FAU Winter Contest 2017

January 28th

acm International Collegiate Programming Contest

FAU FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Problems

A Heizlüfterparty
B Cats and Yarn
C Memory with Memes
D Movie *(easy)*
E Feeding the Cats
F Oneko *(easy)*
G The Joy of Cats
H Pet Rivalry *(very easy)*
I The Bridge
J Cat Identification
K Play Of The Game
L Hyperloop
M Consequences

Sponsored by

accenture
High performance. Delivered.

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

This page is intentionally left (almost) blank.

Problem A: Heizlüfterparty

Nothing beats a warm cat basket. Especially if the country you are staying in has very cold nights, as for instance this year, when the ICPC (*Incredibly Collegiate Purring Competition*) is carried out in such a country. The three members of team FAUCat decide that the best way to provide a warm cat basket is to have a heater-party, or as they call it "Heizlüfterparty". The hotel they are staying at already provides a heater which radiates warmth equally in all directions. The only restriction is that the heater cannot be moved away from the wall it has been assigned to by the hotel. Other than that, it can be moved freely along the wall.

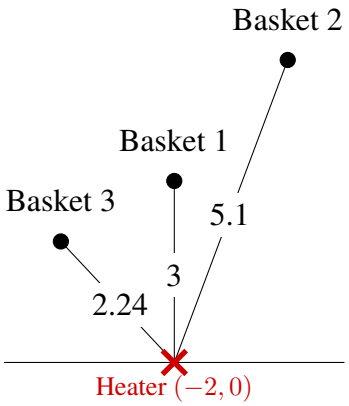

Figure A.1: Sample Output 2

The first course of action is to fully crank up the heat. Still the heater doesn't fully satisfy everyone's desire for warmth, so the team decides to move the heater around. Some attempts later, the members start to question whether it is possible to place the heater in a position that satisfies everyone. After all, every cat has different preferences. While each of the members may have their own ideas about how far from their basket the heater can be to still be warm enough, they all agree that having the heater closer to their basket is never wrong and always appreciated. After all, the night is supposed to be HOT HOT HOT!

Input

The input consists of three lines containing three integers x, y and r ($-1\,000 \leq x \leq 1\,000$, $0 < y < r \leq 1\,000$) each. x and y describe the position of a cat basket and r describes the maximum distance the heater may have from this very basket. Every basket has a unique position.

The wall the heater must never be moved away from, is always described by the function $f(x) = 0$ (the x-axis).

Output

Output `yes` if the heater can be placed so it is at most the described distance away from each cat basket. Otherwise `no`. You can safely assume that if the heater can be placed, then there is a position on the x-axis such that the heater can be placed on all positions within a distance of 10^{-6}.

Sample Input 1

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

Sample Output 1

```
yes
```

Sample Input 2

```
-2 3 5
-1 5 6
-3 2 4
```

Sample Output 2

```
yes
```

Sample Input 3

```
0 1 2
1 1 2
100 1 2
```

Sample Output 3

```
no
```

Problem B: Cats and Yarn

Your little kitten Fluffy discovered your grandma's basket of yarn. While you were out of the room, she started to play with it. Although Fluffy had a lot of fun, she soon got tangled up in the yarn and was unable to free herself.

You opened the door and found Fluffy in this predicament: sitting in the middle of the room and meowing desperately, waiting for you to free her. Unfortunately, the yarn tangled itself not only around Fluffy, but also around a lot of other items. While you try to approach Fluffy and pull some of the threads, you notice that you also pull on a lot of items on the other side of the room.

You want to help Fluffy get free from her self-inflicted mess as soon as possible and decide to cut some of the threads until Fluffy is able to move again. As a first step, you want to be able to somehow move through the room without pulling loads of items from the tables when touching the yarn. This means, for now your goal is to have at least two different items that are not directly or indirectly connected by the threads (i. e. when pulling on the first item, the second item does not move, as there is no path of threads between them).

