Important Contest Instructions!!

Please read the following instructions carefully. They contain important information on how to package and submit your solutions to the judges. If you have any questions regarding these instructions, please ask a volunteer before the start of the contest.

Program Input/Output:
Programming tasks that require interaction with the program should prompt for user input to the screen (STDOUT) and read input from the keyboard/screen (STDIN).

For non-interactive tasks, you will have two options for reading the input data. Sample data for some problems will be provided to you in a file named **probXX.txt**, where 'XX' is the problem number.

Your solution may read the input from the file probXX.txt programmatically—that is, using the File I/O constructs of your programming language.

Most problems will accept input directly from the keyboard (STDIN) instead of performing file operations. For programs with large inputs, this can become very tedious. However, an easy way to enter large amount of data is by redirecting the contents of a file to your program at runtime. For example, a file named **prob01.txt** can be redirected to STDIN of your program using syntax like this:

```
%> java prob01 < prob01.txt
%> java –jar js.jar prob01.js < prob01.txt
%> python prob01.py < prob01.txt
%> prob01.exe < prob01.txt
```

In this example you are executing prob01 and sending the contents of the file prob01.txt to the STDIN of your program. Your program should behave exactly as it would if you were typing the input at the keyboard.

> ***Tip:*** *When entering input directly from the keyboard, type 'Ctrl-Z' <return> to signal the end of input to your program.*

All program output should always be sent to the screen (STDOUT).

Submitting your Programs

Interpreted Programs (JAVA, JavaScript, Python). Your program must be named probXX.java / probXX.js / probXX.py, where 'XX' corresponds to the problem number. Please submit only the source (.java, .js or .py). For java, the main class must be named probXX. Note the capitalization. All main and supporting classes should be in the default (or anonymous) package.

Native Programs (C. C++, etc). Your program should be named probXX.exe, where 'XX' corresponds to the problem number.

You are <u>strongly</u> encouraged to submit Problem #0 (on the next page) <u>prior</u> to the start of the competition to ensure that your build environment is compatible with the judges'.

problem 0

Sweet Sixteen

1 point

Introduction

The sole purpose of this problem is to allow each team to submit a test program to ensure the programs generated by their computer can be judged by our judging system. Each team is **strongly** encouraged to submit this problem **prior** to the start of the competition – hey, it's basically a free point!

Your task for this test program is to write the classic "Hello World!" program with a slight twist. Simply print "16 Great Years of CodeWars!" to the screen.

Sample Output

```
16 Great Years of CodeWars!
```

Selling Shirts

1 point

Introduction

A woman who makes and sells clothing has been selling shirts at a local market. She sells these shirts for $8 more than the cost of the materials and the rental for the booth at the market is $95 per day. Write a program to calculate the total profit P that she makes in one day based on the number N of shirts sold.

```
    P = 8 * N – 95
```

Input

The input is the number of shirts sold in one day. On a really bad day, this number could be zero!

```
31
```

Output

The program must print the profit for the day's sales. This value may be positive or negative depending on the value of N.

```
153
```

Rocket Science

2 points

Introduction

You can become a rocket scientist today! Model rockets are inexpensive, easy to build, and can reach heights of a few thousand feet.

If you want to calculate the altitude your model rocket will reach, at some point in the calculations you'll probably need to know the mass of your rocket. You can weigh the rocket on a scale, but you may find that your scale shows weight in ounces. That's all good, but the rocket motor's thrust typically is provided in newtons, which is a metric system unit. If so, you may need to convert the ounces to grams before you can calculate the peak altitude.

The conversion is pretty easy using this formula:

```
    1 ounce = 28.3495 grams
```

Input

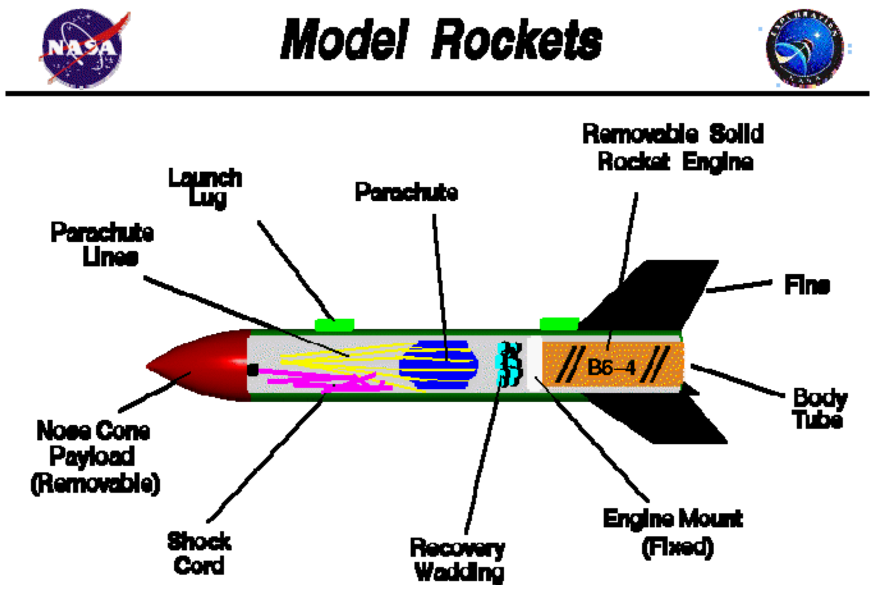
The input is the weight of the rocket, in ounces.

```
11.3
```

