challenge24

9th BME International 24-hour Programming Contest

Electronic Contest Problem Set

2009. 02. 28.

www.challenge24.org

MŰEGYETEM 1782

mave Electrical Engineering Students' Hungarian Association www.eestec.hu

Simonyi Károly SZAKKOLÉGIUM

Sponsors

Diamond grade sponsor

Nav N Go

Golden grade sponsor

Morgan Stanley

Silver grade sponsors

MTA SZTAKI
MAGYAR TUDOMÁNYOS AKADÉMIA
SZÁMÍTÁSTECHNIKAI ÉS
AUTOMATIZÁLÁSI KUTATÓINTÉZET

Professional organizations

NJSZT hte IEEE FDE

Press

Impulzus m•ha

Problems created by

Budapest University of Technology and Economics
Department of Automation and Applied Informatics

MŰEGYETEM 1782

http://www.aut.bme.hu/

iit
Department of Control Engineering and Information Technology

http://www.iit.bme.hu/

as Gergely Mezei, Andras Volford, Peter Vrana, David Borsos, Laszlo Szecsi, Peter Siroki, Peter Horvath, Jozsef Meszaros

Introduction, general information

Welcome to the Electronic Qualifying round of the 9th BME International 24-hour Programming Contest!

This year the EC contains 9 algorithmic problems, and you will have exactly 5 hours to solve them. The inputs of the problems can be found in a zip file that you have probably downloaded from the website already. This year each problem have different number of test cases, these will be indicated at their description. You can use any programming language to create the output files for the problems. We are interested only in the output files, you don't need to upload the source codes of the programs that solved them. Once you are done, you can upload these files via the EC submission site. Your solutions will be evaluated on-line.

Be quick about uploading the output files, because the scores awarded for every output file decrease with time Uploading it just before the end of the contest is worth **`70%`** of the maximum points achievable for the test case. During the contest its value decreases linearly with time. However you should also be careful with uploading solutions. Uploading an incorrect solution is worth **`-10`** points.

Please note that if you upload a solution for a test case that receives positive points, you won't be able to upload another solution to the same test case. Obviously there is no point in uploading another solution for an already solved testcase because you cannot achieve more points with it. Therefore the system will not register these uploads.

Note that points are awarded per output file and not per problem. If your solution only works for some of the input files, you will still be awarded points for the correct output files. A single output file however is either correct or wrong – partially correct output files are not worth any points. Achievable maximal points can differ per test case, these will be indicated at the problem descriptions.

Good luck and see you in the finals!

About the EC Submission site

You will be able to log in to the EC site with your registered email address and password. After login you will see your login email and team name in the top right corner. You can submit solutions by logging in with any of your team members' account. You can also use multiple accounts to submit the solutions. The site has five pages:

Announcements

This page will show the announcements made during the contest. These can be general information, or something connected to a problem (e.g. if an ambiguity is found in the description, we will correct it by posting an Announcement).

Team status

You can see your team's status here, with all your submissions and the points received for them. If you click on the points, you will be able to download the file you have submitted.

Submit solution

This is where you can post your solution files. You have to upload a single zip file that can contain multiple output files. The naming of the output files must strictly match the following format: **`X99.out`** where X is the problem's character code followed by one or two numbers, identifying the test case.

Standings

Here you can see the current standings of the contest. This will not be available in the last hour.

Maximal points

This page will show you how many points you can get if you submit the perfect solution for a test case at that time.

Contact

During the contest you will be able to contact us in various ways:

- We will of course read the email addresses challenge2009@challenge24.org, challenge24-2009@challenge24.org and the mailing list contest2009@challenge24.org
- Microsoft Windows Live Messenger: challenge24-2009@challenge24.org
- Skype: challenge24

Problem A: Oil lakes

The neighbouring countries of Atlardaia and Burgevya have been living in peace with each other for many generations. Now they are running short on oil, so they decided to look for new sources. Once upon a time their oil experts discovered a rectangular shaped area where they found a lot of „oil lakes" (these are big areas where oil can be found on the surface). Strangely, most of the lakes have a shape of a circle, but some of them looks like as if they were two circles merged. The mission is to divide this area as fairly as they can. The oil experts could not make a decision, so they went home. Seven months later the Minister of Justice from Atlardaia and Burgevya arrived. They agreed on the following:

- the two countries divide this area into two parts
- the total surface of the oil lakes (single and merged) located on first part of this area must be the same as the total surface of the oil lakes (single and merged) located on second part.
- they are going to travel first class on their way home

Unfortunately the ministers were not so good at technical implementation, so they went home. Seven days later the engineers arrived. They agreed on the following:

- they divide the area with a straight wire netting fence , which connects the two opposite sites of this rectangular shaped area (as they are really lazy and it is much easier to build a straight fence)
- the intersection of the fence and the oil lakes is prohibited

Seven hours later the Prime Minister of Atlardaia and Burgevya signed the agreement.

