Olympiad Levels

This olympiad is held in two levels: basic level and advanced level.

You can choose the level to participate in by yourself.

If you would like to take part in the basic level contest, you must solve problems A, B, C and D.

If you would like to take part in the advanced level contest, you must solve problems C, D, E and F.

There will be separate standings for basic and advanced levels.

If you submit a solution for problem E or F, you will be automatically classified as a participant of the advanced level (even if your solution does not pass sample tests). In all other cases you will be classfied as a participant of basic level contest.

Problem A. Hotel

Input file:	`hotel.in`
Output file:	`hotel.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Pavel was dreaming to become an architect since his early childhood. He often drew plans of buildings on sheets of paper, and sometimes on tables and walls. Now he has a university degree and is a famous architect.

Once Pavel was digging in his child drawings and found an interesting plan of the hotel floor. The floor is a rectangle with size $n \times m$ meters. In the plan, each square corresponding to a square meter of the floor was painted with either blue or red color. After thinking a little, Pavel understood that the blue color is for corridors, and the red color stands for hotel rooms.

Since his childhood, Pavel tries to comply with international standards. Following the standards, a hotel room must be rectangular, so it should be represented on the plan as a rectangle filled with red color. In order to comply with the fire safety regulations, there should be corridors at the sides of the floor. So, the first and the last rows, as well as the first and the last columns of the plan are blue, and each hotel room is surrounded with the corridors.

Looking at his plan, Pavel wondered, what is the maximal area of a hotel room on his plan. Write a program to find this area.

Input

The input file contains the description of Pavel's plan.

The first line of the input file contain two integers n and m ($3 \le n, m \le 30$). The following n lines contain m symbols each. The symbol "`*`" corresponds to a square filled with blue color (a square meter of a corridor), and the symbol "`.`" is a red square — a square meter of a room.

Output

In the first line of the output file print the area of the largest hotel room complying with the standard. In there is no such a room, print "`-1`".

Examples

hotel.in	hotel.out
5 5 ***** *...* *...* *...* *****	9
4 5 ***** *.*** *.*.* *****	2
3 3 *** *** ***	-1

Problem B. Great Wall of Flatland

Input file:	`greatwall.in`
Output file:	`greatwall.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Once Emperor of Flatland Hui Shuandi decided to strengthen his great empire and commanded to build the Great Wall of Flatland. At that moment Flatland consisted of a number of Warring Kingdoms. Each of the them had triangular shape and they did not overlap with each other. Along the perimeter of each of them were built unbreakable walls, and in every corner of each kingdom there was a guard tower.

The land had been distributed by the previous ruler of Flatland in such a manner that from any of the towers on a clear day one could see any other. As we know from the writings of the Great Flatland Mathematicians this is possible only in the case where are no three towers on a same straight line.

In addition, since the imperial tax office had to collect taxes from all the Warring Kingdoms, and tax officers had to carry with them large sums of money, the country was constructed in such a way that the tax officer could get from any of the Kingdom to any other using only unbreakable walls, possibly of different Kingdoms.

To find out how much money should be allocated for the construction of the Great Wall of Flatland, Hui Shuandi need to know the total length of all unbreakable walls, which separates the territory of some Warring Kingdom from the territory not belonging to Flatland.

Input

First line of the input file contains an integer number m — the number of Warring Kingdoms ($1 \le m \le 1000$). Each of the following m lines contains six numbers x_1, y_1, x_2, y_2, x_3, y_3 — coordinates of vertices of triangle that describes the area of the corresponding Warring Kingdom. All coordinates are integer and do not exceed 10^4 by absolute value.

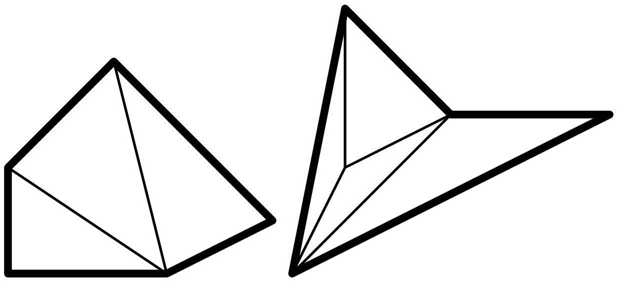
Output

Output the length of The Great Wall of Flatland. You answer should be accurate up to 10^{-4}.