Output

The program must print the mass of the rocket in grams as shown below. The output must match the expected value to within +/- 1 gram.

```
320.3496
```



Notice: This programming problem is not endorsed by NASA, the National Association of Rocketry, or any other rocket-related organization.

problem **3**

Weight Calculator

3 points

Introduction

Did you know that if you were standing on the moon you'd only weigh about 1/6 of your weight on Earth? The surface gravity of celestial bodies (moons, planets, Pluto, etc.) in the solar system varies widely according to the body's mass and radius.

Write a program to compute a person's weight on the surface of a celestial body.

Input

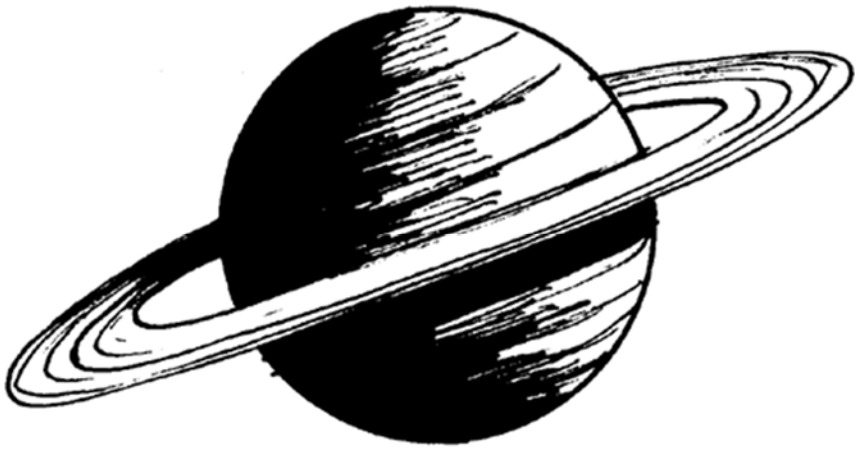
Each line of input will contain a person's name, their weight (in pounds, on Earth), a single-word name of a celestial body, and a conversion factor for the surface gravity on that body. The last line of input is the word “END” followed by three zeros.

```
Fred 179.0 Luna 0.1654
Layla 131 Mars 0.376
Pat 145.2 Neptune 1.14
Rajavel 156.4 Ganymede 0.146
END 0 0 0
```

Output

The program must convert each weight and print the result using the format shown below. Weights must match the expected values within +/- 1 pound.

```
On Luna, Fred would weigh 29.6066 pounds.
On Mars, Layla would weigh 49.256 pounds.
On Neptune, Pat would weigh 165.528 pounds.
On Ganymede, Rajavel would weigh 22.8344 pounds.
```



Go Ballistic

3 points

Introduction

Ballistics (gr. βάλλειν /BALL•ein/, "throw") is the science of mechanics that deals with the flight, behavior, and effects of projectiles. You can calculate the distance that an angry bird (or golf ball or other object) will travel if it is thrown (launched, catapulted, spit, etc.) from a fixed location over a flat surface using the following formula:

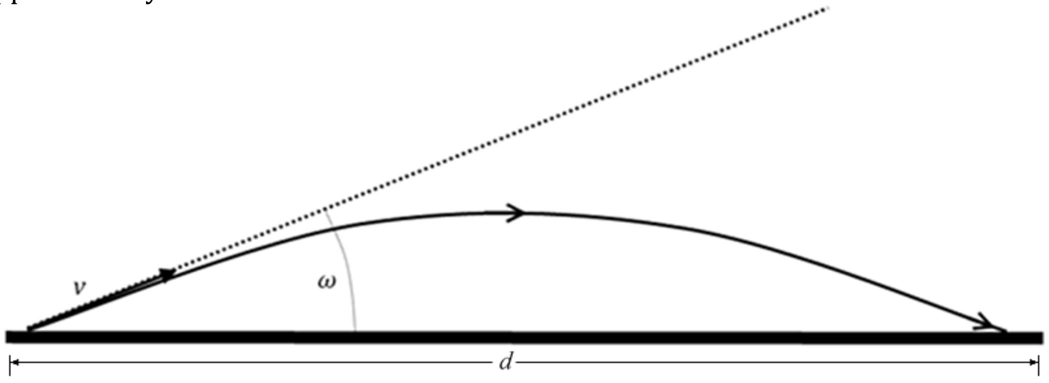
$$d = v^2 \times \sin(2\omega) / g$$

Here v is the initial (thrown) velocity, ω is the angle above horizontal (i.e. the ground), and g is the acceleration of gravity. The acceleration of gravity at the Earth's surface is approximately 9.80665 m/s^2.

