Problem A. Alternative Scale of Notation

Input file:	`alter.in`
Output file:	`alter.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

One may define a map of strings over an alphabet $\Sigma_B = \{C_1, C_2, \dots C_B\}$ of size B to non-negative integer numbers, using characters as digits $C_1 = 0$, $C_2 = 1$, ..., $C_B = B - 1$ and interpreting the string as the representation of some number in a scale of notation with base B. Let us denote this map by U_B, for a string $\alpha[1..n]$ of length n we put

$$U_B(\alpha) = \sum_{i=0}^{n-1} \alpha[n-i] \cdot B^i.$$

For example, $U_3(1001) = 1 \cdot 27 + 0 \cdot 9 + 0 \cdot 3 + 1 \cdot 1 = 28$.

However, this correspondence has one major drawback: it is not one-to-one. For example,

$$28 = U_3(1001) = U_3(01001) = U_3(001001) = \dots,$$

infinitely many strings map to the number 28.

In mathematical logic and computer science this may be unacceptable. To overcome this problem, the alternative interpretation is used. Let us interpret characters as digits, but in a slightly different way: $C_1 = 1$, $C_2 = 2$, ..., $C_B = B$. Note that now we do not have 0 digit, but rather we have a rudiment B digit. Now we define the map V_B in a similar way, for each string $\alpha[1..n]$ of length n we put

$$V_B(\alpha) = \sum_{i=0}^{n-1} \alpha[n-i] \cdot B^i.$$

For an empty string ε we put $V_B(\varepsilon) = 0$.

This map looks very much like U_B, however, the set of digits is now different. So, for example, we have $V_3(1313) = 1 \cdot 27 + 3 \cdot 9 + 1 \cdot 3 + 3 \cdot 1 = 60$.

It can be easily proved that the correspondence defined by this map is one-to-one and onto. Such a map is called *bijective*, and it is well known that every bijective map has an inverse. Your task in this problem is to compute the inverse for the map V_B. That is, for a given integer number x you have to find the string α, such that $V_B(\alpha) = x$.

Input

The first line of the input file contains B ($2 \le B \le 9$). The second line contains an integer number x given in a usual decimal scale of notation, $0 \le x \le 10^{100}$.

Output

Output such string α, consisting only of digits from the set $\{1, 2, \dots, B\}$, that $V_B(\alpha) = x$.

Sample input and output

alter.in	alter.out
3 60	1313
3 0	

Problem B. Bring Them There

Input file:	`bring.in`
Output file:	`bring.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

By the year 3141, the human civilization has spread all over the galaxy. The special hypertunnels are used to travel from one star system to another. To use the hypertunnel, you fly to a special location near the source star using your spaceship, activate the hyperjumper, fly through the hypertunnel, get out near your destination star and fly to the planet you need. The whole process takes exactly one day. A small drawback of the system is that for each tunnel every day only one spaceship can travel using this tunnel.

You are working in the transportation department of the "Intergalaxy Business Machines" company. This morning your boss has assigned a new task to you. To run the programming contest IBM needs to deliver K supercomputers from Earth where the company headquarters are located to the planet Eisiem. Since supercomputers are very large, one needs the whole spaceship to carry each supercomputer. You are asked to find a plan to deliver the supercomputers that takes as few days as possible. Since IBM is a very powerful corporation, you may assume that any time you need some tunnel for hyperjump, it is at your service. However, you still can use each tunnel only once a day.

Input

The first line of the input file contains N — the number of star systems in the galaxy, M — the number of tunnels, K — the number of supercomputers to be delivered, S — the number of the solar system (the system where planet Earth is) and T — the number of the star system where planet Eisiem is ($2 \le N \le 50$, $1 \le M \le 200$, $1 \le K \le 50$, $1 \le S, T \le N$, $S \ne T$).

Next M lines contain two different integer numbers each and describe tunnels. For each tunnel the numbers of star systems that it connects are given. The tunnel can be traveled in both directions, but remember that each day only one ship can travel through it, in particular, two ships cannot simultaneously travel through the same tunnel in opposite directions. No tunnel connects a star to itself and any two stars are connected by at most one tunnel.

Output

On the first line of the output file print L — the fewest number of days needed to deliver K supercomputers from star system S to star system T using hypertunnels. Next L lines must describe the process. Each line must start with C_i — the number of ships that travel from one system to another this day. C_i pairs of integer numbers must follow, pair A, B means that the ship number A travels from its current star system to star system B.

