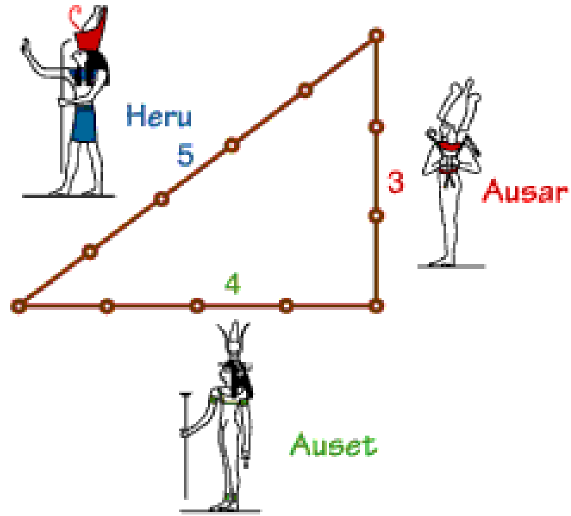
Problem A: Egypt



A long time ago, the Egyptians figured out that a triangle with sides of length 3, 4, and 5 had a right angle as its largest angle. You must determine if other triangles have a similar property.

The Input

Input represents several test cases, followed by a line containing 0 0 0. Each test case has three positive integers, less than 30,000, denoting the lengths of the sides of a triangle.

The Output

For each test case, a line containing "right" if the triangle is a right triangle, and a line containing "wrong" if the triangle is not a right triangle.

Sample Input

```
6 8 10
25 52 60
5 12 13
0 0 0
```

Output for Sample Input

```
right
wrong
right
```

Problem B: Buzzwords

The word the is the most common three-letter word. It even shows up inside other words, such as "o**the**r" and "ma**the**matics". Sometimes it hides, split between two words, such as "no**t he**re". Have you ever wondered what the most common words of lengths other than three are?

Your task is the following. You will be given a text. In this text, find the most common word of length one. If there are multiple such words, any one will do. Then count how many times this most common word appears in the text. If it appears more than once, output how many times it appears. Then repeat the process with words of length 2, 3, and so on, until you reach such a length that there is no longer any repeated word of that length in the text.

Input Specification

The input consists of a sequence of lines. The last line of input is empty and should not be processed. Each line of input other than the last contains at least one but no more than one thousand uppercase letters and spaces. The spaces are irrelevant and should be ignored.

Sample Input

```
OTHER MATHEMATICS NOT HERE
AA

```

Note that the last line of the sample input is a blank line.

Output Specification

For each line of input, output a sequence of lines, giving the number of repetitions of words of length 1, 2, 3, and so on. When you reach a length such that there are no repeated words of that length, output one blank line, do not output anything further for that input line, and move on to the next line of input.

Output for Sample Input

```
5
4
4
2
2

2
```

C. Restaurant Ratings

A famous travel web site has designed a new restaurant rating system. Each restaurant is rated by n ($1 \leq n \leq 15$) critics, each giving the restaurant a nonnegative numeric rating (higher score means better). The restaurants in each city are ranked as follows. First, sum up the ratings given by all the critics for a restaurant. A restaurant with a higher total sum is always better than one with a lower total sum. For restaurants with the same total sum, we rank them based on the ratings given by just critic 1. If there is still a tie, it is broken by comparing the ratings by critic 2, etc.

A restaurant owner received the ratings for his restaurant, and is curious about how it ranks in the city. He can easily find the sum of his own ratings, but he does not know the ratings or sums for all the other restaurants in the city. He decides to estimate his ranking by comparing his ratings to the number of possible unique sets of ratings that is no better than his own. You are asked to write a program to calculate this estimate (whether or not you think this will be very accurate).

Input:

The input may consist of a number of cases. Each case is specified on one line. On each line, the first integer is n, followed by n integers containing the ratings given by the n critics (in order). You may assume that the total sum of ratings for each restaurant is at most 30. The input is terminated by a line containing a 0 (zero). All restaurants in one city are assumed to be rated by the same number of critics, n.

Output:

For each input, print the number of different sets of ratings that is no better than the given set of ratings. You may assume that the output fits in a 64-bit signed integer. Follow this format exactly: "`Case`", one space, the case number, a colon and one space, and the answer for that case with no trailing spaces.