For this program we'll assume that air resistance and the curvature of the Earth are insignificant. Also, we'll measure in units of meters, seconds, and degrees. Your language's *sin*() function probably expects angles in radians instead of degrees. If so, you can convert degrees to radians using this formula:

$$radians = \pi \times degrees / 180$$

Here π is approximately 3.14159265.



Input

The input will be two floating-point numbers, each on a separate line. The first is the initial velocity of a projectile in meters per second and the second is the launch angle in degrees.

```
63.9
65
```

Output

The program must print the distance in meters that the projectile travels before striking the ground. The answer must match the expected value within +/- 1 meter.

```
318.9591
```

Tribonacci Numbers

3 points

Introduction

You may recall that Fibonacci numbers are formed by a sequence starting with 0 and 1 where each succeeding number is the sum of the two preceding numbers; that is, F[n] = F[n-1] + F[n-2] with F[0] = 0 and F[1] = 1.

Tribonacci numbers are like Fibonacci numbers except that the starting sequence is 0, 1 and 1 and each succeeding number is the sum of the three preceding numbers; that is, T[n] = T[n-1] + T[n-2] + T[n-3] with T[0] = 0, T[1] = 1 and T[2] = 1.

The first eleven terms of the Tribonacci sequence are 0, 1, 1, 2, 4, 7, 13, 24, 44, 81 and 149.

Input

Each line of input is an integer. The maximum possible input value is 30. The last line of input is a -1.

```
3
9
11
0
-1
```

Output

For each non-negative input, the program must use the integer as an index to the Tribonacci sequence and print the Tribonacci number corresponding to that index.

```
2
81
274
0
```

Triangle Area

4 points

Introduction

One formula for the area of a triangle is:

$$\text{Area} = \frac{a \times b \times \sin w}{2}$$

Where *a* and *b* are the lengths of two sides and *w* is the angle between those same two sides. The angle *w* can be computed in two steps, first by calculating *cos w* using the length *c* of the third side in this formula:

$$\cos w = \frac{a^2 + b^2 - c^2}{2 \times \mathrm{a} \times \mathrm{b}}$$

Then use your language's arc-cosine function to compute *w*.

Java:	Math.acos()
C++:	acos()
Python:	math.acos()
JavaScript:	Math.acos()

The formula for the distance between two points is

$$d = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$$

Input

Each line of input will contain the x and y coordinates (in that order) for three distinct points. The input ends with six zeros.

```
3.1415 2.7777 -3.9123 0.2133 0.4324 -11.111
-8.675309 1.41421 9.999 0.0001 9.999 1.41421
0.7071 7.732 2.718 -1.005 -6.931 0.866025
0.6125 0.03125 99.999 0.9125 99.999 -0.56875
0 0 0 0 0 0
```

Output

For each line the program must print the area of the triangle. Answers should be accurate to within ±1 of the expected value. For example, if the expected value is 13.2038, any answer between 12.2038 and 14.2038 will be considered correct.

```
45.5104
13.2038
40.2704
73.6081
```

problem **7**

A Question of Time

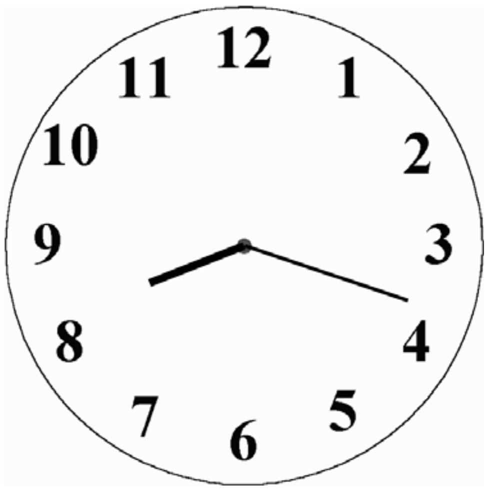
4 points

2013

Introduction

This program will calculate time based on a traditional round clock with hour and minute hands. The hour hand on this clock moves slowly from hour to hour; it does not jump from one hour to the next.

If both hands are on opposite sides of, and an equal distance from, the number six, then for a given number of minutes after the hour the program should determine the current time.



Input

Each line of input is an integer value indicating the number of minutes after the hour, in the range zero through fifty-nine. The input ends with a -1.

```
18
5
46
0
-1
```

Output

For each input, the program should print the time formatted digitally, i.e., "hours:minutes". Single-digit hours must not have a leading zero, but single-digit minutes must have a leading zero.

```
8:18
11:05
2:46
12:00
```

Distinct Letters

4 points

Introduction

The word “uncopyrightable” has the interesting property that each of the fifteen letters it contains is only used once. In the English language, there is no word longer that uses distinct letters (each letter used in the word is used once, and once only).