As Fluffy spread a lot of different balls of yarn across the room, most of the connecting threads between two items differ in strength. You only have one old pair of worn-out scissors, and cutting the threads is quite fiddly and takes you some time. Thus, you try to minimize the sum of the strengths of the threads that you have to cut through.

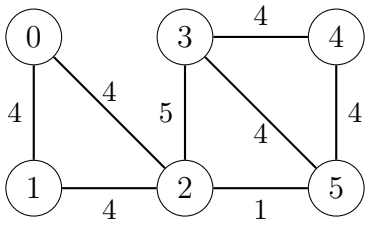

Figure B.1: The third sample input. The easiest way to separate at least two items is cutting through the threads between 2 and 3 and between 2 and 5, giving a total strength of 6.

Input

The input consists of:

- one line with two integers n and m ($2 \leq n \leq 50$, $0 \leq m \leq 1\,225$) denoting the number of items in the room and the number of threads connecting these items;
- m lines specifying the threads of yarn, each consisting of three integers a, b and c ($0 \leq a, b < n, 1 \leq c \leq 10^9$); a and b are the two connected items and c is the strength of the thread. All threads are distinct and no thread connects an item to itself.

Output

Print one line with one integer, the minimum sum of thread strengths to cut through.

Sample Input 1

```
2 1
0 1 5
```

Sample Output 1

```
5
```

Sample Input 2

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

Sample Output 2

```
3
```

Sample Input 3

```
6 8
0 1 4
1 2 4
2 0 4
3 4 4
4 5 4
5 3 4
2 3 5
2 5 1
```

Sample Output 3

```
6
```

Problem C: Memory with Memes

Peter loves playing *Memory with Memes* with his friends. *Memory with Memes* is a card game played with $2n$ cards where there are n pairs of cards that have matching cat memes on them. In the beginning of the game all cards are placed on the table with the cat memes facing down. In each round a player turns up two cards, one after the other (visible to everybody) and if they show the same cat meme they are both removed from the game and that player scores a point. Otherwise they are placed back face down. The game ends once all cards have been removed from the game.

Peter is very good at *Memory with Memes*, in fact he is so good that he always remembers every card that was turned up during the entire game. While this is good for his winning chances, it also means that none of his friends want to play with him anymore! Because of this, Peter has taken to playing on his own and tries to complete the game using as few rounds as possible. If he plays his cards right, what is the expected number of rounds needed to clear all the cards off the table? Remember that each round the two cards are turned up one at a time, so Peter picks the second card only after seeing the meme on the first.

Input

The input consists of one line with one integer n ($1 \leq n \leq 1\,000$), the number of pairs of cards.

Output

Output one line with one number, the expected number of rounds needed to complete the game when following a perfect strategy. Your output will be accepted if the absolute or relative error does not exceed 10^{-6}.

Sample Input 1

```
2
```

Sample Output 1

```
2.666666666667
```

Sample Input 2

```
4
```

Sample Output 2

```
5.923809523810
```

This page is intentionally left (almost) blank.

Problem D: Movie

Your friend Max is *the* up-and-coming star in the *YouTube* scene. At least that's what he keeps telling you and everybody who wants (or doesn't want) to listen. Last month he discovered the video *The entire bee movie but every time they say bee it gets faster*, which gathered more than 16 million views on the site – and now he wants to create his own version of that video.

The original video: https://youtu.be/E6iN6VTL7v8

"Surely", Max says, "if that guy can reach millions of views with such a simple video, then so can I." He goes on to explain that the video starts at normal speed, but whenever the word "bee" is said, the playback speed gets multiplied by a fixed factor s. Of course, Max cannot just copy the original video, so he has opted to use a film about cats and to have speedups happen for every occurrence of the word "cat" instead. ("The internet totally loves cats", he explains, "so that should get me a lot of clicks.")

He has already marked all occurrences of "cat" in several movies in his video editing software. To help him decide which movie to choose he has asked you to calculate how long the final edited videos would be.