Sample Input	Sample Output
1 3	Case 1: 4
2 4 3	Case 2: 33
5 4 3 2 1 4	Case 3: 10810
0	

Problem D: Driving Range

These days, many carmakers are developing cars that run on electricity instead of gasoline. The batteries used in these cars are generally very heavy and expensive, so designers must make an important tradeoffs when determining the battery capacity, and therefore the range, of these vehicles. Your task is to help determine the minimum range necessary so that it is possible for the car to travel between any two cities on the continent.

The road network on the continent consists of cities connected by bidirectional roads of different lengths. Each city contains a charging station. Along a route between two cities, the car may pass through any number of cities, but the distance between each pair of consecutive cities along the route must be no longer than the range of the car. What is the minimum range of the car so that there is a route satisfying this constraint between every pair of cities on the continent?

Input Specification

The input consists of a sequence of road networks. The first line of each road network contains two positive integers *n* and *m*, the number of cities and roads. Each of these integers is no larger than one million. The cities are numbered from 0 to *n*-1. The first line is followed by *m* more lines, each describing a road. Each such line contains three non-negative integers. The first two integers are the numbers of the two cities connected by the road. The third integer is the length of the road. The last road network is followed by a line containing two zeros, indicating the end of the input.

Sample Input

```
3 3
0 1 3
1 2 4
2 1 5
2 0
0 0
```

Output Specification

For each road network, output a line containing one integer, the minimum range of the car that enables it to drive from every city to every other city. If it is not possible to drive from some city to some other city regardless of the range of the car, instead output a line containing the word `IMPOSSIBLE`.

Output for Sample Input

```
4
IMPOSSIBLE
```

Problem E: Frosh Week

During Frosh Week, students play various fun games to get to know each other and compete against other teams. In one such game, all the frosh on a team stand in a line, and are then asked to arrange themselves according to some criterion, such as their height, their birth date, or their student number. This rearrangement of the line must be accomplished only by successively swapping pairs of consecutive students. The team that finishes fastest wins. Thus, in order to win, you would like to minimize the number of swaps required.

Input Specification

The first line of input contains one positive integer n, the number of students on the team, which will be no more than one million. The following n lines each contain one integer, the student number of each student on the team. No student number will appear more than once.

Sample Input

```
3
3
1
2
```

Output Specification

Output a line containing the minimum number of swaps required to arrange the students in increasing order by student number.

Output for Sample Input

```
2
```

```
Problem F: Find the Key
-----------------------

PROBLEM: Can you find the secret keys?

INTRODUCTION
In this problem, a TEXT is a nonempty string of lower case letters. We assume
the standard order of lower case letters and that the POSITION of 'a' is 0,
of 'b' is 1, etc. We write    pos('a')=0,  pos('b')=1,.....,  pos('z')=25.

We are dealing with encrypting a text T using a secret key K, where the key K
also is a text whose length is always at least that of the text T. The
encryption process is simple. The encrypted version of T is the string A that
has the same length as T and each letter of A is computed as follows:

    A[i] = the letter at position (pos(T[i])+pos(K[i])) mod 26

For example, if T = "phrase" and K = "aaadaw" then A = "phrdsa".

WHAT TO DO
Your program should read a positive integer N and then N text pairs. For each
text pair A, B you know that both A and B have been encrypted using the same
key. Your task is to find the key that was used for each pair, or simply say
that the key was not found---see sample input-output cases below. You have
information that every given text pair was obtained by choosing two
distinct strings from the following SET of 12 texts:
         "firstphrase", "secondphrase", "thirdphrase",
         "fourthphrase","fifthphrase", "sixthphrase",
         "wordone", "wordtwo", "wordthree",
         "wordfour", "wordfive", "wordsix"
The output "Key not found" is produced for a text pair A, B, if no two of the
above texts can be encrypted as A, B using the same key.

You may assume that N < 10.  On the judges' input, computation should
complete within 5 seconds.

#########################
----- Sample Input  -----
4
aaaaaaaaaaa aaaabbbbbbb
firsyphrdsa fiftmphrdsa
tloaclro tloaqeobb
atxkamc atxkblwkl
----- Sample Output -----
Key not found
Key = aaaafaaadaw
Key = xxxxxxxx
Key = efghiefgh
#########################
```

Problem G: IRIS CLASSIFICATION

Problem Description:

This problem is to predict the specie of Iris based on four features: the length and width of the sepals and petals, measured to the nearest tenth of a centimetre. The training set consists of samples from each of the three species of Iris, which are labeled as 1, 2 and 3.

