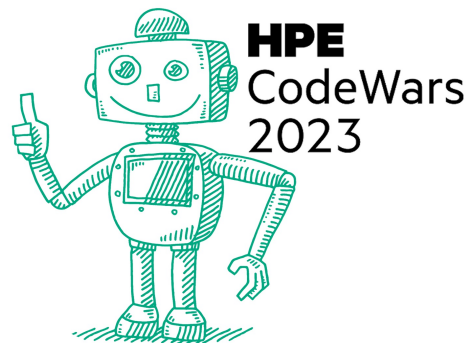


# Important Contest Instructions

**Please** read the following instructions carefully. They contain important information on how to run your programs and how to prepare them for submission.



## NEW for 2023

At least one problem requires working with directories. Details are below under "Directory/Folder input".

## Language Versions

The judges will be using the following versions for each of our supported languages.

Language	Version
Java	19.0.2
Python	3.11.2
C	11.2
C++	11.2

## Java

Your program file name must be **probXX.java** and your class name must be **probXX**, where **prob** is all lower case and **XX** corresponds to the two digit problem number. (ex. `public class prob01, prob01.java`).

Java programs can rely on the standard Java library.

## Python

Python programs can rely on the standard python library.

## C / C++

C or C++ programs must be submitted as source files and can rely on glibc/libm and the standard headers.

Download our "C/C++ Developers Guide" from <https://hpecodewars.org/downloads>

# Program Input

Most programs will require input. You have two options:

## File Input

Your program may read input from a file named **input.txt** for all problems. The file will be in the same directory as your program. Use "input.txt" as the filename and do not prefix it with a path.

| Download our "Guide to data file I/O" from <https://hpecodewars.org/downloads>

## Keyboard Input

Your program may read input from standard in (the keyboard). Do **not** include any prompts in your output. There are two options to provide input to your program via standard in (the keyboard).

- (1) The preferred option is to redirect the contents of a file to standard in of your program.

```
java prob01 < input.txt
java -jar js.jar prob01.js < input.txt
python prob01.py < input.txt
prob01.exe < input.txt
```

- (2) Otherwise you may type everything manually, or copy/paste from the packet or data set files.

| Tip: Type Ctrl-Z <return> to signal the end of keyboard input.

## Directory/Folder Input

For some programs, you will need to read from a folder named "files". The folder must be in the same directory as your program. The student dataset includes a new folder called "files". **MOVE IT** to the same directory as your program. Use "files" as its name and do not prefix it with a path. The judges also place the "files" folder in their testing directory.

| Download our "Guide to Directory I/O" from <https://hpecodewars.org/downloads>

## Program output

Your program must send the output to the screen (standard out, the default for any print statement). Do **not** include any prompts for input in the output! The only output for your program should be the expected output for the given problem.

## Testing your program

Test your program like a judge will with the Windows Batch script (checkProb.bat) in the student ZIP file. It tests your program with all student data sets for a given problem. It will redirect the data set file contents to standard in and create the **input.txt** file.

## Online Code Submission

Programs will be submitted through the contest site using a text box (with a large character limit, don't worry!) instead of a file upload. You will need to copy and paste the program from your local source code file into the text box. Once you select the problem number and language from the respective drop-downs, you can submit the program for judging. For Java, the main class must still be named probXX.

Ms. Marvel welcomes you to CodeWars, and invites you to embiggen your coding experience.



## Input

There is no data file to read in for this **(and only this)** problem.

## Output

Print to the screen the following greeting from Ms. Marvel:

```
Sometimes someone can come out of nowhere and do something amazing.
```



Russell Simmons is rebooting Def Comedy Jam for 2023. Please welcome the legendary founding comic acts as they take the stage to kick off the show!

## Input

You will receive one name on a single line. It will not contain spaces or punctuation (even if the proper spelling of the name would include them).

EddieMurphy

## Output

Print to the screen the following sentence. **The only part of the sentence which should change is the name!**

Now give it up for, EddieMurphy, give it up!

## Discussion

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

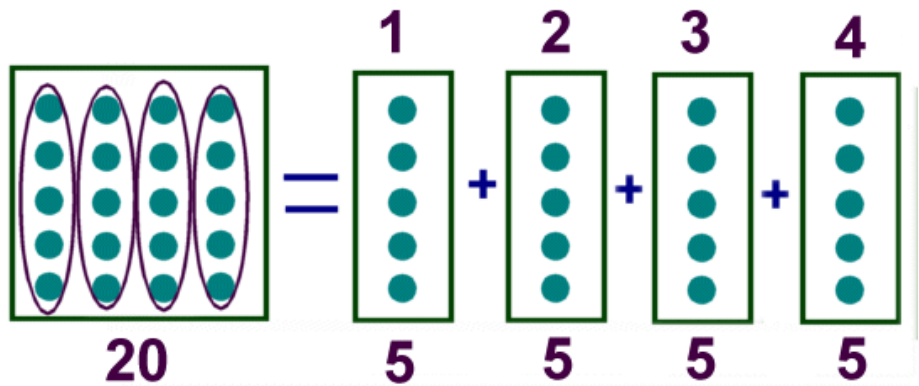
Input 1	Output 1
KevinHart	Now give it up for, KevinHart, give it up!

Input 2	Output 2
ChrisTucker	Now give it up for, ChrisTucker, give it up!

Input 3	Output 3
ChrisRock	Now give it up for, ChrisRock, give it up!



You and three friends keep finding treasure each weekend (that's a different story.) Every time, you count out the piles to make sure it's the same for all. Perhaps you could write a little program to let you know if the pile can be evenly split among the four of you or if you'll have to hide some away.



## Input

You will receive a list of integers, each less than 1 million. "0" on a line by itself ends the input.

```
87
9
15
62
40
19
7
88
0
```

## Output

For each number in the list, check to see if it is evenly divisible by 4. If so, print the division result as shown below. Since you're only printing exact quad divisions, there should not be a decimal point anywhere.

```
40/4 = 10
88/4 = 22
```

## Discussion

Reminder: have you run your solution against **all** of the student data sets?

The ponies are out of control! They have been making new ponies at an incredible rate. Princess Celestia is worried that the castle may soon not be big enough to hold everyone!

## Input

The input has 4 lines:

Line 1= Current pony population (integer)

Line 2= Number of ponies the castle can hold (integer)

Line 3= Rate of growth each year (decimal percentage)

Line 4= Number of years into the future Celestia is worried about (integer 2 through 1000).



```
217
1026
0.37
5
```

## Output

Use the PonyPopulationPrediction formula  $p = i(1+r)^t$  to find out if there will be enough room for the ponies, or if we will need to make the castle bigger. Output the expected population at the given year. Don't count fractions of ponies (truncate to just the whole number). Then show how much space for ponies will need to be added. Or, if there will be enough room, simply say there will be enough room like the examples below.

```
At the current rate of growth there will be 1486 ponies in 5 years.
Celestia will need to add space for at least 460 ponies!
```

## Discussion

PonyPopulationPrediction formula:  $p = i(1+r)^t$ . **p** is the final population. **i** is the initial population. **r** is the rate of growth. **t** is the total number of years.

Maximum start and end population size will be less than 10 million. The PPP formula is not a real mathematical population growth formula, and should only be used for counting ponies.

## Additional Examples

```
506      At the current rate of growth there will be 1022 ponies in 2 years.
2033     Celestia won't need to add space yet!
0.01
2
```

```
9035     At the current rate of growth there will be 118358 ponies in 10 years.
118358   Celestia won't need to add space yet!
0.31
10
```

Sometimes the best place to hide is in plain sight! Help find the crafty "odd-one-out" in the following groups of characters, which is a (shockingly) common problem in the "real" world.



## Input

There are 2 lines of input. The first line is the character to find. The second line is 10 to 1000 characters long, containing exactly one of the character from the first line.

```
1
llllllllllllllllllllllllllllllllllllllllllllllll
```

## Output

Find the zero-based index of the character from line one, in line two. "Zero-based" means that the first (left-most) position is 0.

```
l is at index: 21
```

## Discussion

If we mistakenly put more than one of the character to find in the second line, return the position of the first character found.

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

Input 1	Output 1	Input 2	Output 2
b dddddddddb	b is at index: 12	q pppppppppppppppp	q is at index: 15

Hexima is a math prodigy who finds it easier to think in base-6 than in decimal. She even came up with a new messaging app that uses base-6 to store data instead of binary. But her friends need your help sending her messages (they still write with letters, despite Hexima trying to teach them otherwise.) Convert their messages to heximal, so that Hexima can read them.



## Input

A one-line sentence (up to 50 characters):

HPE CodeWars Rocks!

## Output

Convert each character in the sentence to base-6 (heximal) using the base-10 (decimal) ASCII value for the character. 3 non-alphabetic characters are used: **space** (ASCII decimal 32), **exclamation mark** (ASCII decimal 33), and **period** (ASCII decimal 46). All other characters are A-Z (ASCII decimal 65-90) and a-z (ASCII decimal 97-122). Print all converted characters on one line, each as a 3-character block. Left zero pad if the base-6 number is less than 3 in length. For example,  $33_{10}$  (decimal) is  $53_6$  in base-6 (heximal). So print **053**. The final exclamation mark above is the **053** in the output below.