Examples

greatwall.in	greatwall.out
3 0 0 3 0 0 2 3 0 0 2 2 4 3 0 2 4 5 1	14.3071357893652
4 0 0 1 5 1 2 1 5 1 2 3 3 0 0 3 3 1 2 0 0 3 3 6 3	17.6356505708383

Illustrations for the examples:



Problem C. Enigmatic Number

Input file:	`enigmatic.in`
Output file:	`enigmatic.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

John is a junior researcher in the *Research Institute for the Meaning of Life*. Recently, he discovered that the decimal number N contains knowledge that can shed light on some of the issues of human life. However, John is selfish person and he does not want to share his discovery. So he decided to remember this number and to eat all records associated with it.

Unfortunately, John is a bit distracted and forgetful, so he decided to use the following way to remember the number N. There is a set of K numbers connected with the life of John, which he remembers very well, even better than the date of his birth. All these numbers contain no more than three digits.

John is trying to represent the number N as a concatenation of numbers from this set (e.g., a concatenation of the numbers 1 and 2 can give 12 or 21 depending on the order). No number can be used twice, on the other hand it is not obligatory to use all numbers. John wants to find (for easier remembering) a representation, containing as few numbers as possible.

Write a program that will find such a representation.

Input

First line of the input file contains an integer number N ($1 \le N < 10^{45}$). Second line contains an integer number K — size of the memorable numbers set ($1 \le K \le 1000$). Third line contains K memorable numbers ($0 \le a_i < 1000$). All numbers in the set are different and do not have extra leading zeroes.

Output

In the first line of the output file output an integer number M — size of the desired partition. In the following M lines output the numbers forming the partition (in the order in which they need to be concatenated to obtain the number N).

If there exist multiple partitions satisfying the criteria described, output any of them. It is guaranteed that at least one desired partition of N exists.

Examples

enigmatic.in	enigmatic.out
123 4 1 3 12 23	2 12 3
123 4 1 2 3 123	1 123

In 40% of test cases K will not exceed 8.

Problem D. If My Memory Doesn't Fail Me...

Input file:	`memory.in`
Output file:	`memory.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

John Cameroff, a widely known film director, is preparing for shooting of his next film, "Smiley". There will be lots of modern 3-D visual computer effects in the film, which will definetely lead to unprecedented success. More specifically, in the first week it is planned to gather less than 1000000000 milligrams of smashed tomatoes and rotten eggs that will be thrown to the cinema screens by indignant spectators.

John has calculated everything thoroughly. To make the protographic quality of visual effects, he needs N computers which he plans to buy at the computer stall nearby. To make sure that the computers he is buying are of the highest quality, he decided to test the RAM of each of the computers. He has even bought M USB devices designed specially for that.

Each of these devices, when plugged into a computer, tests more and more cells of RAM. It takes K hours for one computer to be tested completely. If one unplugs a device from a computer, the memory checking process is paused. If then one plugs a device again (the same, or another one), the process resumes from the point where it stopped previously. Nothing bad will happen in the device remains plugged after memory is checked.

John Cameroff appreciates his time. So he wants to know what is the minimal possible time for N computers to be tested. Moreover, he wants to know the sequence of actions that leads to this result. Your task is to help him to solve this problem.

Input

The first line of the input file contains three integers N, M and K ($1 \le N, M, K \le 1000$).

Output

In the first line of the output file print the minimal time T_{min}, which takes all computers to be tested. The time should be printed as an irreducible fraction in the format `A/B`, where A is the numerator and B is the denominator. The following inequations should be held: $0 \le A \le 10^9$, $1 \le B \le 10^9$.

In the second line, print the number of actions C. Each of the following C lines should contain information about an action in the following format:

`N_i/D_i: Connect V_i to K_i`

This means that after $\frac{N_i}{D_i}$ hours from the beginning of the testing the device with number V_i ($1 \le V_i \le M$) should be plugged to the computer with number K_i ($1 \le K_i \le N$). If the device V_i has been plugged to any computer, it is unplugged first. If a device was plugged to computer K_i, it is also unplugged.

