# Problem 1: FizzBuzz

## Background

The game of FizzBuzz was originally intended to help British school children learn to quickly determine integer divisibility. Not content with demoralizing the youth of Great Britain, FizzBuzz has reinvented itself in recent years as an interviewing tool used to assess entry-level programming ability.

FizzBuzz is "played" by counting from 1 to 100 (or some other limit) and, for each number, saying

* "Fizz" if the number is evenly divisible by 3
* "Buzz" if the number is evenly divisible by 5
* "FizzBuzz" if the number is evenly divisible by both
* The number itself otherwise

## The Problem

Write a program that can "play" FizzBuzz by reading a number from the console and printing the appropriate response.

*Hint: most programming languages feature a "modulus" operator (in Python, Java, C, and many others it is %) that will return the remainder of a division operation. E.g., 5 % 2 would result in 1, the remainder of taking 5 / 2...*

## Input

Input will consist of integers, one per line. Input is terminated by the integer 0 (for which you should generate no output).

## Output

Emit one line of text per [non-zero] input line. Output must consist of either "Fizz", "Buzz", "FizzBuzz", or the input integer itself.

## Sample Input

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
0

## Sample Output

1  
2  
Fizz  
4  
Buzz  
Fizz  
7  
8  
Fizz  
Buzz  
11  
Fizz  
13  
14  
FizzBuzz  
16  
17  
Fizz  
19  
Buzz

# Problem 2: FizzBuzz 2.0

## Background

For years J. Random Hacker has been filtering job applicants to her company, Hacktastic Young Programming Exploits (HYPE), with the classic FizzBuzz test, in which they must write a program that will "count" from 1 to 100 and, for each number, print

* "Fizz" if it is divisible by 3
* "Buzz" if it is divisible by 5
* "FizzBuzz" if it is divisible by 15
* The number itself otherwise

But thanks to social media buzz, *everybody* and her sister is now using FizzBuzz to screen applicants, and J. Random is worried that HYPE will become flooded with run-of-the-mill hipster hacker-wannabes who read about FizzBuzz on Twitter but who otherwise can't program their way out of hello world.

Her solution? **FizzBuzz 2.0:** the *polymorphic* code interview test!

## Input

Input will begin with one or more *divisibility tests*, one per line, in the format

N W

N is the number to test divisibility by, W is the word to print on success. You may assume that 1 < N < 1000 and that W is no more than 30 characters long.

This section of input ends with a line in the form

0 -

The remaining lines of input contain integers (one per line) to test against the divisibility rules, in the classic FizzBuzz manner. This section of input ends with a line containing the integer 0 (for which there is no output).

## Output

Emit one line of text per [non-zero] input integer following the classic FizzBuzz rules, but with 3/Fizz and 5/Buzz replaced by the divisibility tests included in the input.

If an integer is divisible by more than one test number, the output words should be printed in order of their associated numbers. (Recall the original game, where "Fizz" represented divisibility by 3 and "Buzz" by 5, that "Fizz" came before "Buzz" on numbers divisible by both.)

## Sample Input

3 Blob  
5 Glob  
0 -  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
0

## Sample Output

1  
2  
Blob  
4  
Glob  
Blob  
7  
8  
Blob  
Glob  
11  
Blob  
13  
14  
BlobGlob  
16  
17  
Blob  
19  
Glob

# Problem 3: Digit Solver

## The Problem

Given three five-character strings, each on a separate line, find the values of the variables A, B, and C, such that the addition of lines one and two equals line three. You may find any correct solution to this expression.

Note: each string contains at least one of the three variables.

## Input

Three lines of five characters each

## Output

Each string, one per line, with the variables replaced by their appropriate values.

## Sample Input

C793A  
B83B5  
4ABAC

## Sample Output

17936  
28325  
46261

# Problem 4: Longest Bitlist

## The Problem

Given a list of numbers, find the longest vertical or horizontal list of binary digits (bits). Picture the numbers as a table of bits, and then search the table for the longest string of bits with the value of "1". Note: The table of numbers will always be square (i.e. if there are 5 numbers in the list then only the *least significant* five bits of the numbers will be considered).

## Input

The input will contain several problems. Each problem's input will be made up of the following: A line containing one integer which indicates how many numbers will be in the next line. A second line will contain integers (separated by one or more spaces) to be used in the square. The end of the list will be indicated by a -1. Do NOT process this value.

