Codeforces Beta Round #82 (Div. 2)

A. Card Game

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is a card game called "Durak", which means "Fool" in Russian. The game is quite popular in the countries that used to form USSR. The problem does not state all the game's rules explicitly — you can find them later yourselves if you want.

To play durak you need a pack of 36 cards. Each card has a suit ("S", "H", "D" and "C") and a rank (in the increasing order "6", "7", "8", "9", "T", "J", "Q", "K" and "A"). At the beginning of the game one suit is arbitrarily chosen as trump.

The players move like that: one player puts one or several of his cards on the table and the other one should beat each of them with his cards.

A card beats another one if both cards have similar suits and the first card has a higher rank then the second one. Besides, a trump card can beat any non-trump card whatever the cards' ranks are. In all other cases you can not beat the second card with the first one.

You are given the trump suit and two different cards. Determine whether the first one beats the second one or not.

Input
The first line contains the tramp suit. It is "S", "H", "D" or "C".

The second line contains the description of the two different cards. Each card is described by one word consisting of two symbols. The first symbol stands for the rank ("6", "7", "8", "9", "T", "J", "Q", "K" and "A"), and the second one stands for the suit ("S", "H", "D" and "C").

Output
Print "YES" (without the quotes) if the first cards beats the second one. Otherwise, print "NO" (also without the quotes).

Sample test(s)

input
H QH 9S
output
YES

input
S 8D 6D
output
YES

input
C 7H AS
output
NO

B. Choosing Laptop

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya is choosing a laptop. The shop has n laptops to all tastes.

Vasya is interested in the following properties: processor speed, ram and hdd. Vasya is a programmer and not a gamer which is why he is not interested in all other properties.

If all three properties of a laptop are **strictly less** than those properties of some other laptop, then the first laptop is considered outdated by Vasya. Among all laptops Vasya does not consider outdated, he chooses the cheapest one.

There are very many laptops, which is why Vasya decided to write a program that chooses the suitable laptop. However, Vasya doesn't have his own laptop yet and he asks you to help him.

Input

The first line contains number n ($1 \le n \le 100$).

Then follow n lines. Each describes a laptop as $speed\ ram\ hdd\ cost$. Besides,

- $speed$, ram, hdd and $cost$ are integers
- $1000 \le speed \le 4200$ is the processor's speed in megahertz
- $256 \le ram \le 4096$ the RAM volume in megabytes
- $1 \le hdd \le 500$ is the HDD in gigabytes
- $100 \le cost \le 1000$ is price in tugriks

All laptops have different prices.

Output

Print a single number — the number of a laptop Vasya will choose. The laptops are numbered with positive integers from 1 to n in the order in which they are given in the input data.

Sample test(s)

input
5 2100 512 150 200 2000 2048 240 350 2300 1024 200 320 2500 2048 80 300 2000 512 180 150
output
4

Note

In the third sample Vasya considers the first and fifth laptops outdated as all of their properties cannot match those of the third laptop. The fourth one is the cheapest among the laptops that are left. Thus, Vasya chooses the fourth laptop.

C. Buns

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Lavrenty, a baker, is going to make several buns with stuffings and sell them.

Lavrenty has n grams of dough as well as m different stuffing types. The stuffing types are numerated from 1 to m. Lavrenty knows that he has a_i grams left of the i-th stuffing. It takes exactly b_i grams of stuffing i and c_i grams of dough to cook a bun with the i-th stuffing. Such bun can be sold for d_i tugriks.

Also he can make buns *without stuffings*. Each of such buns requires c_0 grams of dough and it can be sold for d_0 tugriks. So Lavrenty can cook any number of buns with different stuffings or without it unless he runs out of dough and the stuffings. Lavrenty throws away all excess material left after baking.

Find the maximum number of tugriks Lavrenty can earn.

Input

The first line contains 4 integers n, m, c_0 and d_0 ($1 \le n \le 1000$, $1 \le m \le 10$, $1 \le c_0, d_0 \le 100$). Each of the following m lines contains 4 integers. The i-th line contains numbers a_i, b_i, c_i and d_i ($1 \le a_i, b_i, c_i, d_i \le 100$).

Output

Print the only number — the maximum number of tugriks Lavrenty can earn.