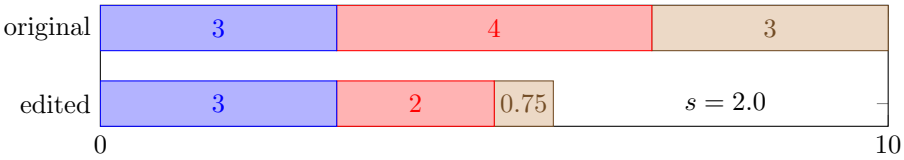

Figure D.1: Illustration of the first sample input. After 3 seconds the playback speed doubles; after 5 seconds the 7-second marker is reached and the speed doubles again.

Input

The input consists of:

- One line with three numbers:
  - an integer L ($1 \leq L \leq 100\,000$), the length of the movie in seconds;
  - a real number s ($1.1 \leq s \leq 3.0$), the speedup factor;
  - an integer n ($1 \leq n \leq 500$), the number of occurrences of the word "cat" in the movie.
- One line with n distinct integers $t_1, \ldots, t_n$, where $0 < t_i < L$ is the time in seconds from the beginning of the movie to the i^{th} occurrence of the word "cat". These times are given in increasing order.

Output

Output one line with one number, the final length of the edited video. Your output will be accepted if the absolute or relative error does not exceed 10^{-6}.

Sample Input 1

```
10 2.0 2
3 7
```

Sample Output 1

```
5.75
```

Sample Input 2

```
14 1.3 2
3 9
```

Sample Output 2

```
10.57396449704142
```

Sample Input 3

```
100 1.5 3
20 60 70
```

Sample Output 3

```
60.0
```

Problem E: Feeding the Cats

One of the cats calls for your attention.

Having a side job to make a little extra money is always appreciated, especially if the job description sounds easy. “Help wanted to feed my cats. – Ms Purr” – How hard can it be?

Turns out that it is a bit more complicated than it appeared. Firstly, Ms Purr has a lot of cats. Polite as you are, you listened to her listing off all their individual names, but after a short while you gave up and just numbered the cats from 0 to $n-1$. Secondly, cats can get really jealous if they see that other cats are fed before they are, so they might start stealing food. However, you noticed that there is a cat rank order. Cats of lower rank will never dare to steal food from cats of higher rank. Therefore, the only way to keep peace between the cats is to feed them in descending order of rank (conveniently, no two cats have the same rank).

In the past couple of days, while already performing your job, you made a large number of memos to remind yourself which cats must be fed before others. Every memo describes exactly two different cats and which of those has the higher rank. You know for sure that with those memos you can reconstruct the correct rank order and now you are wondering if you always have to bring all those heavy notes with you. Wouldn’t it be easier to just leave the unnecessary ones at home? You start by giving the memos consecutive indices from 0 to $m-1$. Which memos can be removed without making it impossible to reconstruct the correct rank order?

Input

The input consists of:

- One line with two integers n and m ($1 \leq n \leq 100\,000, n-1 \leq m \leq 200\,000$), which describe the number of cats and the number of memos.
- m lines of the form a `>` b ($0 \leq a, b < n; a \neq b$), which means cat a is of higher rank than cat b. The memos are listed with ascending index, from memo 0 to memo $m-1$. It is guaranteed that every memo is unique and that the unique correct rank order can be reconstructed with those m memos.

Output

Output the indices of all the memos that are not needed to reconstruct the correct rank order. These indices should be printed in strictly ascending order, one per line. If no such index exists, print `meow` instead.

Sample Input 1

```
3 3
0 > 1
0 > 2
1 > 2
```

Sample Output 1

```
1
```

Sample Input 2

```
2 1
1 > 0
```

Sample Output 2

```
meow
```

Sample Input 3

```
6 8
1 > 5
0 > 3
4 > 3
1 > 3
2 > 3
5 > 4
0 > 1
2 > 0
```

Sample Output 3

```
1
3
4
```

Problem F: Oneko

Do you know oneko, the on-screen-cat? It is a cat running on your computer screen, chasing your mouse. Each time Lorenz is bored at work, he starts multiple onekos to have the entire group chase the mouse. After some time playing, he tries to let the onekos perform some tricks, like lining up in a straight line with the mouse. To form the line, he moves the mouse to the bottom left corner of the screen. You are given the coordinates of all onekos, can you write a program that checks if they are in one line with the mouse at $(0, 0)$?