It is guaranteed that there is a way to travel from star system S to star system T.

Sample input and output

bring.in	bring.out
6 7 4 1 6 1 2 2 3 3 5 5 6 1 4 4 6 4 3	4 2 1 2 2 4 3 1 3 2 6 3 4 3 1 5 3 6 4 4 2 1 6 4 6

Problem C. Code Formatting

Input file:	`code.in`
Output file:	`code.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Programmers are known to wage religious wars when issues of proper code formatting are discussed. When new team of programmers starts working on a project, it often brings slightly different code formatting style and wants to reformat old source code according to their own style. Moreover, inexperienced programmers often neglect the importance of good and consistent code style, thus complicating the work of their teammates and themselves. This situation creates thriving market for code formatting tools.

You are taking part in a proof-of-concept project for a new code formatting tool code named Salvation. This is only a pilot project aimed not for a practical usefulness, but to demonstrate your ability to parse and format code of a high-level language. Your task is to write code formatter for a language called TRIVIAL (The Rival Implementation-Agnostic Language). This language has trivial lexical and grammatical structures. It does not have any keywords and control structures, because all constructs of the language are represented as function calls and closures.

The lexical structure consists of identifiers, opening and closing parenthesis and curly braces, commas, and semi-colons. Identifiers consist only of digits '0'–'9' and Latin letters 'a'–'z', 'A'–'Z'. Lexical terms may be separated by whitespaces, leading and trailing whitespaces in the file are also allowed. Whitespace may consist of spaces, tab characters (ASCII code 9), and line separators (a pair of ASCII 13, 10).

The structure of the valid trivial program is derived from the following productions:

- Program ::= Block
- Block ::= '{' Statements '}'
- Statements ::= Statement | Statement Statements
- Statement ::= Expression ';'
- Expression ::= *identifier* ['(' Arguments ')'] [Block]
- Arguments ::= Expression | Expression ',' Arguments

Properly formatted trivial program additionally conforms to the following rules:

- There are no empty lines.
- Tab characters are not used.
- The first character of the file is opening curly brace '{' (no preceding whitespaces), and the last character of the file is closing curly brace '}' (no trailing whitespaces).
- Each line is preceded by $4N$ space characters, where N is called *indentation level*.
- The first and the last lines of the program have zero indentation level.
- Lines that constitute block body and are enclosed in curly braces '{'...'}' have one more indentation level.
- No whitespace is allowed inside the line with the exception of the following two cases where a single space character is mandatory: before opening curly brace character '{' and after comma ','.
- Lines (with the only exception of the last line) end with semicolon ';' or opening curly brace '{' characters. These characters cannot appear in the middle or at the beginning of any line (including the last one).
- Closing curly brace '}' characters appear only at the beginning of lines after indentation spaces.

See sample output section for an example of properly formatted trivial program.

Input

The input file contains valid trivial program. Size of the input file does not exceed 2000 bytes.

Output

Write to the output file properly formatted trivial code for the program given in the input file.

Sample input and output

code.in

```
{class(Point)
{
  member ( int ( x ) ) ; member ( int ( y ) ) ;
  member ( fun ( Length )
  {
    return ( sqrt ( sum ( sqr ( x ),sqr ( y ) ) ) );
  } ) ;
};
Main
{
  repeat
  {
    set ( n,input ( int ) ) ;
    for ( int ( i,0 ) , lt ( i,n ) , inc ( i ) )
    {
      print ( mult ( n,n ) ) ;
    };
  };
}; }
```

code.out

```
{
    class(Point) {
        member(int(x));
        member(int(y));
        member(fun(Length) {
            return(sqrt(sum(sqr(x), sqr(y))));
        });
    };
    Main {
        repeat {
            set(n, input(int));
            for(int(i, 0), lt(i, n), inc(i)) {
                print(mult(n, n));
            };
        };
    };
}
```

Problem D. Data Mining

Input file:	`data.in`
Output file:	`data.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Dr. Tuple is working on the new data-mining application for Advanced Commercial Merchandise Inc. One of the subroutines for this application works with two arrays P and Q containing N records of data each (records are numbered from 0 to $N-1$). Array P contains hash-like structure with keys. Array P is used to locate record for processing and the data for the corresponding record is later retrieved from the array Q.