There is one other 15 letter word with this property: “dermatoglyphics” (from ancient Greek derma = "skin", glyph = "carving"), which is the scientific study of fingerprints.

Write a program to determine if a word uses distinct letters.

Input

The input is a series of lines, with one word per line. The input ends with a single period.

```
UNCOPYRIGHTABLE
FLIPPER
EXECUTABLE
UNPROFITABLE
QUESTIONABLY
WINDOW
TAMBOURINE
.
```

Output

The program must print each word with the correct message; either "`USES DISTINCT LETTERS`" or "`DOES NOT USE DISTINCT LETTERS`.”

```
UNCOPYRIGHTABLE USES DISTINCT LETTERS
FLIPPER DOES NOT USE DISTINCT LETTERS
EXECUTABLE DOES NOT USE DISTINCT LETTERS
UNPROFITABLE USES DISTINCT LETTERS
QUESTIONABLY USES DISTINCT LETTERS
WINDOW DOES NOT USE DISTINCT LETTERS
TAMBOURINE USES DISTINCT LETTERS
```

Counting Ones

5 points

Introduction

This program is a fun math problem!

Consider a function *f* (*n*) that takes a positive integer *n* and returns the number of 1's in the decimal representation of all the integers from 0 to *n*, inclusive. For example, f(13) = 6, for the numbers 1, 10, 11 (twice, for two 1s), 12 and 13. Notice that *f* (1) = 1.

Your task is to write a program that calculates *f* (*n*).

Input

Each line of input is a single integer, up to a maximum of 9999. The input ends with a -1.

```
13
1
999
23
1111
9997
511
-1
```

Output

For each non-negative input, the program must print the value of *f* (*n*).

```
6
1
300
13
448
4000
204
```

problem **10**

A Weighty Problem

5 points

2013

Introduction

Five school children are weighed in pairs on a scale. The five children can be paired in ten different weighs, er, ways. Given the ten paired weights, determine the weights of the individual children.

You may assume all the weights are integers.

HINT: Each child is weighed four times.

Input

Each line of input consists of ten integers. The input ends with ten zeros.

```
114 129 118 125 123 122 121 124 120 116
109 118 114 116 125 107 111 112 121 123
121 114 100 110 100 121 114 111 104 125
0 0 0 0 0 0 0 0 0 0
```

Output

The program must print the weight of each child in ascending order on a single line. Each child's weight must be an integer value.

```
56 58 60 64 65
52 55 57 59 66
45 55 55 59 66
```

Room For An Argument

5 points

Introduction

"An argument is a connected series of statements intended to establish a proposition."
"No it isn't."
"Yes it is! It's not just contradiction."
"Look, if I argue with you, I must take up a contrary position."
"Yes, but that's not just saying 'No it isn't'."
"Yes it is."
"No it isn't."

Write a program to reverse the negative of a sentence.

Input

The first line of input indicates the number of sentences to follow. Each line of input will contain a single sentence that will use the verb "is" and end in a period.

```
7
This is not an argument.
An argument is an intellectual process.
It is fair if you do not go.
The ferris wheel is not working.
A butterfly is beautiful, but litter is not.
A lady discerns that which is not elegant from that which is.
A lemur is a monkey and a grivet is a monkey but a chimp is not.
```

Output

For each sentence the program must print the sentence with the negative of the "is" inverted. That is, if the original sentence had the word "not" after the word "is", then the output should not. If the original sentence did not have the word "not" after the word "is", then the output sentence should print the word "not" immediately after the word "is". Is that clear?

```
This is an argument.
An argument is not an intellectual process.
It is not fair if you do not go.
The ferris wheel is working.
A butterfly is not beautiful, but litter is.
A lady discerns that which is elegant from that which is not.
A lemur is not a monkey and a grivet is not a monkey but a chimp is.
```

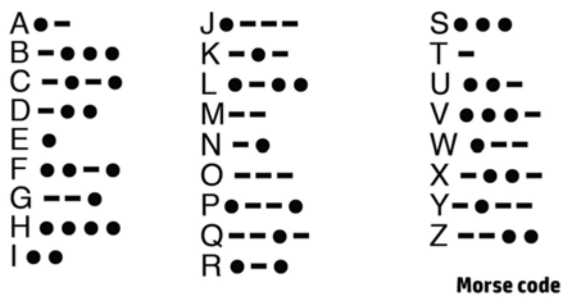
problem **12**

Morse Code Palindrome

6 points

Introduction

Let’s combine two of our favorite memes: Palindromes and Morse code.