Figure F.1: Sample Input 2

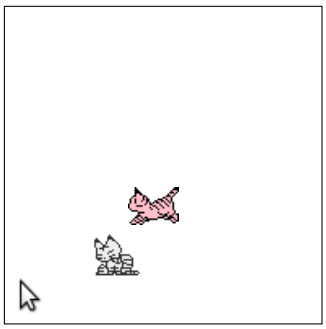

Figure F.2: Sample Input 3

Input

The input consists of:

- one line with one integer n ($1 \leq n \leq 1\,000$), the number of onekos;
- n lines, each with two integers x_i and y_i, where (x_i, y_i) is the position of the i^{th} oneko and $0 \leq x_i, y_i \leq 10\,000$.

Output

Output `yes` if all onekos and the mouse are on one line, `no` otherwise.

Sample Input 1

```
1
1 0
```

Sample Output 1

```
yes
```

Sample Input 2

```
3
2 2
10 10
11 11
```

Sample Output 2

```
yes
```

Sample Input 3

```
2
2 1
3 2
```

Sample Output 3

```
no
```

This page is intentionally left (almost) blank.

Problem G: The Joy of Cats

The book *Abstract and Concrete Cats: The Joy of cats* by Adámek, Herrlich, Strecker is full of definitions and interesting concepts, for example **Cat** or **CAT** on page 40.

In order not to lose track of all the definitions, the authors want to assemble an index using a *trie*. A trie is a tree data structure that can be used to store a set of strings. In the tree each node corresponds to a prefix of one of the strings, with the root node being the empty string. The edges go from shorter to longer prefixes so that for every node the labels on the path from the root spell out the corresponding prefix. An example can be seen below:

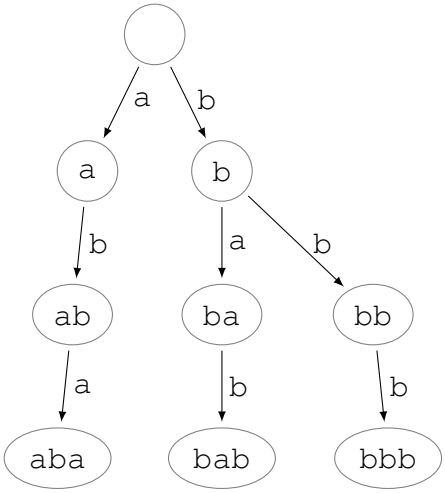

Figure G.1: An example trie built on the strings `aba`, `bab` and `bbb`, with 9 nodes.

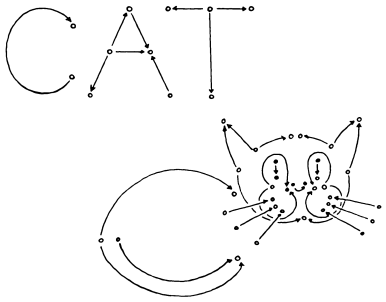

Figure G.2: An example illustration from *The Joy of Cats* (page 12)

The authors have coded a trie in their favourite programming language, but they are unsure whether their implementation is efficient enough. To have a good stress test for their code they are interested in test data that will cause as many trie nodes as possible to be allocated. They have asked you to come up with such test data.

More specifically, they know that the book defines up to n words, where each definition is not longer than m letters. Since most of the defined words are quite similar (like **cat** or **ccc**), your test data should only use the first k lowercase letters of the English alphabet. Given these constraints, find a set of strings that generates a trie of maximal size.

Input

The input consists of one line with three integers:

- n ($1 \leq n \leq 10\,000$), the number of words;
- m ($1 \leq m \leq 20$), the maximum length of each word;
- k ($1 \leq k \leq 26$), the size of the alphabet.