All records in array P have a size of S_P bytes and records in array Q have size of S_Q bytes. Dr. Tuple needs to implement this subroutine with the highest possible performance because it is a hot-spot of the whole data-mining application. However, S_P and S_Q are only known at run-time of application which complicates or makes impossible to make certain well-known compile-time optimizations.

The straightforward way to find byte-offset of i-th record in array P is to use the following formula:

$$\mathrm{Pofs}(i) = S_P \cdot i, \tag{1}$$

and the following formula for array Q:

$$\mathrm{Qofs}(i) = S_Q \cdot i. \tag{2}$$

However, multiplication computes much slower than addition or subtraction in modern processors. Dr. Tuple avoids usage of multiplication while scanning array P by keeping computed byte-offset $\mathrm{Pofs}(i)$ of i-th record instead of its index i in all other data-structures of data-mining application. He uses the following simple formulae when he needs to compute byte-offset of the record that precedes or follows i-th record in array P:

$$\mathrm{Pofs}(i+1) = \mathrm{Pofs}(i) + S_P$$
$$\mathrm{Pofs}(i-1) = \mathrm{Pofs}(i) - S_P$$

Whenever a record from array P is located by either scanning of the array or by taking $\mathrm{Pofs}(i)$ from other data structures, Dr. Tuple needs to retrieve information from the corresponding record in array Q. To access record in array Q its byte-offset $\mathrm{Qofs}(i)$ needs to be computed. One can immediately derive formula to compute $\mathrm{Qofs}(i)$ with known $\mathrm{Pofs}(i)$ from formulae (1) and (2):

$$\mathrm{Qofs}(i) = \mathrm{Pofs}(i)/S_P \cdot S_Q \tag{3}$$

Unfortunately, this formula not only contains multiplication, but also contains division. Even though only integer division is required here, it is still an order of magnitude slower than multiplication on modern processors. If coded this way, its computation is going to consume the most of CPU time in data-mining application for ACM Inc.

After some research Dr. Tuple has discovered that he can replace formula (3) with the following fast formula:

$$\mathrm{Qofs}'(i) = (\mathrm{Pofs}(i) + \mathrm{Pofs}(i) \mathbin{<<} A) \mathbin{>>} B \tag{4}$$

where A and B are non-negative integer numbers, "$<< A$" is left shift by A bits (equivalent to integer multiplication by 2^A), "$>> B$" is right shift by B bits (equivalent to integer division by 2^B).

This formula is an order of magnitude faster than (3) to compute, but it generally cannot always produce the same result as (3) regardless of the choice for values of A and B. It still can be used if one is willing to sacrifice some extra memory.

Conventional layout of array Q in memory (using formula (2)) requires $N \cdot S_Q$ bytes to store the entire array. Dr. Tuple has found that one can always choose such K that if he allocates K bytes of memory for the array Q (where $K \geq N \cdot S_Q$) and carefully selects values for A and B, the fast formula (4) will give non-overlapping storage locations for each of the N records of array Q.

Your task is to write a program that finds minimal possible amount of memory K that needs to be allocated for array Q when formula (4) is used. Corresponding values for A and B are also to be found. If multiple pairs of values for A and B give the same minimal amount of memory K, then the pair where A is minimal have to be found, and if there is still several possibilities, the one where B is minimal. You shall assume that integer registers that will be used to compute formula (4) are wide enough so that overflow will never occur.

Input

The input file consists of three integer numbers N, S_P, and S_Q separated by spaces ($1 \leq N \leq 2^{20}$, $1 \leq S_P \leq 2^{10}$, $1 \leq S_Q \leq 2^{10}$).

Output

Write to the output file a single line with three integer numbers K, A, and B separated by spaces.

Sample input and output

data.in	data.out
20 3 5	119 0 0
1024 7 1	1119 2 5

Problem E. Entropy

Input file:	`entropy.in`
Output file:	`entropy.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

In 1948 Claude E. Shannon in "The Mathematical Theory of Communication" has introduced his famous formula for the entropy of a discrete set of probabilities $p_1, \ldots, p_n$:

$$H = -\sum p_i \log_2 p_i \ .$$

We will apply this formula to an arbitrary text string by letting p_i be the relative frequencies of occurrence of characters in the string. For example, the entropy of the string "Northeastern European Regional Contest" with the length of 38 characters (including 3 spaces) is 3.883 with 3 digits after decimal point. The following table shows relative frequencies and the corresponding summands for the entropy of this string.