## Output

For each problem, output the number of *one* bits in the longest vertical or horizontal string, and indicate whether the list was "vertical" or "horizontal" or "both" (in case of a tie).

## Sample Input

## 3 5  1  3 2 3  1 1 1  -1

## Sample Output

3 vertical   
2 both   
1 both

# Problem 5: Missing Letters

## The Problem

The owner of a local newspaper is concerned that his type setting equipment is not functioning properly. He suspects that some of the characters never print out. You have been hired to write a program which will read in a file of text and output a list of all characters which are not in the text. You are to check for all ASCII characters between a blank and lower case z (this happens to be decimal values between 32 and 122.

## Input

A series of lines of text, terminated by EOF.

## Output

List of characters which do not occur in the text. This list should appear as one contiguous string from lowest to highest valued character.

## Sample Input

## !@#$%^&\*()\_+-=` abcdefghijklmnopqrstuvwxyz A BCDEFGHIJKLMNOPQRSTUVWXYZ

## Sample Output

"',./0123456789:;<>?[\]

# Problem 6: Problem Shmoblem

## Background

A common form of word-play indicating disdain or mockery is so-called "Shm-reduplication."

A word (typically a name or other word central to an opponent's argument) is repeated, and the second instance is mangled according to the following (informal) rules:

* Any leading consonent cluster is replaced with "shm" (e.g., "fancy-shmancy")
* If the word begins with a vowel, it is prefixed with "shm" (e.g., "ape-shmape")
* If the word already contains an "sh" pair, the prefix "sm" is used instead of "shm" (e.g., "crash-smash")
* The second (i.e., duplicated) instance is fully lower-cased (e.g., "Ashmont-smashmont")

## The Problem

Write a program that can transform free-form text using shm-reduplication.

## Input

Lines of plain [ASCII] text, terminated by end-of-file. "Words" (i.e., contiguous ranges of upper/lower-case alphabetic characters) *immediately preceeded* by an asterisk (\*) are to be transformed and printed according to the above rules. You may assume that every \* will be *immediately* followed by at least one alphabetic character.

## Output

Lines of plain text, with \*-tagged words transformed and *all other characters copied verbatim to the output*.

## Sample Input

We went to a \*fancy restaurant.  
  
"\*Ashmont," he snorted.  
  
He's just a baby!  
 \*Baby! He's 5 years old!  
  
Commander Spock says that is illogical.  
 \*Spock! I'm a doctor, not a potato!

## Sample Output

We went to a fancy-shmancy restaurant.  
  
"Ashmont-smashmont," he snorted.  
  
He's just a baby!  
 Baby-shmaby! He's 5 years old!  
  
Commander Spock says that is illogical.  
 Spock-shmock! I'm a doctor, not a potato!

# Problem 7: Numbers in Circles1

## Background

In Figure 1, each circle contains a number. Each pair of circles is joined by a line on which is written the sum of the numbers in the two circles. That is, 3 + 8 = 11, 3 + 7 = 10 and 7 + 8 = 15. In figure 2, the number on each line again shows the sum of the numbers in the circles joined by that line, but the numbers have been left out of the circles. Can you work out what the missing numbers are?

## The Problem

Suppose that instead of writing the sums of the numbers on the lines, we were to write their products. For instance figure 3 shows that A × B = 2, B × C = 8, C × D = 12, D × E = 15, and E × A = 5.

Can this new problem be solved similarly?

|  |  |  |
| --- | --- | --- |
| **Figure 1** | **Figure 2** | **Figure 3** |

## Input

The first line of input will be the number of problems. Each problem will occur on its own line. The first character on a line will be a '+' or 'x' indicating whether it is an addition or a multiplication problem. Next there will be an odd integer *N* ≥ 3, followed by *N* positive integers corresponding to the sum/product of the values in consecutive circles.

## Output

For each problem, output one line that contains the *N* nonnegative integers (ordered so that the sums/products are correct). If no solution exists for nonnegative integers, output "no solution" instead.

## Sample Input

3

+ 3 11 15 10

x 5 2 8 12 15 5

+ 3 9 1 2

## Sample Output

3 8 7

1 2 4 3 5

no solution

1Adapted from *Numbers in Circles* by John Parker and published in Mathematics in School November 2005.

# Problem 8: Non Attacking Queens

## Background

In chess, the queen can move vertically, horizontally, or diagonally as far as desired.  A queen is said to attack another chess piece if the queen can move to the square that the other piece is occupying.  For any integer *n*, greater than or equal to 4, *n* non-attacking queens can be placed on an *n* x *n* board.