200212153052151303244245223241310311052214303243255311053

## Discussion

**Quick Review:** In base-10 there are 10 symbols: 0-9. In base 6, there are 6 symbols: 0-5. To show a number higher than 9 in base-10, you need to start over at zero and add 1 to the column to the left. It works the same way in base-6, except the highest you can count is 5. The columns are all powers of 6 instead of 10. So, to convert 33 (decimal) to base-6, first you would make a table of the powers of 6. Then, work left to right and determine how many of each column you need. Divide by 36 and place 0 in the 36s column. Divide by 6 and place 5 in the 6s column, and subtract 30 from 33, leaving us with 3. Place 3 in the 1s column as the remainder, so  $33_{10}$  is  $053_6$ .

	$6^2$	$6^1$	$6^0$
	36s	6s	1s
$33_{10}$	0	5	3
$65_{10}$	1	4	5

To convert the letter 'A' which is 65 (decimal): Divide by 36 and place 1 in the 36s column, and subtract 36 from 65, leaving us with 29. Divide by 6 and place 4 in the 6s column, and subtract 24 from 29, leaving us with 5. Place 5 in the 1s column as the remainder, so  $65_{10}$  is  $145_6$ .

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

Input Output  
Brady is The GOAT! 150310241244321052253311052220252245052155211145220053

Input  
Pink is a gateway color.

Output  
212253302255052253311052241052251241312245315241321052243303300303310114

Many families have chosen to print their favorite pictures, and to show them off, they are purchasing custom-built frames. **We-Frame-It** has been taking many orders every day. The online system collects the dimensions of each picture and the choice of frame.

There are 2 options for the frames. The simple Square-Corner frame has four lengths of wood, which are flatly-connected at each corner with a screw. Each beam is 2 cm thick and 2 cm wide. These frames can be covered with colored fabric to complement the picture. The Diagonal-Corner frame is more decorative. It has four lengths of wood, glued in a 45-degree angle at each corner. Each beam is 2 cm thick and 4 cm wide.



Neither frame covers any part of the picture, and all sides of the picture touch the frame.

Each order arrives in the shop, and the carpenters have to convert the picture dimensions and frame-choice into the length of wood needed to create the beams. For a square-corner frame, the side beams must be 2cm



(20mm) higher and lower than the picture so they can connect to the top and bottom beams. For a diagonal-corner frame, all beams must be 4cm (40mm) longer than the picture at **every corner**, so they can be glued to the neighboring beams.

Write a program to tell the carpenters the total beam-length needed for the frame for each order.

## Input

Each line of input has the height, length and frame-choice for an order. Height and length are positive integers in **millimeters**, and the frame-choice is either "S" or "D" for the Square and Diagonal Corner. The maximum of either distance is 10,000 mm. The final line is "0 0 S" to mark the end of input.

```
200 300 S
3171 5554 D
57 57 D
0 0 S
```

## Output

Your program must print the type of beam and total length in millimeters required for each order.

```
Square 1080
Diagonal 17770
Diagonal 548
```



Willy Wonka just discovered pop-rocks, and he is obsessing over the fizzy-fuzzy effects they have in soda. Wonka has found that when multiples of 5 pop-rocks are placed in soda, it generates a pleasant fizzing effect. However, when multiples of 9 pop-rocks are placed in soda, it generates a large fuzzy overflow. Wonka is trying to find combinations of pop-rocks which are multiples of both 5 and 9. That creates a nice foaming fuzzy carbonation bloom, without overflowing the cup. Help Wonka find the combinations.



## Input

You will receive a list of integers (int-32), one per line. Input will end when you encounter zero (0) on a line by itself.

```
1
5
16
27
-50
450
0
```

## Output

If the integer is evenly divisible by 5 print the integer followed by FIZZ. If the integer is evenly divisible by 9 print the integer followed by FUZZ. If the integer is evenly divisible by **both** 5 and 9, print the integer followed by FIZZ FUZZ! (include the exclamation mark)

```
5 FIZZ
27 FUZZ
-50 FIZZ
450 FIZZ FUZZ!
```

## Discussion

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

Input 1	Output 1	Input 2	Output 2
3	-10 FIZZ	-36	-36 FUZZ
-2	-15 FIZZ	7	2147483610 FIZZ FUZZ!
-7	-45 FIZZ FUZZ!	2147483610	18 FUZZ
-10		101	
-15		18	
-45		0	
0			



Tim loves adding more power! Everything can benefit from a little more power, right? Tim gave more power to a few integers and forgot to record how much, so he needs your help to get that data. Tim thinks he could do a lot more. I don't think so, Tim.



## Input

There are 2 lines of input. The first line is an integer (the base). The second line is another integer, the power after the base integer was raised to a whole-number exponent.

```
3
19683
```



## Output

Determine the exponent used to raise the base integer to its power, and output it as an equation. Use the "^" symbol to show an exponent. Use the spacing as shown in the examples.

```
3^9 = 19683
```

## Discussion

Some of the integers are very big. Use variables with enough room to store them. Student datasets have values larger than judge datasets.

## Additional Examples

Input 1	Output 1	Input 2	Output 2	Input 3	Output 3
16 268435456	16^7 = 268435456	2 32768	2^15 = 32768	62 238328	62^3 = 238328

The Binary Bot Bullets racing club is a collection of programs running in the CodeWars mainframe. They meet every millisecond or so and hold (virtual) races. We can't see them of course, but they let us know details about each race, and we can cheer them on (though never in real time.) Each bot has picked its own ASCII art representation, so we can print a snapshot of a race.



## Input

The first line is the count of race cars. The second line is the time (in minutes) of the race. On each car's line is their sustained speed (in miles per hour) followed by a space and then their car's ASCII art. All numbers are integers.

```
3
61
65 [ _o_o_>
92 ` _o_o_|
109 ` _o_o_>
```

## Output

Calculate the total miles (distance == rate \* time) each car has traveled at the end of the race. **Truncate** (cut off) to 2 decimal places. **DO NOT ROUND!** If you do not have 2 decimal places to output, add zeros. Then print the relative position of each car at the end of the race. First print the car's distance in parentheses. Then print one dash for every 5 miles traveled. Then one tilde (~) for every single remainder mile traveled. Then a pair of curly braces ( {} ) for every remaining quarter mile traveled. Ignore any remaining distance smaller than a quarter mile.

```
(66.08) -----~[_o_o_>
(93.53) -----~{}{}`_o_o_|
(110.81) -----{}{}{}`_o_o_>
```

## Discussion

Above, the first car travels 66.08 miles.  $65 / 5 == 13$ . So print 13 dashes. That leaves a remainder of 1.08. Print 1 tilde (~) for the 1 remaining full mile. The remainder is 0.08, smaller than a quarter mile, so you're done.

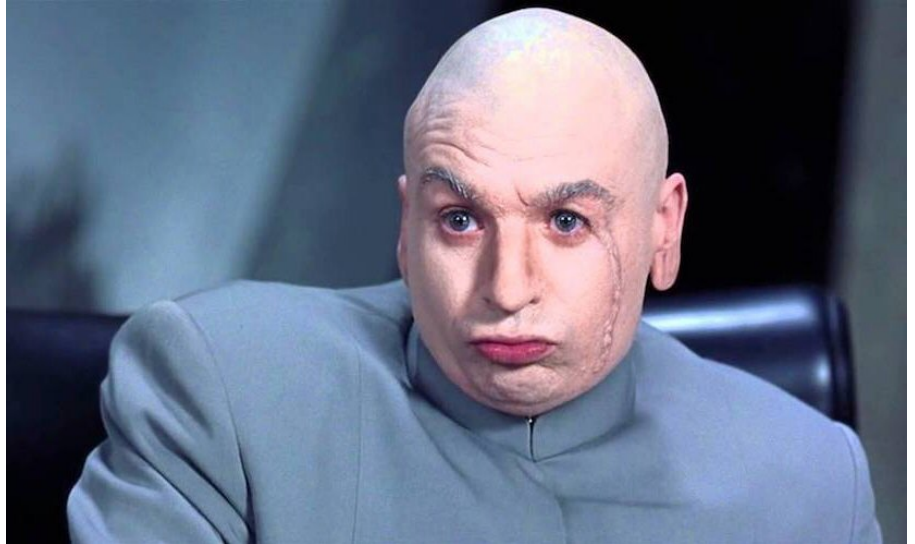
The second car travels 93.53 miles.  $90 / 5 = 18$  dashes. The 3 remaining full miles become 3 tildes (~~~). The remaining .53 miles becomes {}{} (one pair of curly braces for each quarter mile.) And the final .03 miles is ignored.

Reminder: have you run your solution against **all** of the student data sets?

## Additional Example

Input	Output
2	(125.00) ----- _o_o_>
100	(146.66) -----{}{}[_o_o_>
75  _o_o_>	
88 [_o_o_>	

New parents often refer to their baby's age in terms of months. E.G. she's 4 months old, or he's 11 months old. **Some parents** though (you know who they are) take it too far. It's not uncommon for parents of a toddler to say, "she's 16 months old," for instance. And then there are the people who just can't let go, and will use months no matter what. 😬



## Input

There are 3 lines of input. The first line will be the birth date of the baby in ISO format (yyyy-MM-dd). The second line will be the "current" date (e.g. the date to use in calculations). The third line will be the baby's name.

```
2009-11-13
2023-03-04
Ichiko
```

## Output

Output the *baby's* age in months as follows:

```
Ichiko is 159 months old
```

## Discussion

For this problem, all babies will be at least 2 months old and born before the 29th of their birth month. Also, all babies will be younger than 120 years. No fractions. Only count whole months.