char	occurs	p_i	$-p_i \log_2 p_i$	char	occurs	p_i	$-p_i \log_2 p_i$
space	3	0.079	0.289	i	1	0.026	0.138
C	1	0.026	0.138	l	1	0.026	0.138
E	1	0.026	0.138	n	4	0.105	0.342
N	1	0.026	0.138	o	4	0.105	0.342
R	1	0.026	0.138	p	1	0.026	0.138
a	3	0.079	0.289	r	3	0.079	0.289
e	5	0.132	0.385	s	2	0.053	0.224
g	1	0.026	0.138	t	4	0.105	0.342
h	1	0.026	0.138	u	1	0.026	0.138

Your task is to find a string with the given entropy.

Input

Input file consists of a single real number H ($0.00 \le H \le 6.00$) with 2 digits after decimal point.

Output

Write to the output file a line with a single string of at least one and up to 1000 characters '0'–'9', 'a'–'z', 'A'–'Z', '.' (dot), and spaces. This string must have the entropy within 0.005 of H.

Sample input and output

entropy.in	entropy.out
3.88	Northeastern European Regional Contest

Problem F. Farmer Bill's Problem

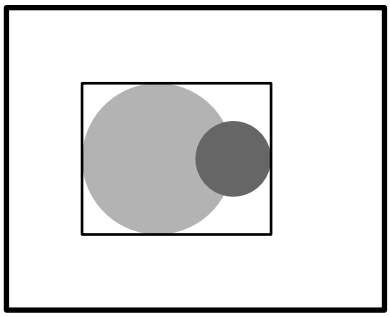
Input file:	`farmer.in`
Output file:	`farmer.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

It is rumored that the planet Earth is often visited by Unidentified Flying Objects (UFOs). Sometimes UFOs land and leave burned out regions. Observations show that these regions have the form of circles.

Recently farmer Bill has found such circles on his nice rectangular wheat field. Bill likes all mysterious things very much, so he has decided to keep these circles on the field. However, although being an ufolog, first of all Bill is the farmer, so he needs to harvest his wheat. Therefore he has decided to keep some regions containing circles intact, and harvest the rest of the field.

All regions that Bill keeps unharvested must be rectangles that neither touch nor overlap each other. The sides of the rectangles must be parallel to the sides of the field. All circles left by UFOs must be inside these regions. The total area of the regions must be minimal possible, so that Bill could harvest the maximal possible part of his field.

Now Bill wants to know the total area of the field that he will be able to harvest. Help him!



Input

The first line of the input file contains two integer numbers x and y — the dimensions of Bill's field ($1 \le x, y \le 1000$). Let Bill's field be positioned on the plane in such a way that its corners are located in points with coordinates $(0, 0)$, $(x, 0)$, (x, y) and $(0, y)$.

The second line of the input file contains N — the number of circles left by UFOs on Bill's field ($0 \le N \le 100$). Next N lines describe circles: each line contains three positive integer numbers x_i, y_i and r_i — coordinates of the center and radius of the circle. Circles may touch, overlap or contain each other. All circles are completely located within the field bounds.

Output

Output a single integer number — the area of the part of the field that Bill will be able to harvest.

Sample input and output

farmer.in	farmer.out
10 8 2 4 4 2 6 4 1	60
10 8 2 3 3 1 1 1 1	64

Problem G. Game

Input file:	`game.in`
Output file:	`game.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

There is a legend that mathematicians in the XVIII century enjoyed playing the following game.

The game was played by three mathematicians. One of them was the game master. First of all, the game master declared some positive integer number N. After that he chose two different integer numbers X and Y ranging from 1 to N and told their sum to one player and their product to the other player. Each player knew whether he was told the sum or the product of the chosen numbers.

After that the players in turn informed the game master whether they knew the numbers he had chosen. First the player who was told the sum said whether he knew the numbers, after that the player who was told the product did, and so on.

For example the dialog could look like this.

Game master: "Let N be 10".

After that he chooses two numbers ranging from 1 to 10 and tells their sum to player S and their product to player P.

Player S: "I don't know these numbers."

Player P: "I don't know these numbers."

Player S: "I don't know these numbers."

Player P: "I don't know these numbers."

Player S: "Oh, now I know these numbers. You have chosen 3 and 6."

Given N and M — the number of times the players have said "I don't know these numbers", you have to find all possible pairs of numbers that could have been chosen by the game master.