Output

On the first line, output the maximum number of nodes in a trie, followed by the number w ($0 \leq w \leq n$) of strings used in your test data. The following w lines should contain your test data, one string per line. The strings must be non-empty.

Sample Input 1

```
3 3 2
```

Sample Output 1

```
9 3
aba
bab
bbb
```

Sample Input 2

```
5 1 3
```

Sample Output 2

```
4 4
b
a
c
a
```

Problem H: Pet Rivalry

The dog doesn't stand a chance!

Bored while waiting for their owner to get back home, the cat and the dog decided to start a game of cards. They often argue about which one of them is the coolest pet in the world. In order to raise the stakes of the game, they settle to declare the winner *Pet of the Year*. Because it is winter and very cold, the cat (having no doubt she will win) suggests the loser also has to live outside until spring. For the rest of us, this game will settle all the disputes on the internet on whether cats or dogs are better.

Their game is very simple: they both draw one card and the winner is whoever has the card with higher value.

Seeing that the stakes of the game are so high and hoping to prove to your friend (who always bugs you with cat pictures) that dogs are better, you decide to help them determine who the winner is. After all, they are just a cat and a dog, and have no idea about numbers...

If I lose, I must sleep outside.

Input

The input consists of one line with two characters d and c, the cards drawn by the dog and the cat, respectively.

The cards are guaranteed to be distinct (they may only be a cat and a dog, but they don't need a human to figure out whether two cards are equal).

The possible card values (from lowest to highest) are:

`2`, `3`, `4`, `5`, `6`, `7`, `8`, `9`, `t` (for 10), `j` (for Jack), `q` (for Queen), `k` (for King) and `a` (for Ace).

Output

For each game output the winner: `cat` or `dog`.

Sample Input 1

```
5 6
```

Sample Output 1

```
cat
```

Sample Input 2

```
5 a
```

Sample Output 2

```
cat
```

Sample Input 3

```
6 5
```

Sample Output 3

```
dog
```

Sample Input 4

```
8 j
```

Sample Output 4

```
cat
```

Sample Input 5

```
q 7
```

Sample Output 5

```
dog
```

This page is intentionally left (almost) blank.

Problem I: The Bridge

The Furrlowship of the Ring is a group of only the bravest kittens from Middle-earth – but unfortunately not the fastest ones. On their way to Mordor, the Furrlowship of the Ring has to cross an old bridge in a dark mine. However, they cannot simply take the bridge altogether:

1. Due to its age, the bridge can carry at most b kilograms at once.
2. Since this happens in darkness, anyone crossing the bridge needs a torch to light the way to the other side.
3. They have only one torch.
4. If some group of members crosses the bridge together, then they are only as fast as the slowest of them.

Legolascat	**Gandalfcat**	**Frodocat**	**Gimlicat**
1 min, 10 kg	2 min, 10 kg	5 min, 10 kg	8 min, 10 kg

Figure I.1: The kittens in the first sample input

Input

The input consists of:

- one line with two integers n and b ($1 \leq n \leq 10, 1 \leq b \leq 1\,000$), denoting the number of group members and the weight the bridge can carry at most;
- one line with n integers $t_1, \ldots, t_n$, where $1 \leq t_i \leq 1\,000\,000$ is the time the i^{th} member needs to cross the bridge;
- one line with n integers $w_1, \ldots, w_n$, where $1 \leq w_i \leq 1\,000$ is the weight of the i^{th} member in kilograms.

Output

Print one line with one integer, the minimum time it takes to bring all group members to the other side of the bridge. If they can not make it, print `impossible`.

Sample Input 1

```
4 20
1 2 5 8
10 10 10 10
```

Sample Output 1

```
15
```

Sample Input 2

```
3 40
4 4 4
30 30 45
```

Sample Output 2

```
impossible
```

Sample Input 3

```
5 6
6 7 8 9 10
1 2 3 4 5
```

Sample Output 3

```
39
```

This page is intentionally left (almost) blank.