Reminder: have you run your solution against **all** of the student data sets?

## Additional Example

Input	Output
2022-12-05 2023-03-04 Midas	Midas is 2 months old

You are in charge of arranging rides for everyone in your class to go on a field trip to the beach. You need to recruit enough adult drivers who have cars which can carry 4 students in the car (3 in the back, 1 in front).



## Input

The input is a positive integer less than 100, the number of students in your class who need a ride for the field trip.

33

## Output

Since each car can hold a maximum of 4 students, determine how many cars are needed to bring all the students on the field trip. Always fill as many cars as possible. Output the count of full cars and partial cars as shown in the examples below. If both full and partial cars are needed, separate them with a comma and a space. Do not include "0 full" or "0 partial" cars. If only 1 full car is needed, print "car", not "cars".

8 full cars, 1 partial car

## Discussion

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

Input 2	Output 2	Input 3	Output 3	Input 4	Output 4
96	24 full cars	3	1 partial car	5	1 full car, 1 partial car

In the year 2091 space-faring civilizations use a wide range of temperature measurement systems named after their inventors from the intellectual revolution in 2056. Most everyone is using the Celery system, except for the U.S. (because reasons) which uses the Fairhead. Scientists almost exclusively use Kevin. (You would think it would be really confusing to anyone who used the prior Celsius, Fahrenheit, Kelvin scales, but few seem to remember those.) Your task is to make a little widget which will convert among all three temperature scales.



Use the formulas listed below to convert between the temperature readings (it's the year 2091, things have gotten weird).

### Conversion Formulas:

```
F->C: C = (Fairhead - 32) / 1.7
F->K: K = (Fairhead - 32) / 1.7 + 273.99
C->F: F = (Celery × 1.7) + 32
C->K: K = Celery + 273.99
K->C: C = Kevin - 273.99
K->F: F = ((Kevin-273.99) × 1.7) + 32
```

### Input

You will receive a single temperature reading as an integer (from -200 to +500) followed by a space, and then either a C, F or K to denote Celery, Fairhead or Kevin.

```
82 F
```

### Output

Repeat the temperature reading and add a parenthesis notation after it for the other two scales, in alphabetical (C,F,K) order. Pay close attention to the single-spaces and the comma. Truncate (DO NOT ROUND) the conversion to a single decimal point AFTER performing the math.

```
82 F (29.4 C, 303.4 K)
```

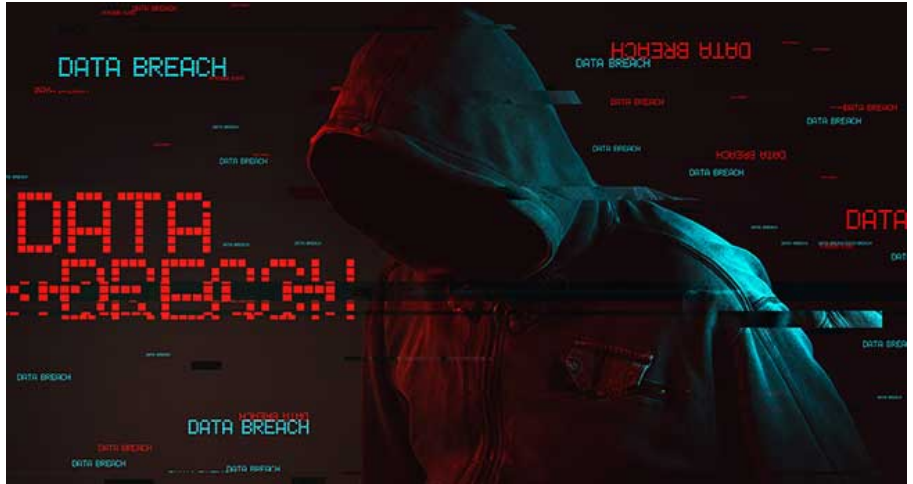
### Discussion

If you use built-in library conversion functions for 2023 Temperature scales, you will get the wrong answers!



Uh oh, the guy in the hoodie is nearby. Better be careful.

Of course, clothing isn't really a reliable indicator of who might be untrustworthy. It could be a man in a suit and tie, or a woman in a dazzling outfit, or a tall man in black with a cape and mask with a respirator (that last might be more obvious.) But since we cannot be sure who might be planning harm, we have to protect our data. (Maybe the guy in the hoodie is just cold?)



## Input

Each line will be either a credit card (#####-#####-####-####) or social security number (###-##-####), up to 10 lines. Zero on a line by itself marks the end of the input.

```
1234-5678-9098-7654
123-45-6789
0
```

## Output

Your job is to use this dangerous data for its purpose, and then delete all but the last 4 digits of the data, and only store the value masked with x's. Under no circumstances should credit card or social security numbers ever be stored in any data storage system you manage. Ever. Print the resulting data after deleting most of it.

```
xxxx-xxxx-xxxx-7654
xxx-xx-6789
```

## Discussion

Reminder: have you run your solution against **all** of the student data sets?



Fibonacci was a mathematician born in the late 12th century. He is most well known for publishing the "Fibonacci Sequence", 1,2,3,5,8,..., where each number in the sequence is the sum of the preceding two numbers.

Not to be outdone, his third cousin Tribonacci invented another sequence, where each number is the sum of the preceding three numbers. The Tribonacci Fan Club has a fascination with the sequence and has created their own ID numbers as the difference between any two Tribonacci numbers.

The Tribonacci sequence is defined:  $T(0)=0$ ,  $T(1)=1$ ,  $T(2)=1$ ,  $T(3)=2$ .  
 $T(k)=T(k-3)+T(k-2)+T(k-1)$ .

With this definition, the Tribonacci Numbers for elements 0 thru 10 are: 0, 1, 1, 2, 4, 7, 13, 24, 44, 81, 149. Help each member of the Tribonacci Fan Club pick their ID number (the difference between two Tribonacci Numbers.)



## Input

Each line has two integers M and N ( $M \geq N$ ) from 0 to 35. M and N are the Element numbers in the Tribonacci Sequence above. The last line is "0 0" to mark the end of the input.

```
6 6
6 0
35 0
35 34
10 5
9 8
0 0
```

## Output

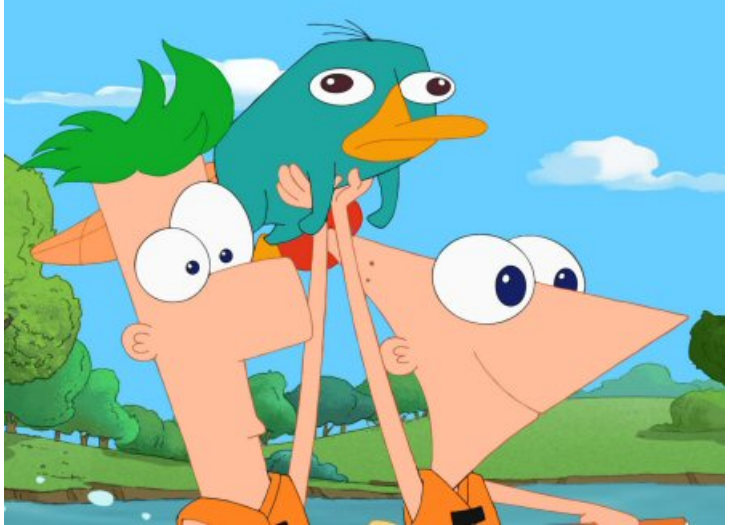
For each input line, print the difference between the two Tribonacci numbers.

```
0
13
615693474
280947697
142
37
```

## Discussion

- For example, line 5 of the input is "10 5".  $T(10)=149$ .  $T(5)=7$ . So  $T(10)-T(5)=142$ .
- Tribonacci Element 35 (615693474) is the largest number your program will need to hold.

Phineas and Ferb are messing around with the timeline again and they've lost track of which dates are real. Help Baljeet check their dates against reality. They will give him part of a date and a range of years they think it should exist. Find the years where that date could actually exist.