Sample test(s)

input
10 2 2 1 7 3 2 100 12 3 1 10
output
241

input
100 1 25 50 15 5 20 10
output
200

Note

To get the maximum number of tugriks in the first sample, you need to cook 2 buns with stuffing 1, 4 buns with stuffing 2 and a bun without any stuffing.

In the second sample Lavrenty should cook 4 buns without stuffings.

D. Treasure Island

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Our brave travelers reached an island where pirates had buried treasure. However as the ship was about to moor, the captain found out that some rat ate a piece of the treasure map.

The treasure map can be represented as a rectangle $n \times m$ in size. Each cell stands for an islands' square (the square's side length equals to a mile). Some cells stand for the sea and they are impenetrable. All other cells are penetrable (i.e. available) and some of them contain local sights. For example, the large tree on the hills or the cave in the rocks.

Besides, the map also has a set of k instructions. Each instruction is in the following form:

"Walk n miles in the y direction"

The possible directions are: north, south, east, and west. If you follow these instructions carefully (you should fulfill all of them, one by one) then you should reach exactly the place where treasures are buried.

Unfortunately the captain doesn't know the place where to start fulfilling the instructions — as that very piece of the map was lost. But the captain very well remembers that the place contained some local sight. Besides, the captain knows that the whole way goes through the island's penetrable squares.

The captain wants to know which sights are worth checking. He asks you to help him with that.

Input

The first line contains two integers n and m ($3 \le n, m \le 1000$).

Then follow n lines containing m integers each — the island map's description. "`#`" stands for the sea. It is guaranteed that all cells along the rectangle's perimeter are the sea. "`.`" stands for a penetrable square without any sights and the sights are marked with uppercase Latin letters from "`A`" to "`Z`". Not all alphabet letters can be used. However, it is guaranteed that at least one of them is present on the map. All local sights are marked by different letters.

The next line contains number k ($1 \le k \le 10^5$), after which k lines follow. Each line describes an instruction. Each instruction possesses the form "dir len", where dir stands for the direction and len stands for the length of the way to walk. dir can take values "`N`", "`S`", "`W`" and "`E`" for North, South, West and East correspondingly. At that, north is to the top, South is to the bottom, west is to the left and east is to the right. len is an integer from 1 to 1000.

Output

Print all local sights that satisfy to the instructions as a string without any separators in the alphabetical order. If no sight fits, print "`no solution`" without the quotes.

Sample test(s)

input

```
6 10
##########
#K#..#####
#.#..##.##
#..L.#...#
###D###A.#
##########
4
N 2
S 1
E 1
W 2
```

output

```
AD
```

input

```
3 4
####
#.A#
####
2
W 1
N 2
```

output

```
no solution
```

E. Space Rescuers

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Galaxy contains n planets, there are many different living creatures inhabiting each planet. And each creature can get into troubles! Space rescuers know it perfectly well and they are always ready to help anyone who really needs help. All you need to do is call for them.

Now the space rescuers plan to build the largest in the history of the Galaxy rescue station; however, the rescue station's location is yet to be determined. As some cases are real emergencies, the rescuers want to find such a point in the Galaxy from which it would be possible to get to the remotest planet in the minimum possible time. In other words, the rescuers need such point in the space that the distance between it and the planet remotest from it was minimal (if we compare this point with all other possible points in the space). Unfortunately, the rescuers can't sole this problem.

As the planets are quite remote from each other, they can be considered as points in Euclidean three-dimensional space. The distance between points (x_i, y_i, z_i) and (x_j, y_j, z_j) can be calculated by the formula $p = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$. The rescue station can be positioned in any point in the space. It can also coincide with some planet.

Galaxy is in danger! Save the space rescuers and find the required point for them.

Input

The first line of the input file contains integer n — the number of planets ($1 \le N \le 100$). Each of the following n lines contains information about the planets. The i-th line contains three integers x_i, y_i, z_i — the coordinates of the i-th planet ($-10^4 \le x_i, y_i, z_i \le 10^4$, $1 \le i \le n$). No two planets coincide.

Output

Print on the first line of the output file three space-separated real numbers x_0, y_0, z_0 — the coordinates for the future base. If there are several solutions, you are allowed to print any of them. The answer will be accepted if the distance from this point to the remotest planet will differ from the juries' variant in no more than 10^{-6} in absolute or relative value.

Sample test(s)

input
5 5 0 0 -5 0 0 0 3 4 4 -3 0 2 2 -2
output
0.000 0.000 0.000