Problem J: Cat Identification

The *Identify Cats Personally Corporation* (ICPC) produces labels for cats with which you can uniquely identify any cat wearing one of their collars. All identification numbers use digits from a fixed set, so each produced package is built from the same ingredients: one collar and one digit-brick per available digit.

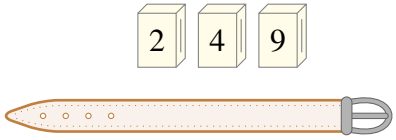

Figure J.1: Package contents for both sample inputs

The only difference between the packages is how they are assembled, namely the order in which the digit-bricks are put onto the collar. Of course the identification collars wouldn't be very useful if there were two with the same number on them, so ICPC guarantees that for each collar the order is unique.

The company distributes the packages in such a way that the collars are shipped out in descending order of identification number. Given that you were the i^{th} customer to buy a collar for your cat, which number is in your package?

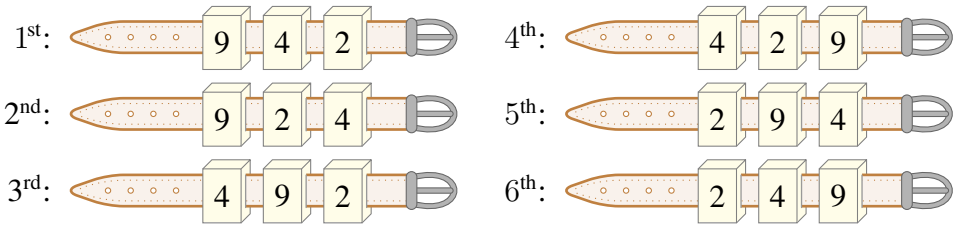

Figure J.2: The assembled packages

Input

The input consists of a string s and an integer i, where s specifies the digit-bricks used by the company and i ($i \geq 1$) is as described above. The digits in s will be between `0` and `9` inclusive, and no digit will appear more than once. You may safely assume that at least i collars have been produced.

Output

Print the cat identification number in the i^{th} package.

Sample Input 1

```
249 1
```

Sample Output 1

```
942
```

Sample Input 2

```
942 4
```

Sample Output 2

```
429
```

This page is intentionally left (almost) blank.

Problem K: Play Of The Game

You always knew: Hard work pays off. Now look at you: You are an important part of the development team of the highly anticipated game OVERCATCH. OVERCATCH is a team based First Person Shooter with cats fighting against the evil forces of dogs.

OverCATch character (by leah.c.andersen on instagram.com)

After every game of 30 minutes (1 800 000 milliseconds) the *play of the game* is shown. It is a replay of fixed length t milliseconds showing the best sequence of actions by one player during the game. Your task is to determine when the *play of the game* should start and which player should be shown.

For this purpose, the game creates a log file for each player, where all events involving that player are recorded. Each log entry includes a time stamp (the time from the start of the game until the event) and the score awarded to the player for that event. The scores may be zero or even negative depending on the player's performance – for instance, getting hit would result in a negative score.

You decided that the best way to determine the *play of the game* is to pick the player and starting time such that the sum of scores for all of his/her actions during the following t milliseconds is maximal. This is an important task and you need to be done with it today, so hurry up, because it's high noon already.

Input

The input consists of:

- one line with two integers p and t ($1 \leq p \leq 12, 1 \leq t \leq 40\,000$) where p is the number of players and t is the length of the *play of the game* in milliseconds.
- p blocks each describing one of the players:
  - One line with two strings and one integer e ($0 \leq e \leq 40\,000$), where the strings describe the player's unique name and the in-game-character they played as and e is the number of log entries for that player.
  - e lines with two integers t_i and s_i ($0 \leq t_i < 1\,800\,000, -10\,000 \leq s_i \leq 10\,000$) each, describing that at t_i the player got a score of s_i. The t_i are given in strictly increasing order.

All strings in the input are alphanumerical and at most 20 characters long.