## Input

You will receive a date-part on the first line, in the form of: DDD MMM NN where DDD = the day of the week (MON, TUE, WED, etc.), MMM is the month (JAN, FEB, MAR, etc. (always first 3 letters)) and NN is the day number (01-31). On the second line you will receive a year range in the form of YYYY-YYYY, inclusive.

```
TUE JUL 17
1950-2022
```

## Output

Output all years where the given date part existed. E.G. July 17th didn't fall on a Tuesday in 2015, 2016 or 2017, but it did in 2018. Output all years where the date given was actually on the calendar, in ascending order. There will always exist at least one year where the input date-part matches the range of years given.

```
1951
1956
1962
1973
1979
1984
1990
2001
2007
2012
2018
```

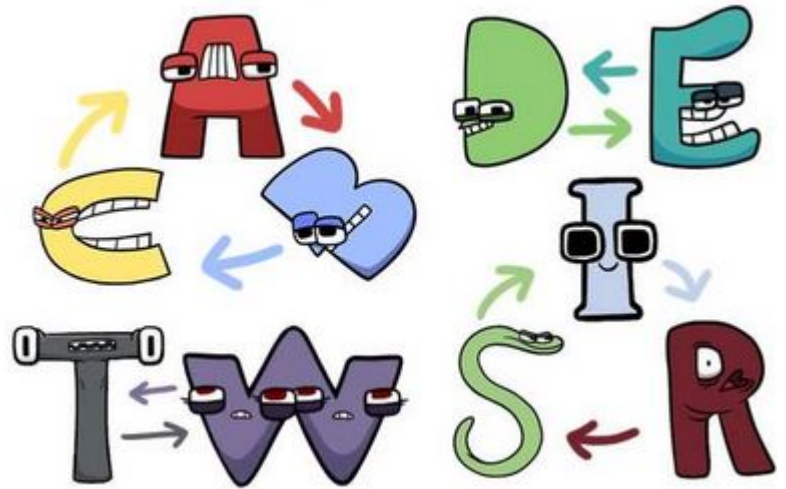
## Discussion

It will be difficult to do this problem without using your language's date/time library. However, it is not impossible. You can take today's date and simply keep subtracting days in your code to do the same comparisons that a date/time library would allow. Don't forget leap years! **Hint:** Creating a date which cannot/did not exist, will likely cause your code to throw an exception and crash. You will need to anticipate that.

## Additional Examples

Input 1	Output 1	Input 2	Output 2
SAT FEB 29 1890-2021	1896 1908 1936 1964 1992 2020	MON DEC 02 1866-1915	1867 1872 1878 1889 1895 1901 1907 1912

"Everybody Alphabetize!" And everybody groaned. "Okay, I'll make it more interesting. Pick 2 letters from our Scrabble bag." The letters B and W were picked. "Now, alphabetize, but every time you see a B, you must treat it as a W, and vice-versa." We still groaned, but it was a little bit more interesting.



## Input

The first line has 2 letters, separated by a space; these are the letters to be swapped. The next lines each have one of the words to be sorted. The word END on a line by itself signals the end of input. (Don't sort "END".)

```
B W
Boy
Tree
Walrus
Girl
Able
Away
Apple
END
```

## Output

Sort the list of words in ascending order, with the swapped letters changing the sort order. E.G. if A and Z are swapped, all Z-letter words should be listed first, and all A-letter words should be listed last. Note, sorting must take into account all letters in the words, not just their first letter. Output the swapped sorted list of words.

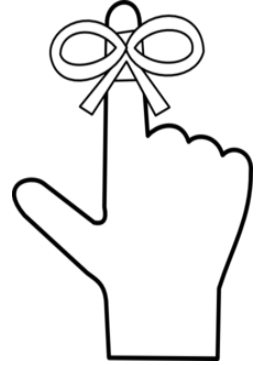
```
Away
Apple
Able
Walrus
Girl
Tree
Boy
```

## Discussion

Reminder: have you run your solution against **all** of the student data sets?

Could you repeat that? Could you repeat that? Could you repeat that? Could...

Studies show repetition can help with memory. That's why practice is important. But for us, repetition is best when applied to sentences (lines in a play) or words (spelling tests). It's not as obvious in long strings of letters or numbers, and that's where a computer is helpful. Can you teach a computer to find a repeating pattern? Maybe another day we'll teach it to remember things...



## Input

The input is one line with a repeating pattern of characters (letters and/or numbers, no spaces or punctuation). The pattern starts with the first character and repeats at least 3 times. Upper case and lower case letters are different.

```
ABBACABBACABBACABB
```

## Output

On the first line of your output, print the missing character(s) to complete the final section of the pattern. On the second line, print the full pattern (just once). Choose the shortest repeating pattern.

```
AC
ABBAC
```

## Discussion

Remember to choose the shortest pattern. For example, in "AbcAbcAbcAbcAbcAbcAbcA", both "Abc" and "AbcAbc" repeat at least 3 times. But choose the shortest pattern: "Abc".

Reminder: have you run your solution against **all** of the student data sets?

## Additional Example

Input	Output
12345123451234512345123451234512345123451234512345123451	2345 12345

J. Wellington Wimpy loves his hamburgers so much, he has one every day! Some days he's in a very picky mood and specifies the exact toppings he wants on his burger with the conjunction "and" – for instance, "cheese, ranch dressing, and tomatoes." On other days, however, he's in a more relaxed mood and is happy if his burger has any one of the toppings he's in the mood for that day. He lets the burger stand know this by the use of the conjunction "or" – for instance, "pickles, lettuce, or cucumbers." He's just finished making up his lists of toppings for the upcoming week but needs your help in translating his lists into proper English for the hamburger stand. Mr. Wimpy's happiness over the upcoming week depends on YOU! Good luck!



## Input

Each of his orders will have the word "AND" or "OR" in CAPS on the first line. The remaining lines will list up to 10 ingredients he's in the mood for, 1 ingredient per line. You'll need to translate these lists into a proper English clause. Example:

```
AND
cheese
ranch dressing
tomatoes
```

## Output

Note that Mr. Wimpy is a strong supporter of the Oxford comma so you'll need to make sure your string includes it (the Oxford comma is the last comma in a list of items; in this case, it's the comma that comes after the word "dressing"). **Oxford commas are not used** when there's just **two items** in a list. If there's just **one** topping on the list, **disregard the AND or OR**.

```
cheese, ranch dressing, and tomatoes
```

## Discussion

Also note that when AND or OR is required, you should print it out in lowercase letters, consistent with English rules of grammar, and not in CAPS as Mr. Wimpy has written it. Your output should not end in a period because this is a clause and not a sentence.

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

Input 1	Output 1	Input 2	Output 2
OR cheese ranch dressing	cheese or ranch dressing	AND cheese	cheese

Author: Kanwal Gill, M.D.

It's craft time again at CodeWars. We're going to start by making a frame for our group photo! We'll put a word into the shape of a square to make our frame. But we need to get an idea of how it will look, so we need your help.



## Input

You will receive 2 lines of data. The first line will give you an UPPERCASE word to use to make a square. The second line will tell you the size of the square, in letters. The largest number that will appear is 15. The smallest is 2.

```
BOX
6
```

## Output

Make a box out of the letters of the word by repeating them. Start from the upper left corner of the box. Continue clockwise until you reach the starting point. Stop before overwriting the first position.

```
BOXBOX
O B
B O
X X
O B
BXOBXO
```

## Discussion

Reminder: have you run your solution against **all** of the student data sets?

## Additional Examples

Input 1	Output 1	Input 2	Output 2
TREE 5	TREET E R E E R E TEERT	WIND 2	WI DN



As the saying goes, there are 10 types of people: those who understand binary, and those who don't. Help write a *show your work* binary conversion program that will teach younger students how to manually convert a decimal (base-10) number to binary (base-2).



## Input

The input is a single integer in decimal (base-10) ranging from 1 to 1023.

126

## Output

Show the steps to convert the decimal number to binary, working largest ( $2^9 = 512$ ) to smallest ( $2^0 = 1$ ), as follows: First walk down the list of numbers (512,256,128,...) which are columns/places in binary, and determine if the decimal number is larger than that column. If not, mark it with "=0". But if so, mark it with "=1" and subtract that column value from the decimal number. Keep going until you get down to zero. Once done, walk through the values marked as being used, and show the addition steps to get back to the decimal number. If only one binary column value is marked, there is no addition to show. Finally, starting with the highest "1" in your list, output the binary number.

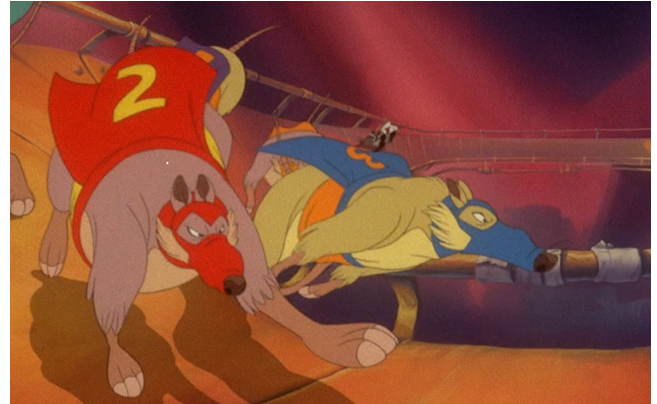
```
512=0
256=0
128=0
64=1
32=1
16=1
8=1
4=1
2=1
1=0
64+32 = 96
96+16 = 112
112+8 = 120
120+4 = 124
124+2 = 126
126 = 1111110
```

## Additional Example

Input	Output
40	512=0 256=0 128=0 64=0 32=1 16=0 8=1 4=0 2=0 1=0 32+8 = 40 40 = 101000

Reminder: have you run your solution against **all** of the student data sets? (There are a lot of them.)