The K-nearest neighbours algorithm can be used to predict the specie of Iris given the training set. The training examples are vectors in a 4-dimensional feature space, each with a class label. To make the prediction, an unlabeled vector (a query or test point) is classified by assigning the label which is most frequent among the K (K is a user-defined constant) training samples nearest to that query point.

The K in this problem must be 3.

The distance metric in this problem must be Euclidean distance. Euclidean distance is the вЂњordinaryвЂќ distance between two points that one would measure with a ruler. For example, the Euclidean distance between points (1, 0, 1, 0) and (0, 1, 0, 1) is the square root of ((1-0)*(1-0)+(0-1)*(0-1)+(1-0)*(1-0)+(0-1)*(0-1)), which is 2.

Input:

The input file is divided into two parts. The first part contains multiple training cases. The first line of the input will contain a single decimal integer, which is the number of training cases to follow. Each training case consists of a single line containing five values separated by commas. The first four values are the features of this sample, and the last value is the class label of the sample. Feature values have one digit after the decimal point and range between 0.1 and 9.9, whereas the class labels are 1, 2, and 3.

The second part of the input file contains multiple test cases. The first line of the second part will contain a single decimal integer, which is the number of testing cases that follow. Each test case consists of a single line containing four values separated by commas, which are the features of this sample.

There will be at least three training cases and at most 200.
There will be at least one test case and at most 30.
It must be possible to classify 30 test cases in at most 30 seconds.

Output:

For each input case, the output is the class label, which is 1, 2, 3 or ?. The label ? is used if there is no most-frequent label among the K nearest neighbours.

Note: you may assume that third nearest neighbour is always nearer than the fourth nearest neighbour (i.e., you need not worry that ties in distance make it ambiguous which 3 training cases are the closest).

Sample Input:

```
8
1.0,1.0,2.0,2.9,1
1.0,1.0,2.5,2.5,1
1.0,1.0,3.0,2.0,2
1.0,1.0,0.9,2.0,3
1.0,1.0,2.0,0.8,1
```

```
1.0,1.0,2.0,0.5,1
1.0,1.0,2.0,0.5,2
1.0,1.0,2.0,0.5,3
2
1.0,1.0,2.0,2.0
9.9,9.9,2.0,0.5
```

Sample Output:

```
1
?
```

Problem H: The Word Chain Game

A popular word game, played over long trips, goes as follows. Play begins by one player selecting a word from a dictionary. Play then continues, in round-robin fashion, by having each player selecting a new word from the dictionary (one that was not previously selected) and that starts with the same letter as the last letter of the previous word. Play continues until the dictionary is exhausted. If the last word selected ends with the same letter that the first word started with, we have a win. If it ends with a different letter or if none of the remaining words in the dictionary can be selected, we have a loss. A dictionary is winnable if and only if all the words it contains can be linked together into a word chain whose ending and starting letters are the same.

Write a program that reads in a series of dictionaries and determines if each of the dictionaries is winnable or not.

Input

Your program will read several lines of text from the keyboard (stdin). The first line of the input consists of an integer, N, denoting the number of dictionaries to be read. The first line of each dictionary consists of an integer, W, denoting the number of words in the dictionary. The next W lines of input contain the dictionary, one word per line. All the words only contain lower case letters, a ... z. There are no duplicate words in a dictionary. A dictionary may contain up to 100 words and each word is at least one character long and at most 80 characters long. There will be at most 30 dictionaries.

Semantics

A word chain consists of a sequence of words from a dictionary. Each word in the chain must begin with the same letter as the last letter of the preceding word. Any word from the dictionary may be used to start the word chain. A dictionary is winnable if and only if there exists a word chain containing all the words in the dictionary.

Output

Your program must output to the console (stdout). For each dictionary in the input your program should output a single line containing the phrase "winnable" if the dictionary is winnable, and the phrase "not winnable" if the dictionary is not winnable. On the judge input, computation must complete within 30s.

Example

```
Input:                   Output:

  3                      not winnable
  2                      not winnable
  good                   winnable
  day
  2
  hello
  world
  3
  high
  ho
  oath
```