Input

The first line of the input file contains N and M ($2 \le N \le 200$, $0 \le M \le 100$).

Output

First output the number of possible pairs of numbers that could have been chosen by the game master from the range 1 to N if both players altogether had said "I don't know these numbers" M times. After that output these pairs in arbitrary order, one on a line.

Sample input and output

game.in	game.out
10 4	3 2 5 3 6 3 10

Problem H. Hypertransmission

Input file:	`hyper.in`
Output file:	`hyper.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

The president of the Galactic Federation has recently decided that all planets of the galaxy must establish hyper-radio centers to broadcast their programs. To ensure the process, the government has signed the contract with well known hyper-radio equipment manufacturer Trojan Horse Ltd. By the terms of this contract the company has to provide N hypertransmitters, one for each planet of the Federation.

It is known that there are two main political movements in the galaxy: industrialism and ecologism. On each planet of the galaxy one of these movements has the majority. It is clear that after establishing the hyper-radio station on the planet, the political programs of the station will support the movement that has the majority on this planet.

All transmitters supplied by Trojan Horse Ltd will have the same range, so hyper-radio programs from each planet will be heard at the distance not exceeding R parsecs from it. Since the company director is actually the agent of the Dark Empire, he wants to choose R in such a way, that it would destabilize the political situation in the Galactic Federation.

More precisely, for each planet A let $N^+(A)$ be the number of planets where the same political movement as in A has the majority and hyper-radio programs from A are received, including A itself. Similarly, let $N^-(A)$ be the number of planets where the other political movement has the majority and hyper-radio programs from A are received. The planet A is called *destabilizing* if $N^+(A) < N^-(A)$.

Your task is to choose such R that the number D of destabilizing planets is maximal possible. Since increasing transmitter's range requires more resources for its manufacturing, you must find the smallest possible R maximizing D.

Input

The first line of the input file contains N — the number of planets in the Galactic Federation ($1 \le N \le 1000$). Next N lines contain four integer numbers x_i, y_i, z_i, and p_i each and describe the planets: x_i, y_i, and z_i specify the coordinates of the planet in space, $p_i = 0$ if the industrialists have the majority on the planet and $p_i = 1$ if the ecologists have the majority. All coordinates do not exceed 10 000 by their absolute value. No two planets occupy the same point.

Output

First output D — the maximal possible number of destabilizing planets. After that output non-negative real number R — the minimal range that hyper-radio transmitters must have so that the number of destabilizing planets is D. R must be accurate within 10^{-4} of the correct answer.

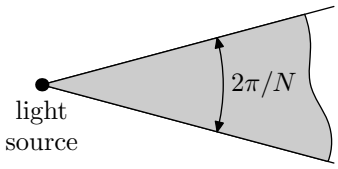
Sample input and output

hyper.in	hyper.out
4 0 0 0 1 0 1 0 0 1 0 0 0 1 1 0 1	4 1.0000
4 0 0 0 1 1 0 0 0 0 1 0 0 0 0 1 1	0 0.0000

Problem I. Illumination

Input file:	`ill.in`
Output file:	`ill.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given N light sources on the plane, each of which illuminates the angle of $2\pi/N$ with the vertex in the source point (including its sides).



You must choose the direction of the illuminated angle for each of these sources, so that the whole plane is illuminated. It can be proved that this is always possible.

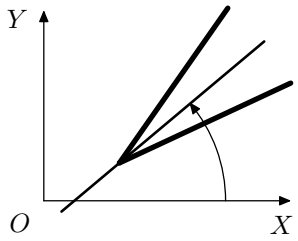
A light source itself casts no shadow and does not interfere with light beams from the other sources.

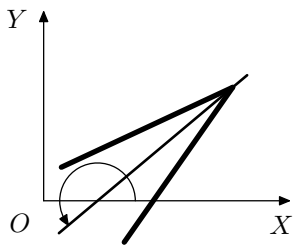
Input

The first line of the input file contains N — the number of light sources ($3 \le N \le 30$). Next N lines contain two integer numbers each — the coordinates of the light sources. All coordinates do not exceed 100 by their absolute value. No two sources coincide.