Carface has started up the rat races again, and you have the job of programming the winner's marquee because Itchy is on vacation



## Input

The first line gives the total length of the race in kilometers. The next lines are a list of 3 to 5 racers and their progress on the track when one of them crosses the finish line in the format of NAME SSSS-SSSS:N.N, where S is the time index (in seconds) of the race, and N is how fast (rate) they are running at that time (in meters per second). Time intervals are separated by semi-colons (;). If a racer has a single rate during the race, they only show one time interval. Input ends with two zeros (00) on a line.

```
1.711
MIGHTY-MORRIS 0000-0050:3.9;0050-0092:1.2;0092-0714:2.2;0714-0726:8.0
RAT-O-WAR 0000-0726:2.3
SECRETARY-RAT 0000-0030:7.5;0030-0632:1.5;0632-0726:2.0
CHEEZE-WHIZ 0000-0033:8.4;0033-0726:1.9
SQUAD-CAR 0000-0050:3.9;0050-0092:1.2;0092-0714:2.2;0714-0726:8.1
00
```

## Output

Give the position and expected place (1st, 2nd, 3rd, 4th, or 5th) of each racer at the time the first racer crosses the finish line in the format: "NAME 1234m in 12.3min (1st)". Truncate the distance and minute value to one decimal place (do not round). Add a ".0" if a whole number. Print all racers in the order they were listed in the input.

```
MIGHTY-MORRIS 1709.8m in 12.1min (2nd)
RAT-O-WAR 1669.8m in 12.1min (3rd)
SECRETARY-RAT 1316.0m in 12.1min (5th)
CHEEZE-WHIZ 1593.9m in 12.1min (4th)
SQUAD-CAR 1711.0m in 12.1min (1st)
```

## Discussion

- distance = rate \* time
- rate = distance/time

## Additional Examples

Input 1	Output 1
<pre>1.9628 LIGHTNING 0000-0030:3.1;0030-0490:2.9;0490-0701:0.9 CRUZ 0000-0111:1.6;0111-0590:1.4;0590-0701:1.0 CHICK-HICKS 0000-0234:2.1;0234-0701:2.0 THE-KING 0000-0701:2.8 00</pre>	<pre>LIGHTNING 1616.9m in 11.6min (2nd) CRUZ 959.2m in 11.6min (4th) CHICK-HICKS 1425.4m in 11.6min (3rd) THE-KING 1962.8m in 11.6min (1st)</pre>

Input 2	Output 2
<pre>2.1154 DOPEY 0000-0500:1.1;0500-1111:1.0 SNEEZY 0000-0250:1.9;0250-0537:1.8;0537-1111:1.7 DOC 0000-0100:8.6;0100-0900:1.2;0900-1111:1.4 GRUMPY 0000-1111:0.0 00</pre>	<pre>DOPEY 1161.0m in 18.5min (3rd) SNEEZY 1967.4m in 18.5min (2nd) DOC 2115.4m in 18.5min (1st) GRUMPY 0.0m in 18.5min (4th)</pre>

It turns out that moving files around on a network is easy, but moving them around on a network at high speeds is hard (and expensive). That is because networks will only operate at the speed of the slowest part involved with the transfer. Help your boss purchase the right equipment for the office so that she is able to transfer files back and forth with your team without complaining to you that the transfer speeds are too slow.



## Input

You will receive combinations of network and computer parts in your input file. There will always be 6 lines of input. **Their order is not guaranteed.** The format of the data lines will always be: LABEL data, with one space between the label and the data. Example:

```
NICSRC 1.0
NICDST 1.0
ROUTER 2.5
SWITCH 0
HDDSRC 1000/1000
HDDDST 115/105
```

Where:

- NICSRC == "source NIC": the speed of your network interface card (NIC) in Gigabits[2]/second (always a decimal number with at least one integer value and always one decimal place.)
- NICDST == "destination NIC": the speed of the destination's NIC in Gigabits/second.
- ROUTER == "router": the speed of your router in Gigabits/second.
- SWITCH == "switch": the speed of any switches your traffic routes through in Gigabits/second. (zero if no switches present)
- HDDSRC == "source drive": source drive read/write speed in Megabytes/second[4]. (Yes, we know HDD technically stands for "hard disc drive", we are using it for solid state drives (SSD) too, for simplicity)
- HDDDST == "destination drive": destination drive read/write speed in Megabytes/second[4].

## Output

Output the following [1][3]:

- Maximum (theoretical) possible transfer speed in Megabytes/second (truncated to 1 decimal place).
- Identify the largest problem with transfer speed (what is causing the largest speed drop): source NIC, destination NIC, router, switch, source drive, destination drive

Example:

```
105.0
destination drive
```

## Discussion

[1] The **highest transfer speed** on your network will be **determined by the slowest device / equipment** on your network. So, if you have a 1 Gigabit router, and 1 Gigabit NICs on 2 computers on your network, and the source drive is a screaming fast PCIe 4.0 M.2 SSD rated at 7,450 MB/s read / 6,900 MB/s write, but the destination drive is an old platter-style physical hard disc drive only rated 115 MB/s read /105 MB/s write, then the fastest you will be able to transfer a file to that destination computer is as fast as that destination drive can write: 105 MB/s.

So you would output "105.0" and "destination drive" in that scenario, as it is the destination drive write speed causing the bottleneck.

[2] A bit is a zero or one, a single item of data. A byte is 8 bits, a collection of 8 items of data.

In modern computing (and advertising), base-10 is used when speeds and sizes of computing equipment are concerned. So, a "gigabit" is 1 billion ( $10^9$ ) bits. (A gigabyte is 1 billion bytes.) Because networking equipment is almost uniformly sold in units of Gigabits/second (Gb/s), but computers report transfer speeds in Megabytes/second (MB/s), you will need to do some conversions on the Gigabit units:

Byte Units: - 1 kilo =  $10^3 = 1,000$  - 1 mega =  $10^6 = 1,000,000$  - 1 giga =  $10^9 = 1,000,000,000$

Conversions - 1 Gb = 1,000,000,000 bits - 1,000,000,000 bits / 8 == 125000000 bytes - 125000000 bytes / 1,000,000 = 125 megabytes

[3] You are guaranteed that one of the items in the list of equipment will be the largest problem. E.G. there will never be a "tie"

[4] If an HDD is listed as the "source" drive (HDDSRC) then it is being **read** from, so only its **read** speed matters. If an HDD is listed as the "destination" drive (HDDDST) it is being **written** to, so only its **write** speed matters.

Reminder: have you run your solution against **all** of the student data sets?

## Additional Datasets

Input 2	Output 2	Input 3	Output 3	Input 4	Output 4
HDDSRC 5543/5383 HDDDST 7874/7606 ROUTER 2.6 NICDST 4.4 NICSRC 5.0 SWITCH 10.3	325.0 router	SWITCH 2.5 ROUTER 1.2 HDDDST 3451/3451 NICSRC 0.7 HDDSRC 4432/4309 NICDST 10.0	87.5 source NIC	SWITCH 7.1 NICSRC 9.4 HDDDST 6393/6393 NICDST 6.2 ROUTER 3.2 HDDSRC 222/200	222.0 source drive

Input 5	Output 5	Input 6	Output 6
ROUTER 4.4 NICSRC 3.4 SWITCH 1.6 HDDSRC 3458/3458 HDDDST 6036/6036 NICDST 4.6	200.0 switch	SWITCH 0 NICSRC 0.2 HDDSRC 200/180 ROUTER 0.7 NICDST 0.1 HDDDST 151/150	12.5 destination NIC

We need a way to quickly fix a typo problem in a word. For our first proof of concept, we'll assume there is only one mistake in any word, and that mistake is caused by hitting a neighboring key on the keyboard. Each word always has the right count of letters and always has the correct beginning and ending letter. But a letter in the word may be swapped with a letter next to it on the keyboard. So, as long as we have a map of the keyboard, and a list of acceptable words, we have a good chance of correcting to the right word.



Given a sentence with keyboard-typo errors in words, rebuild the sentence by finding the right words.

## Input

Each data file will be in the following format: Line 1: sentence to correct. Lines 2-4: top/middle/bottom row of QWERTY-US keyboard layout (letters only) Line 5: list of keys which touch other keys in the form of: LETTER:list,of,keys,touching;NEXT-LETTER:list,of,keys,touching... Line 6-onward: list of (less than 10000) dictionary words to use for auto-correcting. Final line: ZZZZZ

```
we fond autovorrect is oftrn wfong
QWERTYUIOP
ASDFGHJKL
ZXCVBNM
Q:W,S,A;W:Q,E,A,S,D;E:W,S,D,F,R;R:E,D,F,G,T;T:R,F,G,H,Y;Y:T,G,H,J,U;U:Y,H,J,K,I;I:U,J,K,L,O
;O:I,K,L,P;P:O,L;A:Q,W,S,X,Z;S:W,Q,A,Z,X,C,D,E;D:E,W,S,X,C,V,F,R;F:R,E,D,C,V,B,G,T;G:T,R,F,
V,B,N,H,Y;H:Y,T,G,B,N,M,J,U;J:U,Y,H,N,M,K,I;K:I,U,J,M,L,O;L:O,I,K,P;Z:A,S,X;X:Z,S,D,C;C:X,S
,D,F,V;V:C,D,F,G,B;B:V,F,G,H,N;N:B,G,H,J,M;M:N,H,J,K,L
an
apple
autocorrect
branch
extra
fancy
find
great
is
it
often
right
we
wrong
ZZZZZ
```

## Output

Fix all misspellings using the touching keys list and the given words. All letters are lowercase. Print the fixed sentence.

```
we find autocorrect is often wrong
```

## Discussion

Misspelled words will only have one letter wrong. Reminder: have you run your solution against **all** of the student data sets? Student Data 2 & 3 have a VERY long list of words to consider.