Your task is to find the end points of the fence. If your division is far away from the ideal division state Atlardaia and Burgevya will use their A-bomb, and the era of peace will be over. Be careful and precise!

Input specification:

The first and second line contains one integer pair separated by a space. These pairs are the coordinates of the new area (left upper and bottom right)

The next line contains a single integer, **N** the number of lakes. The following **N** lines contain the **X** and **Y** coordinates of the lakes' center and **R** (radius, float, using four decimal places) separated by spaces

Output specification:

The first and second line contain the **X** and **Y** (integer) coordinates of the wall's start & end point.

Example input:

```
0 0
100 100
4
18 69 8.0420
38 64 8.0000
71 18 8.0420
87 51 8.0000
```

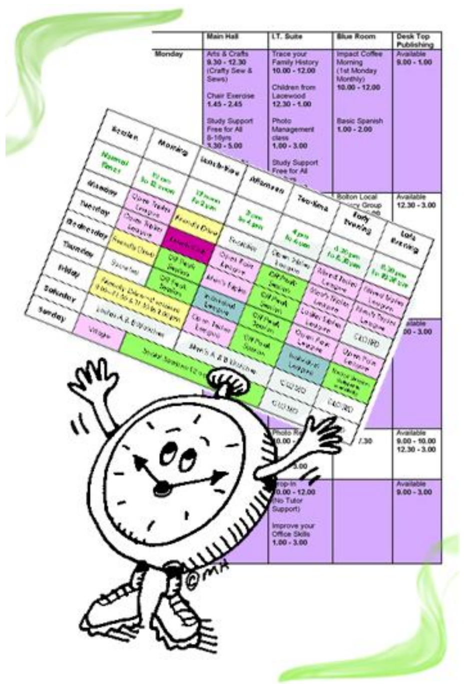
Example output:

```
0 0
100 100
```

Test case information

Number of test cases:	10
Maximal points:	100 per test case

Problem B: Course schedule



Every semester, the administrators of BME struggle a lot to make a good schedule for the students. Now you have a chance to help them. Given the list and length of the different courses, the professors teaching them and the available classrooms, you have to find a valid schedule. The goal is to make it possible for everybody to leave the university early (and do something more interesting), so we ask you to make the last course of the day end as soon as possible.

Some restrictions about the schedule:

- One professor can teach multiple courses
- One course can be taught by multiple professors
- One professor can teach only one course at the same time
- When a course starts, every professor teaching it must be available
- Only one course can be held in a classrom at the same time
- If a course is started it must be finished before another course can be started in the same classroom
- The classrooms are completely identical
- The start time of the scedule is 0
- When a course ends in a room, the next one can start immediately after it (so if there is a 4-long course starting at 0 in a room; another course can be scheduled in the same room starting at 4)

Input specification

The first line of the input contains three integers: **C,P,R** and **L** where **C(0<C<10000)** is the number of courses to schedule, **P(0<P<=50000)** is the number of professors, **R(0<R<=500)** is the number of available classrooms and **L** is the time limit (the last course must be finished by this time). In each of the following **C** lines there are several integers: the first is the identifier of the course, the next is the length of the it. The remaining numbers are the identifiers of the professors who are required for this course.

Output specification

If you think the problem cannot be solved on the given data (there is no valid schedule that fits in the time limit), the output must be the string „**IMPOSSIBLE**”. In any other case the output will contain a schedule of the courses. Each line of the output must be three integers, the first is the start time of the course, the second is the ID of the room the course is scheduled (**0<=ID<R** specified in the input), the third is the course ID.