A	•-	J	•---	S	•••
B	-•••	K	-•-	T	-
C	-•-•	L	•-••	U	••-
D	-••	M	--	V	•••-
E	•	N	-•	W	•--
F	••-•	O	---	X	-••-
G	--•	P	•--•	Y	-•--
H	••••	Q	--•-	Z	--••
I	••	R	•-•		

Morse code

Remember that a palindrome is a string that is the same forward and backward, ignoring any spaces or punctuation. So the word “ATE”, which is encoded as “•- - •”, is a palindrome. So is the phrase “WE TAN”.

Input

The input consists of one or more English words, written in upper case, per line. The input ends with a single period.

```
ELEGIZED
QUIRKILY
MERCURY
FACE A WINE
HAPPY DAY
FEVER REBEL
SOPRANOS
EMIT OLD UFO TIME
PROTEIN POWDER
ANNEXING
ENJOIN
.
```

Output

The program must convert the words to Morse code and determine if the pattern of dots and dashes is a palindrome. For each line of input, the program must print the line with the correct message, either "is a MCP", or "is *not* a MCP". Be sure to use the * characters to help the judges!

```
ELEGIZED is a MCP
QUIRKILY is a MCP
MERCURY is *not* a MCP
FACE A WINE is a MCP
HAPPY DAY is *not* a MCP
FEVER REBEL is a MCP
SOPRANOS is a MCP
EMIT OLD UFO TIME is a MCP
PROTEIN POWDER is *not* a MCP
ANNEXING is a MCP
ENJOIN is a MCP
```

problem **13**

Banner Reading

7 points

Introduction

The high school has a scrolling display that can advertise or display any message the administration wants to show. It is five pixels high, so the letters are drawn to fit in that space. The school newspaper has a webcam focused on the display and wants to translate the results into text so they can copy any important announcements.

Input

The input is a single banner string on 5 lines. It consists of letters, numbers and spaces. Each is 5 characters high and 3 characters wide. Letters and numbers are separated by 1 space. Words are separated by 5 spaces.

```
 ##  #  ##  ### # # ### ###  ##   ### ### ##  ###   ###  #   ## # #  ##
#   # # # # #   # # # # # # #       # # #  #    #   # # # # #   # # #
#   # # # # ### # # ### ##   #    ### # #  #  ###   ##  # # #   ##   #
#   # # # # #   ### # # # #   #   #   # #  #    #   # # # # #   # #   #
 ##  #  ##  ### # # # # # # ##    ### ### ### ###   # #  #   ## # # ##
```

Output

The program must print the ASCII string the banner represents.

```
CODEWARS 2013 ROCKS
```

```
Characters seen on the banner:

### ##  ### ### # # ### ### ### ### ###
# #  #    #   # # # #   #     # # # # #
# #  #  ### ### ### ### ###   # ### ###
# #  #  #     #   #   # # #   # # #   #
### ### ### ###   # ### ###   # ### ###

### ###  ## ##  ### ###  ## # # ###   # # # #   # #
# # # # #   # # #   #   #   # #  #    # # # #   ###
### ##  #   # # ### ### # # ###  #    # ##  #   # #
# # # # #   # # #   #   # # # #  #  # # # # #   # #
# # ###  ## ##  ### #   ##  # # ### ### # # ### # #

     #  ### ### ###  ## ### # # # # # # # # # # ###
    # # # # # # # # #    #  # # # # # # # # # #   #
### # # ### ### ##   #   #  # # # # # #  #   #   #
# # # # #     # # #   #  #  # # # # ### # #  #  #
# #  #  #     # # # ##   #  ###  #  # # # #  #  ###
```

For your convenience, the above characters can be found in the supplied file **BannerCharacters.txt**.

Hot Pad

9 points

Introduction

Consider a keypad used to enter security codes to gain access to a locked door. The keypad layout looks like a typical phone number pad:

```
1 2 3
4 5 6
7 8 9
* 0 #
```

The user enters a code with four numeric digits and presses # to open the door.

Input

The input will consist of a 2D array of temperature readings from an infrared scanner. The data values are the surface temperatures of the twelve keys on the keypad. Temperature differentials are caused by heat transfer from a person who recently entered a key code.

Example input 1

```
72.1 75.0 75.1
72.0 72.1 72.2
72.0 72.2 77.9
72.1 72.2 75.2
```

Example input 2

```
77.2  77.1  76.9
77.0  83.5  77.0
77.1  77.3  77.1
77.0  79.4  79.1
```

The program will use this data to determine the four digits used in the key code. For each data set the ambient temperature and the differential temperature will vary. Also, a small amount of noise (slight variations in temperature) should be expected.

Output

The program must print a list of all possible permutations of the four digits, in ascending numeric order, without duplicates.

Example output 1

```
2399
2939
2993
3299
3929
3992
9239
9293
9329
9392
9923
9932
```

Example output 2

```
0555
5055
5505
5550
```

problem **15**

We Love Stacking

10 points

Introduction

A well-known shipping company has a distribution center in central Texas. Packages arrive at the truck bay by conveyor belt. There is room at the truck bay for two stacks of packages, which is where they are stacked before being loaded onto the truck. You will write a program to do the stacking.