John had a great puzzle to show us, but was a bit clumsy and dropped several words as he walked. We're pretty sure many are hiding in the letter grid on the floor, but could you help us determine if any are still missing?

## Input

The first line holds the dimensions of the grid as two integers. The first integer is the width (X), the second is the height (Y). The list of words follows, one per line, and terminates with the word "END". There are at most 20 words. After "END" is the grid to search.

```
9 9
BLUE
BLACK
GREEN
GREY
ORANGE
PURPLE
RED
PINK
VIOLET
YELLOW
WHITE
END
DQEBKOMZN
CPULPDIBZ
YELLPELVY
AGZUERBRE
ENUWBLUEL
FANYGLXPL
ORARBNARO
BOEAEQHCS
SYTXWIPZK
```

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

## Output

Check the given grid for the words given in the list. Print which words are found within the grid and which ones are missing. Grids can be as small as 5x5 and as large as 13x13. Grids can be square or rectangular. Words can be placed anywhere in the grid in a straight line (horizontal, vertical, and diagonal). Words may be found in any of the 8 directions. Only complete words should be marked FOUND. Words in the FOUND and MISSING lists should be listed (comma delimited) in ascending alphabetical order: A-Z

```
FOUND: BLACK, BLUE, GREY, ORANGE, PURPLE, RED
MISSING: GREEN, PINK, VIOLET, WHITE, YELLOW
```

## Discussion

The word "END" is not part of the list. See student datasets for additional examples.



As part of your job writing the code for the best new video stream service in town, CodeWars-TV, your manager has asked you to double-check that all of the films in your catalog has a trailer loaded and ready to play when the service is opened up to customers next week.

## Input

You will receive a standard text input file for this problem **plus** information about a directory to search (see "Guide To Directory I/O" on our Downloads page). The first line of the text input file will give you the name of the directory to search in your **files** directory of your student dataset. The rest of the lines of the text file will be

some titles of movies your streaming service is carrying. You need to check that the films listed in have trailers. The titles are not guaranteed to be in any specific order, but they **will** follow the pattern of: *Movie Title (year)* Movie titles may have any combination of: letters, numbers, spaces, and symbols. All valid ASCII characters (0-255 of Unicode spec.) are fair game. Input ends when END is found on a line by itself. Example:

```
CurrentFilmCatalog
Star Wars A New Hope (1977)
My Little Pony The Movie (2017)
Dune (1984)
Dune (2021)
END
```

The directory given will be structured following the pattern of: **DirectoryNameYouAreGiven**\MajorType\Genre, E.G.:

```
CurrentFilmCatalog\Animated\Action
CurrentFilmCatalog\Animated\KidsAndFamily
CurrentFilmCatalog\Live\SciFi
etc.
```

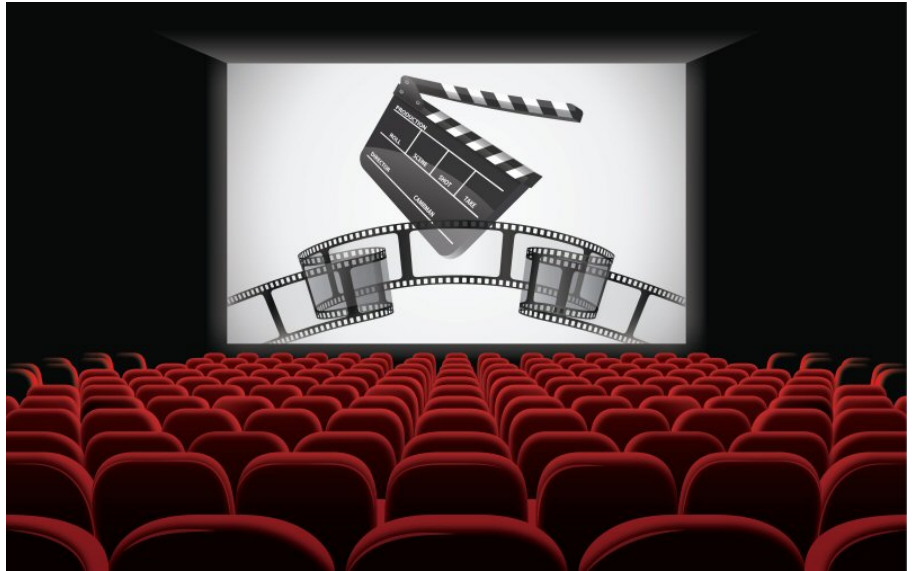
Inside each genre sub-directory will be film directories. One per film. E.G.:

```
CurrentFilmCatalog\Live\SciFi\Star Wars A New Hope (1977)
CurrentFilmCatalog\Animated\KidsAndFamily\My Little Pony The Movie (2017)
CurrentFilmCatalog\Animated\Comedy\Kung Fu Panda (2008)
etc.
```

It is **guaranteed** that the name of the folder holding a movie will match the names of the movies (including the years) given in your input file.

Trailers (if they exist) will be located in a sub-directory named "trailers" inside a film's directory. E.G. [1]:

```
CurrentFilmCatalog\Live\SciFi\Star Wars A New Hope (1977)\trailers\Episode IV -
trailer.m2ts
```



## Output

Your job is to check if each movie in the input list has a trailer file present in a trailers sub-directory for each film listed in your input file. To check that you will need to crawl through the sub-directories in the given directory and check if a "trailers" sub-folder exists for the film.

**Any** file found in a "trailers" directory in a film's directory with an extension of **".m2ts"** [2] will count as that film having a "trailer" available (do not evaluate the name of the trailer, that way lies madness).

Your output should simply list the films from the input file followed by a space and then either: "trailer missing" or "trailer found at: *Path\trailers\Episode IV - trailer.m2ts* (omit from the directory given in your input file down to root. E.G. if a file was found at "C:

\Users\yourUserName\Desktop\CodeWars2023\files\CurrentFilmCatalog\Live\SciFi\Dune (1984)\trailers\Dune (Original) Trailer.m2ts" [1], only output the path following the "CurrentFilmCatalog\"). Example:

```
Star Wars A New Hope (1977) trailer found at: Live\SciFi\Star Wars A New Hope
(1977)\trailers\Episode IV - trailer.m2ts
My Little Pony The Movie (2017) trailer missing
Dune (1984) trailer found at: Live\SciFi\Dune (1984)\trailers\Dune (Original) Trailer.m2ts
Dune (2021) trailer missing
```

## Discussion

Notes:

[1]

All of the path examples given are Windows paths. That is because the overwhelming majority of CodeWars participants (>97%) are using Windows. If you are on a Mac or a Linux computer, simply output the paths as they exist on your system. You do not need to worry about adjusting forward slashes to backslashes, for example.

**However** it would benefit you greatly to use your coding language's file and path libraries to program traversing and using paths in your solution, as those libraries (for example, in Python, `os.path`) will **automatically** adjust to the correct format for whatever computer the code is run on.

You also **must** use relative paths assuming your code will be run from the same directory as your datasets (same as you do for your input files). If you hard-code paths as strings, "E.G."

"\Users\KomikoStudentAccount\Workspace\Home\CodeWars\student\_datasets\files\CurrentFilmCatalog\" that will cause your code to fail when the judges run it (as we won't have the same directory structure as you).

[2]

You may have some familiarity with `.mov/.mp4` video files from your phones and computers. But most streaming services stream films using the `.M2TS` video container format, AKA MPEG-2TS which stands for **Moving Picture Experts Group Transport Stream** (also: BDAV MPEG-2 Transport Stream (closely related to Blu-Ray disc streams)).

M2TS is a derivative of the `.TS` video container format (MPEG-TS). It was created because MPEG-TS isn't great for streaming files over networks because it doesn't provide info about the duration of the film or provide seek-point time codes.

## Additional Examples

The output data for the following two examples wraps to multiple lines on the printed page. However in your output each film must be on one line, just like the student dataset output (have you tried them all?).