Example input:

```
4 4 3 6
0 1 1 2
1 2 1 3
2 3 4
3 4 3
```

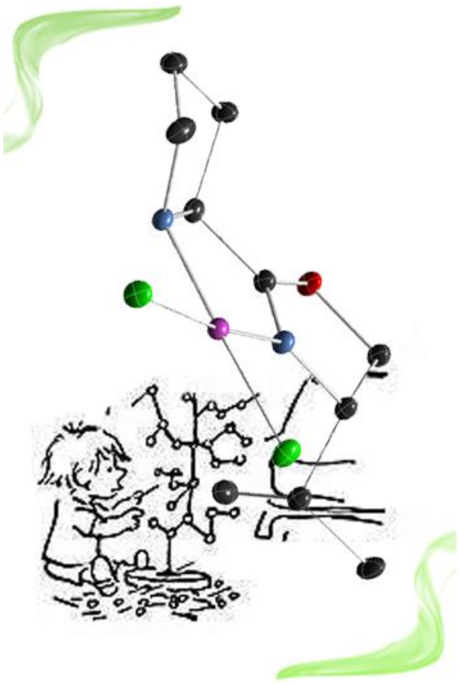
Example output:

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

Test case information

Number of test cases:	10
Maximal points:	100 per test case

Problem C: Quantum circuits



Your task is to calculate the result of a sequence of elementary quantum operations on a register of qubits which are described by square matrices acting from the left on complex vectors. The state space of **n** qubits is the space of 2^n-tuples of complex numbers up to a constant multiplier - nonzero multiples of a vector describe the same state. When we manipulate different qubits separately, the corresponding matrix is the tensor (or Kronecker) product of **2 × 2** matrices.

For example, if we have two qubits and we act with **A** on the first one and with **B** on the second one, then the full process is described by **B⊗A**:

$$A = \begin{bmatrix} a11 & a12 \\ a21 & a22 \end{bmatrix} \qquad B = \begin{bmatrix} b11 & b12 \\ b21 & b22 \end{bmatrix}$$

$$B \otimes A = \begin{bmatrix} b_{11}a_{11} & b_{11}a_{12} & b_{12}a_{11} & b_{12}a_{12} \\ b_{11}a_{21} & b_{11}a_{22} & b_{12}a_{21} & b_{12}a_{22} \\ b_{21}a_{11} & b_{21}a_{12} & b_{22}a_{11} & b_{22}a_{12} \\ b_{21}a_{21} & b_{21}a_{22} & b_{22}a_{21} & b_{22}a_{22} \end{bmatrix}$$

Of course doing nothing is described by the identity matrix **I** so when we act only on the **k**th of **n** qubits with the operation **A** then the corresponding matrix is:

$$\underbrace{I \otimes \ldots \otimes I}_{n-k} \otimes A \otimes \underbrace{I \otimes \ldots \otimes I}_{k-1}$$

which will be denoted by $\mathbf{A_k}$.

For our purposes we need the following seven **2 × 2** matrices:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad Y = i\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \qquad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad P = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \qquad Q = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

Of these the **H , X , Y** and **Z** gates can act on qubits and the rest is needed toexpress another gate not acting on separate qubits but describing interaction between two. The matrix of a **CNOT** (controlled not) gate with control qubit **i** and target qubit **j** is $\mathbf{CNOT_{i,j}=P_i+Q_iX_j}$. (The classical counterpart of this negates the **j**th bit if the **i**th is **1**).

The result of the subsequent operations is the usual product of the corresponding matrices: the leftmost being the last and the rightmost being the first operation. Moreover since multiples of a vector describe the same state, you only have to determine the product matrix up to scalar factor.

Input specification

The first row of the input consists of two integers. The first one is the number n of qubits (**1≤n≤10**) while the second is the number k of elementary operations (**1≤k≤5000**). The following k rows describe the operations as follows. The first character is one of the letters **X,Y,Z,H** and **C**. In the first four cases a single integer number follows separated by a space which is the index of the qubit which is acted on by the matrix denoted above by the same letter (e.g. **X 4** means X_4). If the row begins with a **C** then it is followed by two integers **i** and **j**, and in this case the matrix is $\mathbf{CNOT}_{i,j}$.