Fractions $\frac{N_i}{D_i}$ should all be irreducible. For numbers N_i, D_i the following inequations should be held: $0 \le N_i \le 10^9$, $1 \le D_i \le 10^9$.

The actions must be printed in the order of nondecreasing of time. The number of actions C should not exceed 10000.

If there are several possible sequences of actions that take the minimal time to be executed and satisfy the constraints on the number of actions, output any one.

Examples

memory.in	memory.out
2 2 1	1/1 2 0/1: Connect 2 to 1 0/1: Connect 1 to 2
3 2 1	3/2 4 0/1: Connect 1 to 1 0/1: Connect 2 to 2 1/2: Connect 1 to 3 1/1: Connect 2 to 1

Problem E. Magic Potions

Input file:	`magic.in`
Output file:	`magic.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

L'Ashyto is a famous magician. He is working in his laboratory on a new set of magic potions to sell on the upcoming magic fair. He has bought several bottles of each of n different magic substances that can be used to create magic potions. Each magic potion must be composed of two different magic substances. To create a potion the magician mixes the contains of two bottles and pronounces the magic spell.

The price of a potion is proportional to the quality of the ingredients: the better substances are used — the more expensive the potion is. Of course, L'Ashyto would like to create as expensive potions as possible. On the other side, differences in prices are not as significant as the prices themselves. Therefore the first priority is the number of potions.

L'Ashyto has numbered all substances from 1 to n, substance number 1 being the best, and substance number n being the worst. He would like to create as many potions as possible. If there are several ways to do so, he would like to create as many potions from substances 1 and 2, as possible. If there are still several variants, he would like to maximize the number of $1+3$ potions, then $1+4$ potions, etc, $1+n$ potions, $2+3$ potions, etc. Help him to find out how many potions of which ingredients to create.

Input

The first line of the input file contains n — the number of substances L'Ashyto has ($2 \le n \le 100\,000$). The second line contains n integer numbers: $a_1, a_2, \ldots, a_n$ — the number of bottles of the first substance, the number of bottles of the second substance, etc ($1 \le a_i \le 10^9$).

Output

The first line of the output file must contain m — the number of different types of potions L'Ashyto must create. The following m lines must contain three integer numbers each: i, j, c_{ij} — the numbers of substances to create a potion of, and how many such potions to create. For each description there must be $i < j$. Output potion description in order of decreasing their price.

Examples

magic.in	magic.out
4 1 1 1 1	2 1 2 1 3 4 1

Problem F. Roads

Input file:	`roads.in`
Output file:	`roads.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

After inventing the wheel, the citizens of Flatland decided to connect their n cities by roads. Since building roads turned out to be quite expensive, they decided to build as few roads as possible so that one could get from any town to any other town by roads. Since traffic rules were not invented then yet, it was allowed to travel along each road in any direction. Flatland is flat, so there is no need to go around mountains or valleys, therefore each road is a straight line segment connecting two cities.

Of course citizens immediately started to argue which cities would be connected by the roads. Citizens of each city wanted to have more roads going from their city. To stop negotiations the king of Flatland issued an order: for each city he claimed the number of roads that would go from it. Since the king was very wise, all these numbers were from 1 to $n-1$ and their sum was $2n-2$.

But the minister of road building was not as wise as king was. Even given these numbers, he couldn't find the way to build roads. The problem was being complicated by the additional fact that bridges, tunnels or traffic lights were not known that time, so roads didn't have to intersect.

You are his last hope to be saved from terrible execution.

Input

The first line of the input file contains n — the number of cities in Flatland ($2 \le n \le 1000$). The following n lines contain three integer numbers each: x_i, y_i and d_i. Here x_i and y_i are the coordinates of the corresponding city, and d_i is the number of roads that must go from it. The coordinates do not exceed 10^4 by their absolute values. No three cities are on the same straight line.

Output

If it is impossible to build the roads, output -1 at the first line of the output file.

In the other case, output $n-1$ lines. Each line must contain two integer numbers — the numbers of the cities to be connected by the corresponding road. The cities are numbered from 1 to n in order they are given in the input file.

Examples

roads.in	roads.out
7 0 6 1 0 0 1 2 3 1 3 2 3 4 4 4 7 0 1 7 6 1	1 5 2 4 3 4 4 5 5 6 5 7