Input 1

```
NewFilmCatalog
Ratchet & Clank (2016)
The Boss Baby (2017)
Abominable (2019)
END
```

Output 1

```
Ratchet & Clank (2016) trailer was found at: Animated\Action\Ratchet & Clank
(2016)\trailers\Ratchet & Clank (2016) trailer.m2ts
The Boss Baby (2017) trailer was found at: Animated\Comedy\The Boss Baby
(2017)\trailers\trailer.m2ts
Abominable (2019) trailer was found at: Animated\KidsAndFamily\Abominable
(2019)\trailers\Abominable (2019) trailer.m2ts
```

-----

Input 2

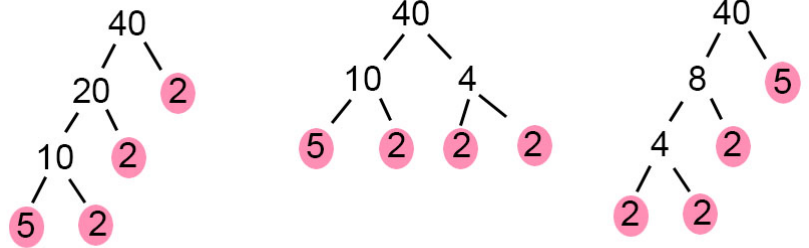
```
FallReleaseFilms
Hotel Transylvania (2012)
Serendipity (2001)
La La Land (2016)
Hello, Dolly! (1969)
END
```

Output 2

```
Hotel Transylvania (2012) trailer missing
Serendipity (2001) trailer was found at: Live\Drama\Serendipity
(2001)\trailers\trailer.m2ts
La La Land (2016) trailer was found at: Live\Musical\La La Land (2016)\trailers\La La Land
(2016) - trailer.m2ts
Hello, Dolly! (1969) trailer missing
```

How do trees get online? They just log in.

Now that you are online, let's look up what a factor tree is. A factor tree is a special diagram where we find the factors of a number, then the factors of those numbers, etc, until we can't factor any more. In other words, a factor tree shows the numbers that come together to create a larger number.



### Input

You will receive a list of integers (32-bit). Zero on a line by itself ends input.

```
40
9
18
7
25
13
12
100
0
```

### Output

If the integer can be factored into the multiplication of smaller integers (not including 1 and itself), list all of the possible factors as follows:

```
40=(20*2), (10*4), (10*2*2), (8*5), (5*4*2), (5*2*2*2)
9=(3*3)
18=(9*2), (6*3), (3*3*2)
25=(5*5)
12=(6*2), (4*3), (3*2*2)
100=(50*2), (25*4), (25*2*2), (20*5), (10*10), (10*5*2), (5*5*4), (5*5*2*2)
```

Any factors which can be broken down further should be listed as separate factor pairs. When listing factors, always list the largest factor **groups** first. Separate factor groups with a comma. Inside factor groups, always show the multiplication as: (largerNumber \* smallerNumber [\* smallerNumber..]). It will be counted as wrong to list the factors of 12 as: (2\*6),(3\*4),(2\*2\*3) List the factor pairs in descending order. It will be counted as wrong to list the factors of 18 as: (3\*3\*2),(9\*2).

### Additional Examples

Input 1	Output 1
4	4=(2*2)
8	8=(4*2), (2*2*2)
16	16=(8*2), (4*4), (4*2*2), (2*2*2*2)
32	32=(16*2), (8*4), (8*2*2), (4*4*2), (4*2*2*2), (2*2*2*2*2)
17	
23	
0	

Reminder: have you run your solution against **all** of the student data sets?

During World War II, messages were encrypted with an Enigma Machine. One young man became so entranced with reversing the Enigma codes, he named his daughter Amgine (the reverse of Enigma) and gave her a small model of an Enigma machine. Each dial had 10 numbers. He gave her cyphers for the machine throughout her life, on birthdays, holidays, and special days (like Pi Day, or Twosday). He also created larger dials with more than 10 numbers. He was kind enough to also provide the message key (the starting dial settings).



## Decryption and Rotation

Amgine's model Enigma machine works like this to find each decoded letter. Only capital letters are included.

- Convert the input letter to a number (A=1, Z=26). Sum that number and the three numbers on the dials.
- Divide the sum by 26 and take the remainder. Subtract the remainder from 26. Convert the result back to a letter.
- If the character is not a letter (space, punctuation, or number), it is copied without change.

After a letter (only a letter) is decrypted, the three dials rotate. Label the dials A, B, C, left-to-right.

- Dial A: If the number on Dial B is a multiple of 3, rotate forward one position. Otherwise do not rotate.
- Dial B: If the number on Dial C is a multiple of 5, rotate forward one position. Otherwise do not rotate.
- Dial C: Always rotate forward one position.

With dials of size 10, and (10,9,8) for a message key (the starting dial settings), "AMGINE" is encoded:

```
"A"=1. 1+10+9+8=28. The remainder is 2. Subtracting from 26 gives 24="X". (A,B,C) -> (1,9,9).
"M"=13. 13+1+9+9=32. The remainder is 6. Subtracting from 26 gives 20="T". (A,B,C) -> (2,9,10).
"G"=6. 6+2+9+10=28. The remainder is 2. Subtracting from 26 gives 24="X". (A,B,C) -> (3,10,1).
"I"=9. 9+3+10+1=23. The remainder is 3. Subtracting from 26 gives 23="W". (A,B,C) -> (3,10,2).
"N"=14. 14+3+10+2=29. The remainder is 3. Subtracting from 26 gives 23="W". (A,B,C) -> (3,10,3).
"E"=5. 5+3+10+3=21. The remainder is 5. Subtracting from 26 gives 21="I". (A,B,C) -> (3,10,4).
```

"AMGINE" becomes "XTXCWE". With the same message key (10,9,8), "XTXCWE" also becomes "AMGINE".

Amgine's father continued to send her cyphers to solve, but he didn't send the message key. Amgine knew that he always encoded her name somewhere in the message. With a little help from a computer, she found the message key and message by hunting for "AMGINE".

## Input

The first line is an integer (from 10 to 60), the maximum number on all three dials. The next lines are encrypted messages using those dials (up to 80 characters). The last line is a single period ('.') marking the end of input.

```
10
XTXCWE
VFK FVUE YW NQHSEFM, PCGCVC!
VINKEL GV UAPNAFYC.
LWY PMQN WAW FPBER VINKEL KSIU TYL RLQ 5 SSLRJT.
UHG QBGM ERWCFM KMQQ WMSLF, WZFHNZ?
.
```

## Output

Write a program to decode Amgine's cyphers. For each line, your program must print the message key (the first values of A B C separated by spaces) then the decoded message on a new line. Each message will include the decoded word "AMGINE" somewhere.

```
10 9 8
AMGINE
10 9 8
CAT FOOD IS AWESOME, AMGINE!
1 1 1
AMGINE IS SLEEPING.
6 8 6
THE MOST ICE CREAM AMGINE EVER ATE WAS 5 POUNDS.
10 1 1
SEE THAT AMGINE EATS LUNCH, PLEASE?
```



The CodeWars Community Park has begun a community garden. Each gardener has their own plot of land. They've agreed to a few guidelines to keep their section of the park looking nice.

- The full community garden is designed as an NxN square, divided into smaller squares like a checkerboard.
- There are N gardens, consisting of adjacent small squares.
- There are N trees. Exactly one tree is found in each row, column, and garden.
- No tree is immediately next to another tree. That is, there is an empty square (horizontally, vertically and diagonally) surrounding any tree's square.

The gardeners need your help to know where they should plant the trees. Based on the configuration of the gardens, there is only one tree-configuration that will work.

## Input

The first line is a single integer N (from 4 to 7), the size of the square community garden. The next N lines are a map, with each of the N gardens identified by a different lowercase letter, any letters from 'a' to 'k'. Each garden is fully connected by the sides of their grid squares. Gardens may have a size of 1 (a single square.)

Example 1	Example 2
4	6
aaab	aaaaaa
aabb	aaaabb
ccdd	cadddb
ccdd	cceffb
	ceffff
	ccffff

## Output

Determine where to place the trees. Print the same map, with a capital letter for each garden's tree.

aAab	aAaaaa
aabB	aaaabB
Ccdd	cadDdb
ccDd	Cceffb
	ceEfff
	ccffFf

## Discussion

Remember once a tree is placed, mark empty all surrounding squares and all squares in its garden, column and row. Here is a logical approach to solving the above grids. Your program may use any algorithm but must print the answer within 30 seconds.

There are only two ways 4 trees can be placed in a 4x4 map. They can't go in center or corner squares (try it), so they can only be in the side squares. Above, the "b" garden includes only one of these squares. The remaining trees are quickly placed.

In the 6x6 map, there are four squares which cannot hold trees: The top "c" square conflicts with the "d" garden. The right "d" square conflicts with the "b" garden. The left "e" square conflicts with the remaining "c" garden. The top "e" square conflicts with the remaining "d" garden. So, there is only one place for E, which leaves one place for D. B must be in the right column, leaving one place for F, then C, then B and A.

Based on the many "Parks" puzzles of Andrea Sabbatini.

Your company, *CodeWars Audio Inc.*, has been hired to publish the mp3 recordings made at various conventions, conferences, and other fan and professional events. The events will provide the list of events recorded, and a directory of the recordings. Now you need to sort that out so you can publish it!