Output specifiaction

The output is the MD5 hash of the result matrix, as if the elements were enumerated first from left to right then from top to bottom, and each cell is a single signed byte. The digest must be printed in lowercase hexadecimal format, without spaces, two characters representing a single byte. So the stream representation of the matrix

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

would be a 4 byte binary buffer containing **00 FF 01 00**, and the digest string output would be

b0fed2ef99f98eec64dc8a3e8ae23e88

Example input

```
3 5
H 3
H 2
C 2 1
C 3 1
Z 1
```

Example output

Result matrix (don't upload this)

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

Result hash (upload this)

```
946fd1fc18af26ebb56562e0d97a0692
```

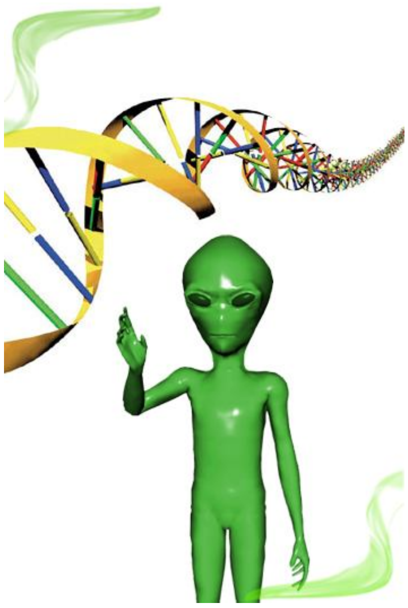
Output hasher tool

We have included in the zip file containing the inputs a simple tool written in Java with source code that calculates the hash. It is a command line tool its first parameter is the name of the file that contains the matrix, the second is the name of the file that will contain the hash value (this is the file that you need to upload. Please do not upload the result matrices)

Test case information

Number of test cases:	10
Maximal points:	100 per test case

Problem D: Alien DNA



In the distant future deep space exploration has become reality. Given the prosperity and plentiness of resources the human race has sent millions of probes to millions of solar systems. Finally a probe found something: a planet filled with alien animals. The probe took DNA samples of each and every animal for further analysis. The alien DNA chain is a sequence of genes, there are 26 types of alien genes represented by the letters from `'a'` to `'z'`. The whole DNA chain is too long to be analyzed, so the DNA chains were cut into sub-sequences with the latest technology: instead of simply cutting, a couple of genes at the beginning of each sub-sequence (but not all of them) was copied and attached to the end of the previous sub-sequence. The cuts and copies are made with care, so the sequences can be traced back by taking the longest possible overlap into account (if the longest possible overlap is the entire sub-sequence it wasn't the result of a cut so those sub-sequences can't be part of the same final sequence). The order of the sub-sequences are preserved. However sub-sequences of other DNA chains may be mixed in the input data. It is also true that the DNA chain that contains the most of the sub-sequences and can be put together from the sub-sequences are real chains.

Input specification

The first line of the input contains an integer **`N`**: the number of sub-sequences **`(1..2097152)`**. This is followed by **`N`** lines each containing a string: a sub-sequence of a DNA chain. A sub-sequence is represented by a series of lowercase letters. The sub-sequences are never longer than 16 characters.

Output specification

The output must contain one line: the length of the analyzed DNA chain, e.g. the longest possible chain that could be assembled from the input.

Example input

```
8
kocorog
retkars
gordivl
rsetko
roghgfet
etokrerf
fetbgaer
tkocge
```

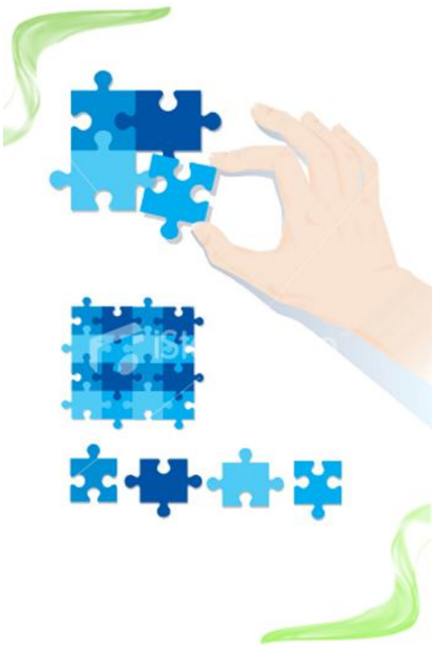
Example output

```
4
```

Test case information

Number of test cases:	10
Maximal points:	100 per test case

Problem E: John’s Puzzle



Before his death, the famous scientist John Harsaw has created a series of puzzles as an encoded explanation of his latest research results. The original purpose of these puzzles were to keep his secrets from unauthorized people. However, John has died before he could publish any hints how to solve or interpret these puzzles. There are many scientists who believe that the solution of the puzzles is the key to several global problems of mankind.

