ACM International Collegiate Programming Contest 2000/2001
Mid-Central European Regional Contest

Problem A

Atlantis

Source: `atlantis.(c|cc|pas|java)`
Input: `atlantis.in`

There are several ancient Greek texts that contain descriptions of the fabled island Atlantis. Some of these texts even include maps of parts of the island. But unfortunately, these maps describe different regions of Atlantis. Your friend Bill has to know the total area for which maps exist. You (unwisely) volunteered to write a program that calculates this quantity.

Input

The input file consists of several test cases. Each test case starts with a line containing a single integer n ($1 \leq n \leq 100$) of available maps. The n following lines describe one map each. Each of these lines contains four numbers x_1, y_1, x_2, y_2 ($0 \leq x_1 < x_2 \leq 100\,000, 0 \leq y_1 < y_2 \leq 100\,000$), not necessarily integers. The values (x_1, y_1) and (x_2, y_2) are the coordinates of the top-left resp. bottom-right corner of the mapped area.

The input file is terminated by a line containing a single 0. Don't process it[1].

Output

For each test case, your program should output one section. The first line of each section must be "`Test case #`k", where k is the number of the test case (starting with 1). The second one must be "`Total explored area: `a", where a is the total explored area (i.e. the area of the union of all rectangles in this test case), printed exact to two digits to the right of the decimal point.

Output a blank line after each test case.

Sample Input

```
2
10 10 20 20
15 15 25 25.5
0
```

Sample Output

```
Test case #1
Total explored area: 180.00
```

[1] We warned you!

Problem C

Erdös Numbers

Source: `erdos.(c|cc|pas|java)`
Input: `erdos.in`

The Hungarian Paul Erdös (1913–1996, pronounced as "Ar-dish") was not only one of the strangest mathematicians of the 20th century, he was also among the most famous ones. He kept on publishing widely circulated papers up to a very high age, and every mathematician having the honor of being a co-author to Erdös is well respected.

Not everybody got a chance to co-author a paper with Erdös, so many people were content if they managed to publish a paper with somebody who had published a paper with Erdös. This gave rise to the so-called *Erdös numbers*. An author who has jointly published with Erdös had Erdös number 1. An author who had not published with Erdös but with somebody with Erdös number 1 obtained Erdös number 2, and so on. Today, nearly everybody wants to know what Erdös number he or she has. Your task is to write a program that computes Erdös numbers for a given set of scientists.

Input

The input file contains a sequence of scenarios, each scenario consisting of a paper database and a list of names. A scenario begins with the line "p n", where p and n are natural numbers with $1 \leq p \leq 32000, 1 \leq n \leq 3000$. Following this line are p lines containing descriptions of papers (this is the paper database). A paper is described by a line of the following form:

LastName1, *FirstName1*, *LastName2*, *Firstname2*, ...: *TitleOfThePaper*

The names and the title may contain any ASCII characters between 32 and 126 except commas and colons. There will always be exactly one space character following each comma. The first name may be abbreviated, but the same name will always be written in the same way. In particular, Erdös' name is always written as "`Erdos, P.`". (Umlauts like 'ö','ä',... are simply written as 'o','a', ...)

Example:

```
Smith, M.N., Martin, G., Erdos, P.: Newtonian forms of prime factors
matrices.
```

After the p papers follow n lines each containing exactly one name in the same format as in the paper database.

The line '`0 0`' terminates the input.

No name will consist of more than 40 characters. No line in the input file contains more than 250 characters. In each scenario there will be at most 10 000 different authors.

Output

For every scenario first print the number of the scenario in the format shown in the sample output. Then print for every author name in the list of names their Erdös number based on the papers in the paper database of the scenario. The authors should be output in the order given in the input file. Authors that do not have any relation to Erdös via the papers have Erdös number `infinity`. Adhere to the format shown in the sample output.

Sample Input

```
2 2
Smith, M.N., Martin, G., Erdos, P.: Newtonian forms of prime factors matrices.
Gardner, M., Martin, G.: Commuting Names
Smith, M.N.
Gardner, M.
0 0
```

Sample Output

```
Database #1
Smith, M.N.: 1
Gardner, M.: 2
```

ACM International Collegiate Programming Contest 2000/2001
Mid-Central European Regional Contest

Problem D

Number Game

Source: `numbergame.(c|cc|pas|java)`
Input: `numbergame.in`

Christine and Matt are playing an exciting game they just invented: the Number Game. The rules of this game are as follows.