Output

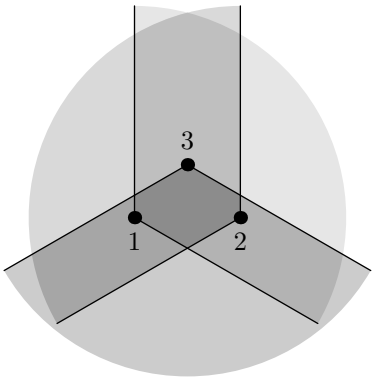
Print N real numbers — for each light source specify an angle that the bisector of the illuminated angle makes with OX axis, counterclockwise. Print at least six digits after the decimal point. No angle must exceed 100π by its absolute value.





Sample input and output

ill.in	ill.out
3 0 0 2 0 1 1	0.52359877559829887 2.61799387799149437 4.71238898038468986



Problem J. Jurassic Remains

Input file:	`jurassic.in`
Output file:	`jurassic.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Paleontologists in Siberia have recently found a number of fragments of Jurassic period dinosaur skeleton. The paleontologists have decided to forward them to the paleontology museum. Unfortunately, the dinosaur was so huge, that there was no box that the fragments would fit into. Therefore it was decided to split the skeleton fragments into separate bones and forward them to the museum where they would be reassembled. To make reassembling easier, the joints where the bones were detached from each other were marked with special labels. Meanwhile, after packing the fragments, the new bones were found and it was decided to send them together with the main fragments. So the new bones were added to the package and it was sent to the museum.

However, when the package arrived to the museum some problems have shown up. First of all, not all labels marking the joints were distinct. That is, labels with letters 'A' to 'Z' were used, and each two joints that had to be connected were marked with the same letter, but there could be several pairs of joints marked with the same letter.

Moreover, the same type of labels was used to make some marks on the new bones added to the box. Therefore, there could be bones with marked joints that need not be connected to the other bones. The problem is slightly alleviated by the fact that each bone has at most one joint marked with some particular letter.

Your task is to help the museum workers to restore some possible dinosaur skeleton fragments. That is, you have to find such set of bones, that they can be connected to each other, so that the following conditions are true:

- If some joint is connected to the other joint, they are marked with the same label.
- For each bone from the set each joint marked with some label is connected to some other joint.
- The number of bones used is maximal possible.

Note that two bones may be connected using several joints.

Input

The first line of the input file contains N — the number of bones ($1 \le N \le 24$). Next N lines contain bones descriptions: each line contains a non-empty sequence of different capital letters, representing labels marking the joints of the corresponding bone.

Output

On the first line of the output file print L — the maximal possible number of bones that could be used to reassemble skeleton fragments. After that output L integer numbers in ascending order — the bones to be used. Bones are numbered starting from one, as they are given in the input file.

Sample input and output

jurassic.in	jurassic.out
6 ABD EG GE ABE AC BCD	5 1 2 3 5 6
1 ABC	0

Problem K. King's Quest

Input file:	`king.in`
Output file:	`king.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Once upon a time there lived a king and he had N sons. And there were N beautiful girls in the kingdom and the king knew about each of his sons which of those girls he did like. The sons of the king were young and light-headed, so it was possible for one son to like several girls.

So the king asked his wizard to find for each of his sons the girl he liked, so that he could marry her. And the king's wizard did it — for each son the girl that he could marry was chosen, so that he liked this girl and, of course, each beautiful girl had to marry only one of the king's sons.

However, the king looked at the list and said: "I like the list you have made, but I am not completely satisfied. For each son I would like to know all the girls that he can marry. Of course, after he marries any of those girls, for each other son you must still be able to choose the girl he likes to marry."

The problem the king wanted the wizard to solve had become too hard for him. You must save wizard's head by solving this problem.

Input

The first line of the input file contains N — the number of king's sons ($1 \le N \le 2000$). Next N lines for each of king's sons contain the list of the girls he likes: first K_i — the number of those girls, and then K_i different integer numbers, ranging from 1 to N denoting the girls. The sum of all K_i does not exceed 200 000.

The last line of the input file contains the original list the wizard had made — N different integer numbers: for each son the number of the girl he would marry in compliance with this list. It is guaranteed that the list is correct, that is, each son likes the girl he must marry according to this list.

Output

Output N lines. For each king's son first print L_i — the number of different girls he likes and can marry so that after his marriage it is possible to marry each of the other king's sons. After that print L_i different integer numbers denoting those girls, in arbitrary order.

Sample input and output

king.in	king.out
4 2 1 2 2 1 2 2 2 3 2 3 4 1 2 3 4	2 1 2 2 1 2 1 3 1 4