You are well known as one of the best puzzle solvers in the world, your task is to help the scientist by giving a solution to all of John’s puzzles.

Input specification

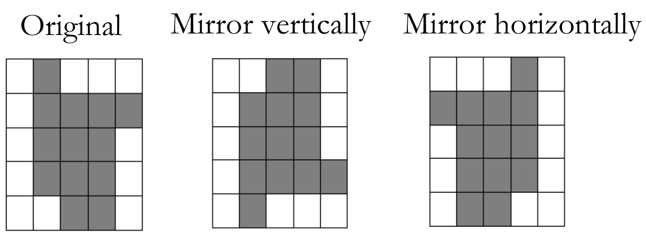
Each puzzle has a frame and several puzzle pieces. Each puzzle (puzzle frame and puzzle piece) can be divided into rectangular cells, no irregular shapes are allowed. The input consists of the definition of the frame and the pieces.

The scientist are not familiar with programming, thus, they have created a quite simple format: the definitions are described in a text file. In both cases (frame/pieces) filled cells are marked with `'#'`, empty cells are marked with space. The first row of the file contains two integers (the width and height of the puzzle) separated with comma. The next few lines contains the definition of the frame (one character for each cell) and an empty line after the frame definition. The structure definition is followed by the definition of the pieces. Each piece can fit in a **`5x5`** cell area. Pieces are defined in five lines, one character for each cell. After the definition of each piece, there is an empty line.

Output specification

You have to solve the puzzle. You can apply five transformations on the pieces (besides moving) before placing them into the frame:

0. None
1. Rotate clockwise 90°
2. Rotate clockwise 180°
3. Rotate clockwise 270°
4. Mirror horizontally
5. Mirror vertically



Your task is done, when the frame is completely filled with the pieces and no pieces overlap each other. You have to submit the solution as follows: for each puzzle piece you have to send the (i) the sequence number of the piece in the input (integer, the sequence number of the first element is 1.), (ii / iii) the x and y position of the piece (two integers, top left corner of the **`5x5`** field, 0-based), (iv) the transformations you applied on the piece (integers separated with comma, the series is enclosed by parentheses, **`'(0)'`** if no transformation was applied). The four data of the piece are separated with comma. In the output file, the data of puzzle pieces are separated by the new line character (each puzzle piece is described in exactly one line).

Example Input

```
11,11
###########
###    ## #
#         #
##       ##
#        ##
#         #
#        ##
#        ##
#         #
####   ####
###########

 # ##
 ###
#####
#####
# ##

 ###
####
 ###
 ####
  ###

 #  #
#####
 ###
 ###
 #

##
 ####
####
####
#  #
```

Example Output

Solution (don’t upload this)

```
###########
###########
###########
###########
###########
###########
###########
###########
###########
###########
###########
```

Output (upload this)

```
1,1,1,(1,4)
2,1,5,(1,4)
3,5,1,(0)
4,5,5,(1,4)
```

Test case information

Number of test cases:	8
Maximal points:	100 per test cases 1-6 and 200 per test cases 7-8

Problem F: Chemical system

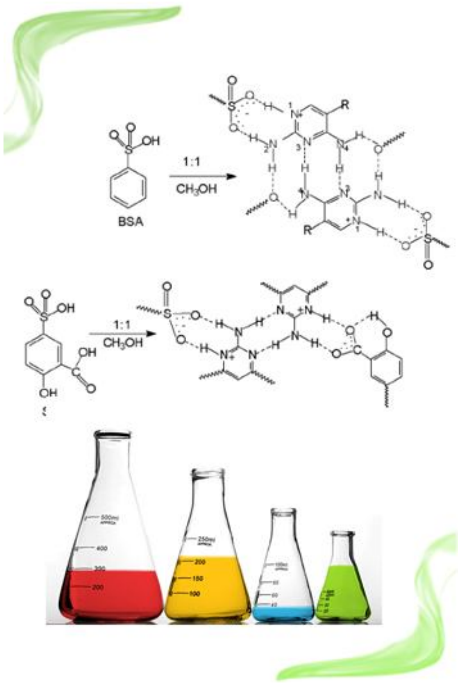
Given a chemical system with simple reactions like:

$A \to X$

$B + X \to Y$

$Y + 2X \to 3X$

$X \to$



These equations describe the transformation of different chemicals. The reactants are on the left side of the reactions and the products are on the right. E.g in the first reaction the chemical **A** is continuously converted to chemical **X** and in the second reaction the **B** and **X** chemicals are converted to **Y** chemical. Moreover, all the reactions takes place in the whole reaction medium, they are not separated in time or space.

Of course the reaction rate of each simple reaction depends on the number of chemicals available in reaction medium. The reactants can generate products if they are in the same place at the same time. Generally, the chemicals are not counted one by one, but measured by its concentration. So one can derive that the reactions depend on the concentrations of each reactant chemical. Taking the first reaction as an example, the reaction rate (which describes how fast a reaction takes place, how fast the chemical **A** consumed and how fast the chemical **X** generated) depends on the concentration of **A** chemical which we denote as **a**. But be careful in taking the third reaction as an example the reaction rate for the third reaction is **y*x*x** as we need one **Y** and two **X** chemical to be in the same time in the same place, where **x** and **y** are the concentrations of chemicals **X** and **Y**. This reaction rate describes how fast the chemical **Y** consumed and how fast the chemical **X** is produced. (The chemical **X** is produced in this reaction as **2X** is consumed but **3X** is produced.)

Let **x**, **y**, **a** and **b** be the concentrations of **X**, **Y**, **A** and **B** chemicals respectively. For the sake of simplicity we assume that the concentrations of **A** and **B** are kept constant (they are continuously refilled) and we are interested in the time evolution of the chemical species **X** and **Y**.

Input specification:

The first line contains a single integer, **N** the number of reactions. The following **N** lines contain the reactions of chemicals. The next line contains the number of parameters. Followed by the parameter values in each line. The next line contains the number of initial concentration for **x** and **y**. Followed by the initial concentration values for **X** and **Y**.

Output specification:

Two double numbers with 6 decimal precision, **x** and **y**, separated by a space, that are the final concentrations of **X** and **Y**.

Example input:

```
4
A -> X
B + X -> Y
Y + 2X -> 3X
X ->
a=1.5
b=2.7
3.0 3.0
```

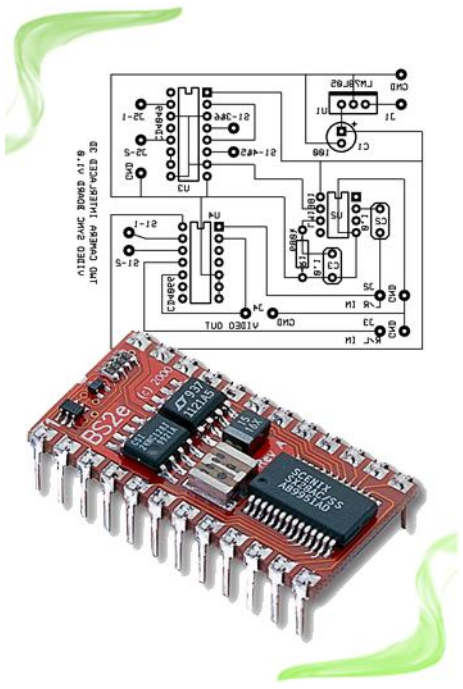
Example output:

```
1.500000 1.800000
```

Test case information

Number of test cases:	5
Maximal points:	200 per test case

Problem G: Erroneous Circuits



In the main plant of American Circuit Inc. the engineers have noticed that sometimes the circuits made by the company cause serious errors. After applying an extensive search for the reason, they have found that certain combination of components lead to these errors. They succeeded in finding all of these unreliable combinations and they suggested a correction for each. Note that this replacement was not an easy task, since the functionality of the circuits was not allowed to change.

Your task is to help the company and correct erroneous circuits. You receive the structure of an existing circuit, the definition of the erroneous configurations and its replacement. You have to eliminate all components that can cause errors by applying the replacements suggested by the company. New components may be added to, and components may be removed from the configuration during the replacement. If a component is removed by the replacement, all of its connections must be removed as well.