The players take turns choosing integers greater than 1. First, Christine chooses a number, then Matt chooses a number, then Christine again, and so on. The following rules restrict how new numbers may be chosen by the two players:

- A number which has already been selected by Christine or Matt, or a multiple of such a number, cannot be chosen.
- A sum of such multiples cannot be chosen, either.

If a player cannot choose any new number according to these rules, then that player loses the game.

Here is an example: Christine starts by choosing 4. This prevents Matt from choosing 4, 8, 12, etc. Let's assume that his move is 3. Now the numbers 3, 6, 9, etc. are excluded, too; furthermore, numbers like: $7 = 3+4, 10 = 2\cdot 3+4, 11 = 3+2\cdot 4, 13 = 3\cdot 3+4, \ldots$ are also not available. So, in fact, the only numbers left are 2 and 5. Christine now selects 2. Since $5 = 2+3$ is now forbidden, she wins because there is no number left for Matt to choose.

Your task is to write a program which will help play (and win!) the Number Game. Of course, there might be an infinite number of choices for a player, so it may not be easy to find the best move among these possibilities. But after playing for some time, the number of remaining choices becomes finite, and that is the point where your program can help. Given a game position (a list of numbers which are not yet forbidden), your program should output all *winning moves*.

A winning move is a move by which the player who is about to move can force a win, no matter what the other player will do afterwards. More formally, a winning move can be defined as follows.

- A winning move is a move after which the game position is a losing position.
- A winning position is a position in which a winning move exists. A losing position is a position in which no winning move exists.
- In particular, the position in which all numbers are forbidden is a losing position. (This makes sense since the player who would have to move in that case loses the game.)

Input

The input file consists of several test cases. Each test case is given by exactly one line describing one position.

Each line will start with a number n ($1 \le n \le 20$), the number of integers which are still available. The remainder of this line contains the list of these numbers $a_1, \ldots, a_n$ ($2 \le a_i \le 20$).

The positions described in this way will always be positions which can really occur in the actual Number Game. For example, if 3 is not in the list of allowed numbers, 6 is not in the list, either.

At the end of the input file, there will be a line containing only a zero (instead of n); this line should not be processed.

Output

For each test case, your program should output "`Test case #`*m*", where *m* is the number of the test case (starting with 1). Follow this by either "`There's no winning move.`" if this is true for the position described in the input file, or "`The winning moves are:` w_1 w_2 ... w_k" where the w_i are all winning moves in this position, satisfying $w_i < w_{i+1}$ for $1 \leq i < k$. After this line, output a blank line.

Sample Input

```
2 2 5
2 2 3
5 2 3 4 5 6
0
```

Sample Output

```
Test Case #1
The winning moves are: 2

Test Case #2
There's no winning move.

Test Case #3
The winning moves are: 4 5 6
```

ACM International Collegiate Programming Contest 2000/2001
Mid-Central European Regional Contest

Problem H

Smith Numbers

Source: `smith.(c|cc|pas|java)`
Input: `smith.in`

While skimming his phone directory in 1982, Albert Wilansky, a mathematician of Lehigh University, noticed that the telephone number of his brother-in-law H. Smith had the following peculiar property: The sum of the digits of that number was equal to the sum of the digits of the prime factors of that number. Got it? Smith's telephone number was 493-7775. This number can be written as the product of its prime factors in the following way:

$$4937775 = 3 \cdot 5 \cdot 5 \cdot 65837$$

The sum of all digits of the telephone number is $4+9+3+7+7+7+5 = 42$[†] , and the sum of the digits of its prime factors is equally $3+5+5+6+5+8+3+7 = 42$. Wilansky was so amazed by his discovery that he named this kind of numbers after his brother-in-law: Smith numbers.

As this observation is also true for every prime number, Wilansky decided later that a (simple and unsophisticated) prime number is not worth being a Smith number, so he excluded them from the definition.

Wilansky published an article about Smith numbers in the *Two Year College Mathematics Journal* and was able to present a whole collection of different Smith numbers: For example, 9985 is a Smith number and so is 6036. However, Wilansky was not able to find a Smith number that was larger than the telephone number of his brother-in-law. It is your task to find Smith numbers that are larger than 4937775!

Input

The input file consists of a sequence of positive integers, one integer per line. Each integer will have at most 8 digits. The input is terminated by a line containing the number 0.

Output

For every number $n > 0$ in the input, you are to compute the smallest Smith number which is larger than n, and print it on a line by itself. You can assume that such a number exists.

Sample Input

```
4937774
0
```

Sample Output

```
4937775
```

[†] What else did you expect???