Output

OverCATch character (by leah.c.andersen on instagram.com)

First output the name of the player followed by `as`, the name of the character and a colon. After that print an integer a ($0 \leq a \leq 1\,800\,000 - t$), the starting point of the *play of the game*. If a is the starting time, the *play of the game* will include all events with time stamp between a and $a + t - 1$ inclusive.

If there is more than one optimal answer, any will be accepted.

Explanation for sample 1

Here t is $12\,000$ ms. Player `BornToMeow` got the *play of the game*, because starting at time $22\,000$ two events are included for a total score of 52. Note that it is not better to start at time $10\,000$ instead, because then the score would only be 50 (the next event is 1 ms too late). The best sum of scores player `Furtastic` got was 51.

Sample Input 1

```
2 12000
BornToMeow Junkcat 3
10000 50
22000 5
25000 47
Furtastic Symmeowtra 2
20000 30
30000 21
```

Sample Output 1

```
BornToMeow as Junkcat: 22000
```

Sample Input 2

```
5 10000
TheLegend27 Nyanzo 5
12300 1000
12400 1000
12500 1000
12600 1000
13000 200
PurrMan Kittenji 1
12300 -1000
WubU Meowcy 1
12400 -1000
RatsOrCats MeowCree 1
12500 -1000
RawMeat Reapurr 1
12600 -1000
```

Sample Output 2

```
TheLegend27 as Nyanzo: 12300
```

Sample Input 3

```
2 14000
TakesALicking Phurah 3
102300 120
113400 150
124100 150
BasketDude Clawmbra 5
1700000 230
1719200 -500
1721500 500
1723300 -450
1730000 250
```

Sample Output 3

```
BasketDude as Clawmbra: 1720999
```

Problem L: Hyperloop

The hyperloop is a new and fascinating mode of cat transportation. The cats sit in special capsules which "float" on a thin air layer in an almost vacuumed steel tube. The projected top speed is above $1\,000$ km/h. As you can imagine, the construction of the new hyperloop system is quite expensive and there is always a shortage of public money. Thus, we have to carefully decide where to build the first hyperloop route.

The current plan is to build the hyperloop next to the existing railway tracks and to upgrade the ordinary railway stations to hyperloop stations. The construction site is first erected at one of the stations and then moves through the rail network as the hyperloop gets constructed. Moving the construction equipment is quite expensive, so the building contract mandates that the construction site may never be moved without the machines doing any work. This means that it can not be moved along any railway track that was already upgraded and it is also not allowed to disassemble the construction site and resume construction from a different station.

You are given the two-dimensional coordinates of the railway stations as well as the directly connected stations. For simplicity reasons, we assume that the distance (in km) between two railway stations is the euclidean distance.

You have to select one route for the hyperloop so that the average travel time between any pair of stations in the upgraded network is minimized. When computing travel times you should assume that it is only possible to change trains (hyperloop or railway) at a station and that changing trains costs no time.

Input

The input consists of:

- one line with three integers n, r, and h ($2 \leq n \leq 15$; $100 \leq r \leq 200$; $800 \leq h \leq 1\,200$), where n is the number of railway stations, r is the average train speed in km/h and h is the average hyperloop speed in km/h;
- one line with with n coordinate pairs describing the positions of the railway stations; all coordinates are integers with absolute value at most $10\,000$;
- n lines; the i^{th} line specifying the direct connections of the i^{th} railway station:
  - the line starts with an integer c_i, the number of connections ($1 \leq c_i \leq 3$);
  - followed by c_i distinct integers listing the connected stations (0-indexed). Connections are always bi-directional and no station is directly connected to itself.

You may assume that there is at least one path to reach any station from any other station.

Output

On the first line of output, print the optimal average travel time (in hours) between any pair of different stations. The travel time should be accurate to an absolute or relative error of at most 10^{-6}.

On the second line, print the length k of the hyperloop track, followed by k numbers specifying the route of the hyperloop. The best answer may include the same station more than once. If there is more than one optimal answer, any will be accepted.