When a package reaches the end of the conveyor it must be placed on the stack with a top surface area that most closely matches the package size. However, a package must not be placed on a stack if its dimensions exceed the length or width of the top of the stack. The floor space for each stack is 120 cm by 120 cm and no packages will be too large to fit in that space. A package should be rotated left or right (but not turned up on its side) to fit. Also, a package cannot be placed on a stack if it would cause the stack height to exceed 200 cm.

If a package cannot be placed on a stack when the other stack space is empty, then the package becomes the bottom of the other stack. A new stack may not be started if the package will fit on the existing stack. If a package cannot be placed on either stack, then the tallest stack is loaded onto the truck and the package becomes the bottom of a new stack. After all the packages are processed, any remaining stacks are loaded into the truck, in order from the tallest to the shortest.

Input

The first line of input indicates the number of packages N that need to be processed. Each line after that will have measurements in centimeters for a single package. Each package is tagged with a single letter and described by three integers: length, width, and height, in that order, in centimeters.

Example 1 *Example 2*

```
13             14
A 59 109 49    A 101 83 67
B 84 76 58     B 82 63 72
C 91 54 42     C 58 79 70
D 43 75 38     D 81 55 39
E 35 45 25     E 80 101 65
F 76 89 40     F 57 85 51
G 26 29 16     G 75 46 68
H 102 84 46    H 97 69 63
I 75 75 75     I 65 91 64
J 71 71 51     J 56 82 72
K 101 82 42    K 65 71 69
L 78 88 58     L 44 97 63
M 77 87 57     M 91 43 65
               N 41 84 72
```

Output

The program must print one line for each stack that is loaded into the truck. The stacks must be printed in the order they are loaded. For each stack, the program must print the package tags from bottom to top.

Example 1 *Example 2*

```
A C D E        A B D
B G            C G
F I J          H I K
H K L          L M N
M              E F J
```

Family Tree

11 points

Introduction

In programming many problems can be solved by using parent-child relationships. This principle is used, for example, in the structure of HTML documents and the implementation of interface hierarchies. This approach is powerful because programs can apply rules to their data based on the notion of inheritance. Of course, the same data structure could also be used to track biological families, as in the recording and researching of people's family trees, or for selective breeding of food crops.

For this program, the relationship between two parents and one or more children is denoted by an expression like this: "`A + B : C , D , E .`" In this example A and B are the parents, while C, D, and E are the children.

There are two types of queries the program must be able to answer. The first is written like this: "`A > D ?`" which means, "is A an ancestor of D?" The second type of query is written like this: "`O ^ V ?`" which means, "do O and V share a common ancestor?"

Input

The first line of input is the number of parent-child expressions, followed by the expressions, one per line. Every expression will have two parents and at least one child. The next line is the number of queries, followed by the queries, one per line. Letters and symbols will be separated by a single space.

```
9
A + B : C , D , E .
F + G : H .
I + J : K , L .
C + H : M , N .
D + K : O .
L + E : P , Q , R , S .
N + Q : T , U , V .
O + S : W , X .
Y + H : Z .
7
K > O ?
F > W ?
B ^ U ?
O ^ V ?
A > Z ?
F > Z ?
X ^ Z ?
```

Output

The program must print each query, followed by the word TRUE or FALSE, indicating if the queried relationship exists. Note that a letter cannot be a parent/ancestor of itself.

```
K > O ? TRUE
F > W ? FALSE
B ^ U ? FALSE
O ^ V ? TRUE
A > Z ? FALSE
F > Z ? TRUE
X ^ Z ? FALSE
```

Return of the Vowels

12 points

Introduction

An eccentric scholar has been doing secretive research with grant money from a large corporate sponsor. Recently the scholar disappeared and no one has been able to locate him. The police and university officials believe there may be clues to what happened in his research notes, but in their present state the notes are difficult to read. It appears that the scholar had the odd habit of writing his research notes by either eliminating or rearranging the vowels in each word. Write a program to convert his notes to standard English spelling.

To complete this assignment, your program will need to read a file containing a dictionary of words. The name of this file is **wordlist.txt** and it is included with the other sample input files for this contest. The file contains one word per line, in alphabetical order.