## The Problem

For a given input *n*, output the number of different placements of *n* non-attacking queens on an *n* x *n* board.

## Input

The input will consist of an unspecified number of lines each containing an integer n (n < 20) which is the number of rows and columns in the chessboard. The last line will be 0 (do not process this line).

## Output

For each input line, output the number of different placements of n non-attacking queens on an *n* x *n* board.

## Sample Input

4

5

0

## Sample Output

2

10

# Problem 9: Cat in the Hat

## Background

(An homage to Theodore Seuss Geisel)

    The Cat in the Had is a nasty creature,   
    But the striped hat he is wearing has a rather nifty feature.

    With one flick of his wrist he pops his top off.

    Do you know what's inside that Cat's hat?   
    A bunch of small cats, each with its own striped hat.

    Each little cat does the same as line three,   
    All except the littlest ones, who just say "Why me?"

    Because the littlest cats have to clean all the grime,   
    And they're tired of doing it time after time!

## The Problem

A clever cat walks into a messy room which he needs to clean. Instead of doing the work alone, it decides to have its helper cats do the work. It keeps its (smaller) helper cats inside its hat. Each helper cat also has helper cats in its own hat, and so on. Eventually, the cats reach a smallest size. These smallest cats have no additional cats in their hats. These unfortunate smallest cats have to do the cleaning.

The number of cats inside each (non-smallest) cat's hat is a constant, *N*. The height of these cats-in-a-hat is   times the height of the cat whose hat they are in.

The smallest cats are of height one;   
these are the cats that get the work done.

All heights are positive integers.

Given the height of the initial cat and the number of worker cats (of height one), find the number cats that are not doing any work (cats of height greater than one) and also determine the sum of all the cats' heights (the height of a stack of all cats standing one on top of another).

## Input

The input consists of a sequence of cat-in-hat specifications. Each specification is a single line consisting of two positive integers, separated by white space. The first integer is the height of the initial cat, and the second integer is the number of worker cats.

A pair of 0's on a line indicates the end of input.

## Output

For each input line (cat-in-hat specification), print the number of cats that are not working, followed by a space, followed by the height of the stack of cats. There should be one output line for each input line other than the "0 0" that terminates input.

## Sample Input

216 125   
5764801 1679616   
0 0

## Sample Output

31 671   
335923 30275911

# Problem 10: The Webster Ranking

## Background

Mr. Webster is teaching his elementary school students how to sort lists of words. He knows some lists of words are harder to sort than other lists. The difficulty of sorting words depends on two things: the length of the list and the difference between the words. For example, consider the two lists:

|  |  |
| --- | --- |
| List 1 | List 2 |
| duck  cat  mouse | Computer  computation  completion  computers  competition |

The first list is significantly easier to put into alphabetical order than is the second list. It has fewer words and you only need to check the first letter of each to determine the relative ordering.

## The Problem

Mr. Webster has asked you to write a program to compute the difficulty of sorting any list of strings. The ranking is defined to be the sum of the number of word in the list and, for each pair of adjacent items in the sorted list, the number of letters it takes to distinguish between the two items. The Webster ranking of List 1 is then 5: there are 3 words in the list and it takes just one letter to distinguish between the pair 'cat' and 'duck' and the pair 'duck' and 'mouse.'

To compute the Webster Ranking of List 2, look at the sorted list:

|  |  |
| --- | --- |
| Sorted list 2 | Number of letters to distinguish the pair |
| competition |  |
| completion | 5 |
| computation | 5 |
| Computer | 7 |
| computers | 9 |

There are five words in the list, so the Webster Ranking of List 2 is 31 = (5+5+7+9)+5.

## Input

The input to your program will be one or more lists of words. The last word in each list is LAST and should not be processed. There will be between 0 and 100 words in each list. Each list has words arranged one per line, starting in the first column of each line. A word is a sequence of 1 to 25 letters. Case is not significant. Words will not be duplicated in any list.

The end of all input is indicated by a set containing *just* the word END. This list should not be processed.

## Output

For each list, print the list number and the Webster ranking of the set in the format shown below. Have a blank line after each line of output.

## Sample Input

duck   
cat   
mouse   
LAST   
Computer   
computation   
completion   
computers   
competition   
Last   
End   
last

## Sample Output

List 1 has Webster ranking = 5   
List 2 has Webster ranking = 31