Input specifiaction

The input contains the definition of the circuit, including

- The list of components in the form of `Type(0..50) - ID(0..100000)` pairs. The ID is the unique ID of the component, while the Type is the type code of the component. This information is the first line of the input file that is followed by an empty line.
- List of connections between the components, as ID pairs. Note that a certain pair of components can have more than one connections (parallel connections are allowed). This is the third line of the input file.

Furthermore, for each configuration, they define

- The erroneous configuration
  - The list of components in the configuration as `Type(0..50) - ID(0..100000)` pairs. (In one line, followed by an empty line)
  - List of connections between the components, as ID pairs. Note that a certain pair of components can have more than one connection. (In one line)
- The replacement
  - The list of components in the replacement as `Type(0..50) - ID(0..100000)` pairs. Note that we use the same ID for nodes, which are preserved by the replacement, but we can add new nodes as well. (In one line, followed by an empty line)
  - List of connections between the components in the replacement, as ID pairs. Note that a certain pair of components can have more than one connection. (In one line)

Output specification

The number of replacements followed by the first configuration instance found in the circuit (in the form of configuration item ID – circuit item ID pairs, order by the configuration item IDs, ascending). You do not need to upload the result model itself and you do not need to send us more than one match (the first one will be enough). If the replacements cannot be applied for some reason, you have to define the number of replacements as `-1`. If the replacement can be applied in more than one ways, you are free to choose the sequence of application. Note however that the result model is not allowed to contain the configuration and only the defined replacements are valid.

Example Input

```
GI0.in
1 - 1;1 - 2;1 - 3;2 - 4;2 - 5;3 - 6
1 - 4;2 - 4;3 - 5;4 - 6;5 - 6

GC0.in
1 - 1;2 - 2;3 - 3
1 - 2;2 - 3

GR0.in
1 - 1;4 - 2;3 - 3
1 - 2;2 - 3
```

Example Output

```
G0.out
2
1-3;2-5;3-6
```

Test case information

The test cases are independent from each other.

Number of test cases:	9
Maximal points:	100 per test cases 1-8 and 200 per test cases 9

Problem H: Gas pipe

As a response to the current gas crisis, the government of country Ugivia decided to build a new network of gas pipes in order to provide the necessary gas supply for the citizens. Each city in the country has a „gas value" that tells how much gas is required by the inhabitants of that city every day. If this value is a negative number that means the city has a surplus of gas that can be given to the other cities of the country. Gas pipes can be built between most of the cities, but not all of them: in some cases it would be extremely expensive, in others the local environmental activists would surely witheld the construction for several hundred years. Of course constructing a gas pipe still has different costs between the different cities. The cost of building also depends on amount of gas transported in the pipe, because if you need to transport more, you will need larger pipes. If the pipe cost between two cities is **`C`** and the amount of transported gas is P than the Total Cost of building is **`TC=C*P`**. (in other words: **`C`** is the cost to transport 1 unit of gas between the cities). If you can build a pipe between two cities, you can use it to transport gas into both directions.

Your task is to create a plan of the gas pipe system that fulfills the requirements of the citizens and as cheap as possible. (You can assume that there is always more gas available than the requirements).

Input specification:

The first line of the input containts two integers **`N(0<=N<=10000)`** and **`E(0<=E<=16000)`**. **`N`** is the number of the cities in the country, **`E`** is the number of specified connections between them. In the following **`N`** lines there are two integers **`I`** and **`R`** where I is the identifier of the city and **`R`** is the required amount of gas by that city (**`0<I<=N`** and **`-1000<=R<=1000`**).

The following **`E`** lines describe the connections between the cities. Each line contains three integers: **`A B C`**, where **`A`** and **`B`** are the identifiers of two cities and **`C`** is the cost to build a gas pipe capable of transporting 1 unit of gas.

Output specification:

The output containts a single number: **`TC`** the total cost of the gas network.

