000 5.000000000
3.136363636 7.136363636
```