Input

Each line of input contains a series of codes written in upper-case letters. Each code represents a single word. In some codes the vowels A, E, I, O, and U have been removed. In other codes all the vowels are present but scrambled. For all codes the consonants are all present and in the proper order. The input will not contain any punctuation marks. The input ends with a series of seven greater-than symbols.

```
TEH TSET SBJCTS AER GRWNG SSPCS NDA PSSMSTC
PRHPS EW SHULDO PEORMSI TEHM ECKA
SMTHNG AHS GOEN TRRBLY OWRNG
GLEEDOS AHS FOLEODD TEH NRCHMNT CNTEER IWTH DDLY NRTXN
>>>>>>>
```

Output

The program must decipher each code of the input and print the corresponding English word. If two or more English words match an input code, the program must print "?WORD1/WORD2/ETC?". If no words match an input code, the program must print the code with question marks, for example "?PSRGCHX?".

```
THE TEST SUBJECTS ?ARE/EAR/ERA? GROWING SUSPICIOUS AND PESSIMISTIC
PERHAPS WE SHOULD PROMISE THEM CAKE
?SMOOTHING/SOMETHING? HAS GONE TERRIBLY WRONG
?GLEEDOS? HAS FLOODED THE ENRICHMENT ?CENTER/CENTRE? WITH ?DEADLY/ODDLY? NEUROTOXIN
```

problem 18 Scramble Search

14 points

Introduction

Word searches are fun! And scrambling letters is fun! So how fun would it be to scramble a bunch of letters and then search for words? Too fun for, um... words!

Let's start with a few lines of text like this:

```
Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
```

Next, let's scramble the letters by picking a number of rows and rewriting the letters top-to-bottom, left-to-right. We'll ignore all non-letter characters in the process and convert the letters to upper-case.

```
TAVDEWNRUTEHEREGOLDOFID
WDEILODYLTLAOARIDODNAC
OSRNLOSIDRBNNVLSAOOERO
RDGAODOCNAODEEOTNKWAAU
OIEYWAROOVTBTLNODENSSL
```

How cool is that? Now, can you find the word GOLD? TOE? HAND? RAW? Can you write a program that could find them?

Input

The first line of input gives the number of rows in the scramble (up to a maximum of twenty-three). The line after the number of rows gives the number of lines in the input text, followed by the input text. The next line gives the number of search words, followed by the search words, one per line.

```
5
4
Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
6
GOLD
NODE
TOE
MELT
RAW
HERE
```

Output

First the program must print the scrambled text. Then the program must print each of the search words, followed by the row and column (in the scramble) of the word's first letter. If a word appears more than once, print all the row/column pairs with commas between each pair. Words may appear horizontal, vertical, backwards, or diagonal, just like a classic word search puzzle. If the program cannot find the search word, it must print the phrase "NOT FOUND" instead of the row and column. The top-left corner of the scramble is row zero, column zero.

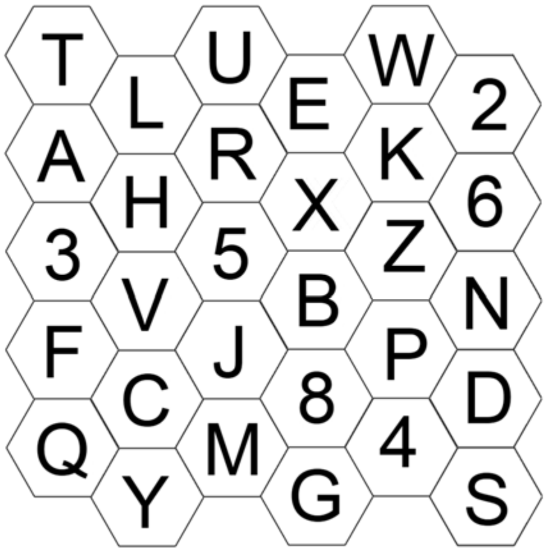
```
TAVDEWNRUTEHEREGOLDOFID
WDEILODYLTLAOARIDODNAC
OSRNLOSIDRBNNVLSAOOERO
RDGAODOCNAODEEOTNKWAAU
OIEYWAROOVTBTLNODENSSL
GOLD 0 15
NODE 4 14
TOE 3 15
MELT NOT FOUND
RAW 2 2, 4 6
HERE 0 11
```

Hexadistance

16 points

Introduction

Many classic table-top games are played on a hexagonal grid. It is often necessary to calculate the distance between two hexes. Write a program to calculate distances on the following grid. For example, the distance from U to B is 3 hexes, and from A to M is 4 hexes.



Input

The first line of input is the number of letter pairs. Each line after is a pair of letters separated by a space.

```
5
U B
M A
3 D
R E
Q K
```

Output

For each letter pair, the program should print the pair and the hex distance between them.

```
U B 3
M A 4
3 D 5
R E 1
Q K 5
```