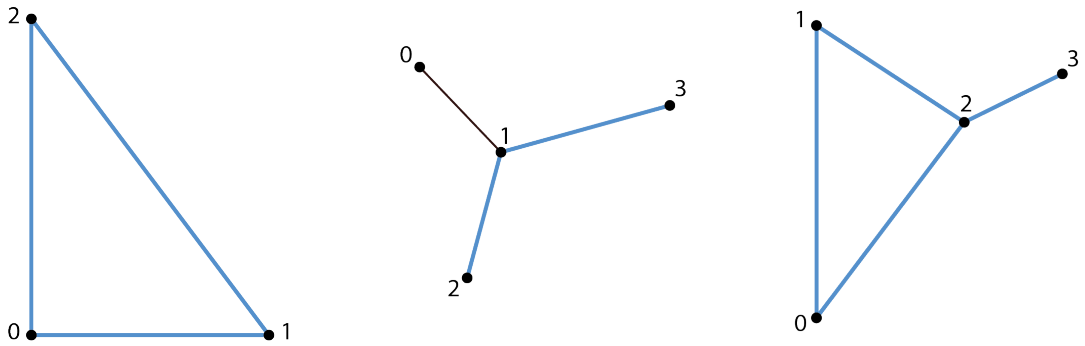

Figure L.1: Illustrations for the sample cases. The hyperloop is marked in blue. In the first sample, the optimal average time is calculated as $\frac{300+400+500}{3 \cdot 1000}$.

Sample Input 1

```
3 180 1000
100 100 400 100 100 500
2 1 2
2 0 2
2 0 1
```

Sample Output 1

```
0.4
4 1 0 2 1
```

Sample Input 2

```
4 150 800
-10 42 9 22 1 -8 49 33
1 1
3 2 0 3
1 1
1 1
```

Sample Output 2

```
0.137287398839
3 2 1 3
```

Sample Input 3

```
4 100 1000
0 0 0 600 300 400 500 500
2 1 2
2 2 0
3 1 3 0
1 2
```

Sample Output 3

```
0.498655108057
5 2 1 0 2 3
```

Problem M: Consequences

When he was young, Charles' biggest wish was to have a pet — a cute cat of course. Sadly, his parents declined his request and wisely advised him to think of the consequences. So Charles focused solely on consequences for years.

He spent a lot of time coming up with logical consequences and thinking about them. Recently, he came up with

$$\big((a \to b) \to a\big) \to a$$

but he still is unsure in which situations the formula holds. That is, for which truth values of the variables a and b does the formula evaluate to true?

He knows that an implication $\alpha \to \beta$ of two formulas α and β is true if the formula α evaluates to false or β evaluates to true. Given a formula of implications Charles comes up with, can you help him finding an assignment of the variables that makes the formula evaluate to true?

Input

The input lists a consequence formula which is satisfiable and at most $100\,000$ characters long. Charles builds consequence formulae by the following rules:

1. Any letter $L \in \{\texttt{a}, \ldots, \texttt{z}, \texttt{A}, \ldots, \texttt{Z}\}$ is a consequence formula. The letters are read case sensitively.
2. If X and Y are consequence formulae, then `(`X `->` Y`)` is a consequence formula as well.

For readability, the outermost parentheses are omitted, e.g. Charles writes `(a -> b) -> a` instead of `((a -> b) -> a)`.

Output

For each letter in the input print precisely one assignment, such that the input formula evaluates to true. Each assignment is a single line containing either L`=0` or L`=1`, where L is as above. The assignments may appear in any order.

Sample Input 1

```
((a -> b) -> a) -> a
```

Sample Output 1

```
a=0
b=1
```

Sample Input 2

```
x -> (y -> z)
```

Sample Output 2

```
x=0
y=0
z=0
```

Sample Input 3

```
(P -> Q) -> P
```

Sample Output 3

```
P=1
Q=0
```

This page is intentionally left (almost) blank.