## Input

You will receive a standard text input file for this problem **plus** information about a directory to search (see CodeWars 2023 download: "CodeWars\_GuideTo\_Directories.pdf"). The first line of the text input file will give you the name of the directory to search in your **files** directory of your student dataset. The rest of the lines of the text file will be some file names that need to be found in that directory. File names may have any combination of: letters, numbers, spaces, and symbols. All valid ASCII characters (0-255 of Unicode spec.) are fair game. Input ends when END is found on a line by itself. Example:

```
WritingConference
Sci-Fi Panel w Smith, Jones & Tanner -2021
Fantasy Workshop w Davies -2021
Cool tips on hot drama with our expert Hanna -2021
Young Adult First Page Critique w Satou and Lovecraft -2020
Crafting Your Perfect Pitch Master Class -2022
An Examination of Poetry -2020
All About Film Writing with Mark, Linda, Susan & Hinato -2022
END
```

In the directory will be files which mostly match up with the text file[1], but with some processing marks, shorthand and re-arrangement of words, and deletion of minor words. That is because the people who make the recordings are not the same people who create the schedule. Your algorithm[2] should ignore non-alphanumeric characters and it should iGnOrE cAsE (match against upper/lowercase versions of the same word). Examples of files you might find in the directory:

```
2020 Poetry Examination.mp3
2020 Tips.mp3
2020 Young Adult 1st Page Critique Satou Lovecraft.mp3
2021 Davies Fantasy Workshop.mp3
2021 Sci-Fi Panel - Smith Jones Tanner.mp3
2022 Crafting Your Pitch Master Class.mp3
2022 Film Writing Mark Linda Susan Hinato.mp3
```

## Output

List the matches (using an *arrow* "->") from the text file to the directory of files. You are guaranteed that there will be a match for each file listed in the text file, present in the directory (it may only match 10% of the original, but it will be there!). For these files, the output wraps to multiple lines on the printed page. However in your output each file must be on one line.

```
Sci-Fi Panel w Smith, Jones & Tanner -2021 -> 2021 Sci-Fi Panel - Smith Jones Tanner.mp3
Fantasy Workshop w Davies -2021 -> 2021 Davies Fantasy Workshop.mp3
Cool tips on hot drama with our expert Hanna -2021 -> 2020 Tips.mp3
Young Adult First Page Critique w Satou and Lovecraft -2020 -> 2020 Young Adult 1st Page
Critique Satou Lovecraft.mp3
Crafting Your Perfect Pitch Master Class -2022 -> 2022 Crafting Your Pitch Master Class.mp3
An Examination of Poetry -2020 -> 2020 Poetry Examination.mp3
All About Film Writing with Mark, Linda, Susan & Hinato -2022 -> 2022 Film Writing Mark
Linda Susan Hinato.mp3
```

## Discussion

[1]

It is entirely possible that there will be **more** files in the directory than you receive in your input text file. Don't worry about the files in the directory which do not have a match. Only worry about matching up the items listed in your text input file.

[2]

You will need to come up with a matching algorithm which can matchup the files to the text titles list. Your algorithm will need to take into account that:

- Words/Years *can* be missing
- Words can be rearranged
- Words can be misspelled

A successful algorithm will assign a matching percentage of likelihood of matching, eliminating impossible matches until a match can be made.

For example, in the text file there is a title for *An Examination of Poetry -2020* however in the files directory nothing is named that. The only possible match is *2020 Poetry Examination.mp3* because it contains the correct year (2020), and 2 of the exact keywords (*Poetry* and *Examination*) That means the file matches 3 of the possible 5 words from the text file, giving it a "matching score" of 60%. So long as no other file has a better matching score against the title, that should be the match.

Do not try to guess at misspellings or altered words, as you can never predict what your A/V people will shorthand next. Trying to account for *First* -> *1st*, *and* -> *&*, *with* -> *w* changes is a fool's errand. Focus on exact matches only.

The best example of matching by elimination and matching percentage is *Cool tips on hot drama with our expert Hanna -2021.mp3* The only file name that comes even close is *2020 Tips.mp3* It only matches 10% of the file name from the list, but it's the only file with the word "Tips" in it. It got the year wrong (that can easily happen when someone is typing fast!), and it has none of the other words, but it is the only possibility from the list when eliminating all of the 0% matches.

## Additional Examples

Reminder: have you run your solution against **all** of the student data sets?

Input 1

```
ComicCon2023
Spoken Word Poetry With William Shatner
Star Trek Lower Decks Cast Panel
How To Intern With Dreamworks Animation
Brooklyn-99 Movie Cast Reunion Panel
Photo Ops and TED Talk w/Jeri Ryan
Shuffle Dancing with Walking Dead Choreographer
Superman Comics Future Announcements
Marvel, Where Do We Go From Here?
END
```

Output 1

```
Spoken Word Poetry from William Shatner -> Shatner Alternative Music.mp3
Star Trek Lower Decks Cast Panel -> Lower Decks Panel.mp3
How To Intern With Dreamworks Animation -> Dreamworks Internships.mp3
Brooklyn-99 Movie Cast Reunion Panel -> Brooklyn Nine Nine Movie Announcement.mp3
Photo Ops and TED Talk w/Jeri Ryan -> Jeri Ryan Stuff.mp3
Shuffle Dancing with Walking Dead Choreographer -> Walking Dead Dancing.mp3
Superman Comics Future Hints -> Superman.mp3
Marvel, Where Do We Go From Here? -> Marvel Phase Six.mp3
```

-----

Input 2

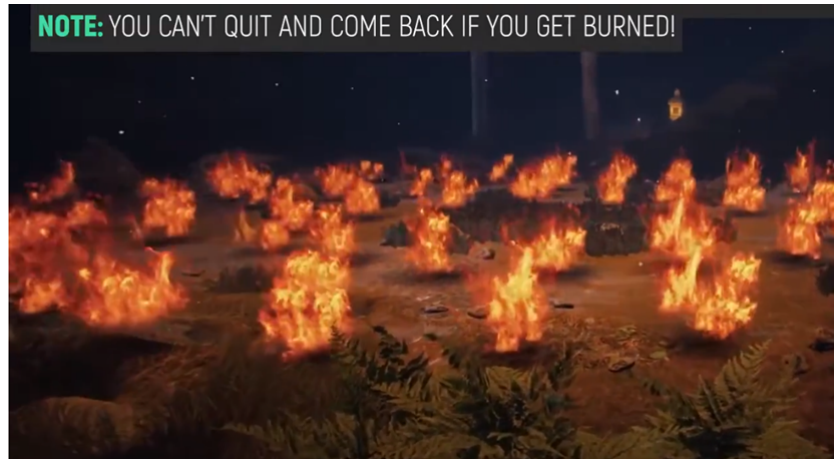
```
LaLakersMeetAndGreet2023
An Evening With Dennis Schroder
Patrick Beverley Real Talk
LeBron James About Records
Q&A With Rui Hachimura
Anthony Davis What's It All About
Can't Stop Lonnie Walker
Thomas Bryant Brings The Thunder
END
```

Output 2

```
An Evening With Dennis Schroder -> Evening With Dennis Schroder.mp3
Patrick Beverley Real Talk -> Real Talk with Patrick Beverley.mp3
LeBron James About Records -> King James.mp3
Q&A With Rui Hachimura -> Rui Hachimura Press Conference.mp3
Anthony Davis What's It All About -> Anthony Davis Interview.mp3
Can't Stop Lonnie Walker -> Lonnie Walker's New Book.mp3
Thomas Bryant Brings The Thunder -> Thomas Bryant Season Highlights.mp3
```

The CodeWars Offices have been built with safety as a top priority. If there is a fire in one room, the sprinkler valve is located in a different room so that you can reach it without going into the fire. All rooms have doors to neighboring rooms (not diagonally), but some rooms are full of extra equipment and cannot be entered. Unfortunately, if multiple fires start, one room's fire could block access to another's valve.

Given a (virtual) scenario with fires, find the shortest route to exit the CodeWars Offices.



## Input

The first line holds two integers M and N (separated by a space, each ranging from 5 to 10). The next M lines each hold N characters to mark each room in the office. 'S' is your starting position. 'X' is the exit. A hashtag '#' is a blocked room. A period '.' is an empty room. An uppercase letter (A-J) is a room on fire. A lowercase letter (a-j) is an empty room with a sprinkler valve for a room on fire (a for A, b for B, etc.)

Example 1	Example 2
5 6	3 5
. . S . . b	S . . g
# . # c ##	F # G # .
d B . # C .	X # . f .
A # . . . #	
X # D # a .	

## Output

Find the **shortest** path from 'S' to the exit 'X'. The exit and valves may be blocked by fires, so you will need to turn on a valve to put out a blocking fire. Once a valve (i.e. 'b') is visited, the fire is extinguished, and you can walk through the wet room (i.e. 'B') that previously was on fire.

Print 'S' for your start room. Then each time you enter any lettered room, print a space and its letter.

S b S B a B d A X                      S g f G S F X

## Discussion

- In Example 1, to reach X, you must put out fires A and B. To reach valve 'a', you must put out fire B, so 'b' is your first destination. There may be fires off of your path, like C and D. You do not need to extinguish them to escape. So valve 'c' is not on the shortest path. Valve 'd' just happens to be on the path anyway.
- All mazes are constructed so that the shortest path to any destination will ALWAYS be the right path to try. If there is a fire in your way along that path, you will need to first extinguish it by changing your destination to its valve. There may be another fire on that new path, so repeat until your path is fire-free.
- There are loops in some mazes. Re-analyze your solution when you extinguish a fire. There may be a shorter path. In Example 2, the path to X is blocked by fire 'F'. The shortest path from 'S' to 'f' goes through 'G'. So we first have to go to 'g'. From 'g', the shortest path goes directly to 'f', then exits by going through 'G'.