Waffle Stacking

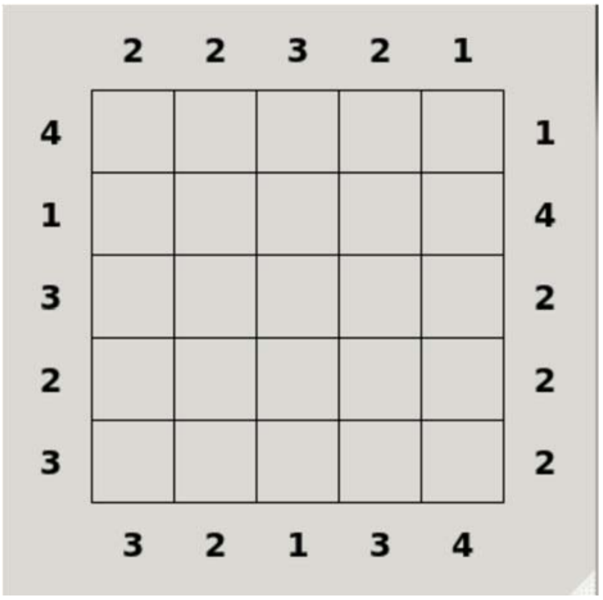
19 points

Introduction

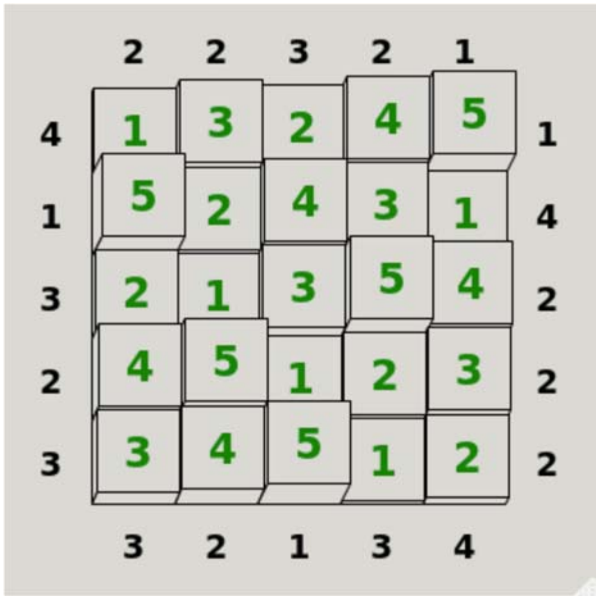
You have a 5x5 square grid. On each square of the grid you can build a stack of waffles, with its height ranging from 1 to 5. Around the edge of the grid are some numeric clues. Your task is to build a stack of waffles on every square, in such a way that:

- Each row contains every possible height of the waffle stack once
- Each column contains every possible height of the waffle stack once
- Each numeric clue describes the number of waffle stacks that can be seen if you look into the row or column from that direction, assuming that shorter waffle stacks are hidden behind taller ones. For example, a clue marked '5' indicates that the five waffle stacks must appear in increasing order (otherwise you would not be able to see all five waffle stacks), whereas a clue marked '1' indicates that the tallest waffle stack (the one marked 5) must come first.

Input



Output



The first line of input shows the clues for the top of each column. The next 5 lines of input show the clues for the left and right sides of the diagram for each row. The final line of input shows the clues for the bottom of each column.

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

Your program should print the 5x5 grid, with a space between each square in a row.

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

problem **21**

Alliterative Analysis

22 points

hp codewars 2013

Introduction

Alliteration is the repetition of a particular sound in the first syllables of a series of words or phrases, as in "If any so hardy in this house holds himself". Repetition of consonant sounds in the middle or at the end of words is sometimes distinguished by the term consonance, but for this program we'll just use the term alliteration no matter where the consonant is located.

Traditionally alliteration has been used in poetry and other literature. Today it is most commonly used in song lyrics but is also seen in magazine article titles, advertisements, common sayings, and a variety of other expressions.

Write a program to analyze alliteration. Remember that alliteration is about sounds, not letters themselves, so the program must observe the following rules:

- the first consonant sound in a word counts triple, even if the word starts with a vowel
- doubled consonants count as one (so "mammal" has two "m" sounds)
- c counted as s before e, i, or y
- c counted as k before consonants, a, o, or u
- c counted as k at the end of a word
- ck counted as a single k
- q counted as k
- x counted as both k and s
- ch is distinct from either c or h
- sh is counted as s
- th is distinct from either t or h
- tch is counted as ch
- gh is counted as h
- wh is counted as w

Input

The input will contain multiple lines of English words. Each line ends with a separate tilde ~ character and the last line of the input will contain a single tilde.

```
Once upon a midnight dreary while I pondered weak and weary ~
Mary sat musing on the lamp-flame at the table ~
Carrie's cat clawed her couch, creating chaos ~
The silken sad uncertain rustling of each purple curtain ~
Hot hearted Beowulf was bent upon battle ~
The daily diary of the American dream ~
The famous families lived along eleven fancy lanes ~
Charmed rich duchesses cherish diamonds ~
~
```

Output

The program must print the dominant consonant sound for each line of input and number of times the sound occurs, according to the rules given above. In case of a tie, print all dominant sounds. For the convenience of the judges, please convert all letters to upper-case.

```
W 9
M 8
K 15
S 8
B 9
D 9
L 13
CH D 8
```