Example input:

```
5 6
1 10
2 12
3 -15
4 3
5 -10
1 2 1
1 3 4
1 4 2
1 5 3
3 4 5
4 5 6
```

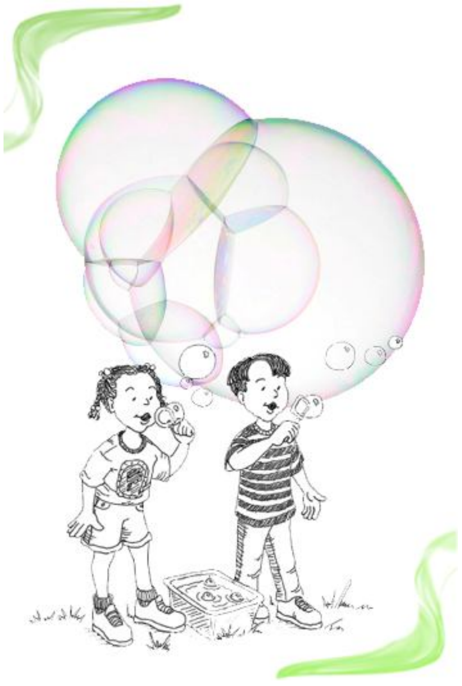
Example output:

```
105
```

Test case information

Number of test cases:	10
Maximal points:	100 per test case

Problem I: Soap bubbles



The industrious workers of the Fire Ant nation supply the nest of their people with gathered plants and seeds. However, this day they have found the usual path blocked by huge blobs of soap foam! It must have been the doing of those menacing human children blowing soap bubbles here! Luckily, the Red Army of Her Majesty the Queen of Fire Ants is ready to solve the problem. This is a modern army, equipped with laser cannons, and operates with surgical precision, based on scientific calculations.

Army officers installed **n** laser cannons, at 3D coordinates $o_0, \ldots, o_{n-1}$ pointing into directions $d_0, \ldots, d_{n-1}$. Workers (who had nothing better to do anyway, the path being blocked) measured all soap bubbles. They counted **m** bubbles, and they found that they all have radius **r**, and their centers are at 3D coordinates $c_0, \ldots, c_{m-1}$. Where two or more bubbles would overlap, they are separated by infinitely thin planar films of soap, positioned at equal distance to the two bubble centers. A point in space **x** is within the i^{th} bubble (with center c_i, radius **r**), if $|x-c_i|<r$ and for every index $j \in \{0, \ldots, m-1\} \setminus \{i\}$; $|x-c_i|<|x-c_j|$. To put that in plain words, every point belongs to the bubble whose center is the nearest to it, or to no bubble if it is farther away from all centers than **r**.

The laser cannon can penetrate **10** layers of soap film. (Well, what can you expect from an ant-sized cannon?) It is never reflected or refracted.

An effective military operation requires exact planning. Therefore, the commander needs to know exactly which bubbles are pierced, if all cannons are switched on at the same time. Intersection may happen both on the spherical surface of bubbles, or on planar soap films between bubble cells. A bubble is popped, if one of the first ten intersections of any ray is on its surface. Write a program that returns the indices of the popped bubbles.

Input specification:

The first line of the input contains the number of cannons **n** as an integer. Then following **n** lines give cannon data: the first triplet of floating point numbers give the **x,y,z** cannon coordinates, and the second triplet gives the laser ray directions. Direction values are not guaranteed to be normalized. Every coordinate is a float, correct to four places of decimals.

After the cannon data, bubble radius **r** is given as a floating point number. In the next line, the number of bubbles **m** follows. Finally, the last **m** lines contain coordinate triplets of bubble centers. The bubbles are indexed from **0** to **m-1** in the order they appear here.

Output specification:

Every line contains the indices of bubbles popped by a laser cannon. Lines appear in the same order of cannons as in the input. A line contains at most **10** integers, separated by spaces. Bubbles must be listed in the order the laser hits them. If a cannon does not pop any bubble, its line should be empty.

Example input:

This input defines 2 laser cannons, one located at position **`[0,0,0],`** firing in direction **`[1,0,0]`**, an another at **`[11,6,0]`** firing in direction **`[0,-1,0]`**. The foam consists of **`2`** bubbles of radius **`0.8`**. The bubble centers are **`[10,0,0]`** and **`[11,0,0]`**, so they overlap.

```
2
0.0 0.0 0.0 1.0 0.0 0.0
11.0 6.0 0.0 0.0 -1.0 0.0
0.8
2
10.0 0.0 0.0
11.0 0.0 0.0
```

Example output:

The first ray pierces both bubbles, the second ray pierces only one.

```
0 1
1
```

Test case information

Number of test cases:	10
Maximal points:	100 per test case