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. If you have any questions regarding these instructions, please ask it in our Discord server (your teacher has the link).

NEW instructions for 2021!

- **Online Code Submission**: Solutions 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 solution 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 solution for judging. For Java, the main class must still be named probXX.

- **C/C++ source code submission instead of executables.**: Any solutions in C or C++ must be submitted as source files and not as a binary executable. Your solutions can rely on glibc/libm and the standard headers. More information about our C/C++ environment and how you can create it locally are provided in the "C/C++ Development Environment Guide" available on the contest site.

Program Input

Most programs will require input. You have two options:

1 Read from file

Your program may read the input from a file. The input data will be in a local directory in the file **input.txt**.

2 Read from keyboard input (standard in)

Your program may read the input from the keyboard (standard in). You may type everything on the keyboard, or you may copy the data from **input.txt** to standard in.

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

Note: An easy way to enter keyboard data is by redirecting the contents of a file to your program. For example, if you are executing prob01, the input file **input.txt** can be redirected to the standard in of your program using syntax like this (examples are shown for each of the allowed languages):

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

Your program will behave exactly as if you were typing the input at the keyboard.

Program output

All programs must send their output to the screen (standard out, the default for any print statement). Please remember to carefully review your file names.

Source code naming convention

We recommend the following naming convention for your local source files: **probXX.java / probXX.js / probXX.py2 / probXX.py3 / probXX.c / probXX.cpp**, where 'XX' corresponds to the problem number (including leading zero, e.g. prob03.java). This naming convention allows you to use the `checkProb.bat` (Windows batch) file to test your solutions.

- For Python you should use the extension that matches the Python version that you are using.
- For Java, the main class must be named probXX. Note there is no capitalization. All main and supporting classes should be in the default (or anonymous) package.

You are strongly encouraged to submit solutions for Problem #0 and #1 (see next pages) prior to the start of the competition to ensure that your environment is compatible with the judges' and that you understand the input and output methods required.

problem 00

Welcome, to the 24th Triwizard Tournament!

1 points

This year Hogwarts will play host to a legendary event: the Triwizard Tournament. During which time a single student gets to represent his or her school in a series of magical contests.

Eternal glory awaits the student who wins the tournament. Please join me in welcoming our guests!

Input

You will not receive any input for this (and only this) problem

Output

Print a welcome message to the screen as follows:

```
We bid welcome to the ladies of Beauxbaton and our friends from the north, the sons of Durmstrang.
```

Discussion

Simply print to the screen the sentence given above as output. There are no datasets for this problem (as it is meant to ensure that your coding environment and setup works when submitting code for judging).

problem 01

The Needs Of The Many

1 points

Help Spock welcome you, the new captain of the Enterprise, to the bridge with a reminder of your duty -- by telling him your name.

Input

You will receive exactly one word on a single line.

```
Jim
```

Output

Print the following sentence to the screen, substituting the word read in from your dataset in place of the placeholder NAME: *NAME, the needs of the many outweigh the needs of the few, or the one; live long and prosper.*

```
Jim, the needs of the many outweigh the needs of the few, or the one; live long and prosper.
```

Discussion

You do not have to worry about spaces, hyphenated names, or words that are too long. You are guaranteed that the words supplied in the datasets will fit in the output, and conform to the guidelines given.

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

Additional Examples

Input 1	Output 1
`Jamie`	`Jamie, the needs of the many outweigh the needs of the few, or the one; live long and prosper.`

Input 2	Output 2
`Sisko`	`Sisko, the needs of the many outweigh the needs of the few, or the one; live long and prosper.`

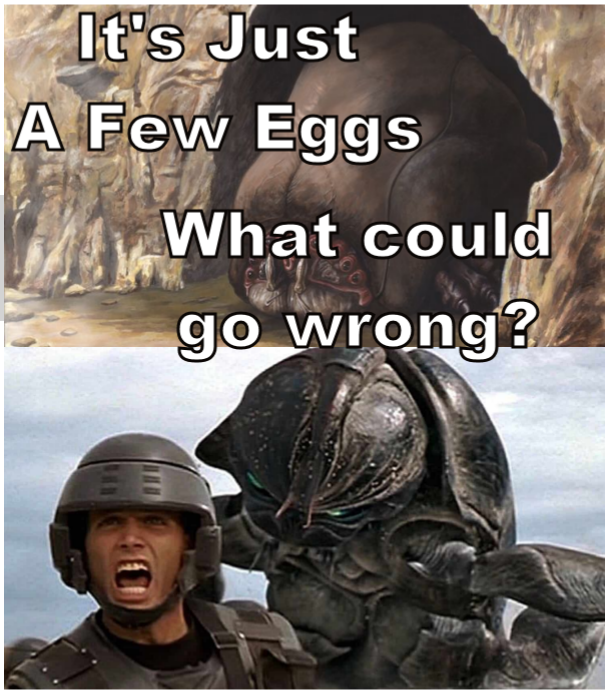
problem 02

Parasitoid / Host Interaction Models

2 points

W. R. Thompson (1924) was interested in what happened to parasitoid/host populations in areas where parasites were released into an area with a large host density. In the model initially he assumed that the parasitoid would only lay one egg in each host that was found. Thus:

```
(No. eggs laid) = (Mean Female Egg Compliment)(No.
Females Searching)
             Pe = C * P
```



However, this model did not work too well because many parasites are unable to distinguish between parasitized and unparasitized hosts. Thus Thompson's model predicted a higher rate of parasitism than would actually occur. In reality, a host can end up with more than one parasitoid egg and is then "superparasitized".
Thompson's model may not have been very accurate, but it makes a great CodeWars program. Write a program that uses Thompson's model to predict the rate of parasitism for a pair of input values.

Input

The first line of input is the value of C, the Mean Female Egg Compliment, an integer between 1 and 10,000. The second line is the value of P, the Number of Females Searching, an integer between 1 and 100,000.

```
1300
97450
```

Output

The program must print the value of Pe, the Number of Eggs Laid. The answer must match the judge's expected value precisely.

```
126685000
```

Discussion

The output should literally just be the product of the multiplication of the integer values from the two lines. You are guaranteed that you will not receive any values for input (or have to calculate for output) which will overflow an int-32.

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

Additional Examples

Input 1	Output 1	Input 2	Output 2
3579 6437	23038023	10000 100000	1000000000

problem 03

No Disassemble!

3 points

There's a short circuit in the management fabric induced by a power surge from a lightning strike. The engineering team needs you to take an unmarked white van and investigate the numbered input nodes in the fabric.

You must inspect each node by disassembling it. If it is working, you'll send a message back to the laboratory at corporate headquarters and reassemble the node. With the fabric not working correctly you'll be using a low-frequency channel that can only send very short messages. In fact, you can only send a single number.

The management at corporate headquarters would like to see a human-friendly message instead of a single number. So before leaving, you must write a program that reads a single number and writes a friendly message on the screen.

Input

The input is a single integer between one and ten, inclusive.

```
5
```

Output

The program must print a sentence in the format shown below, using the input number written as a word.

```
Number five is alive!
```

Discussion

Simply spell out the integer as a word and plug it into the output sentence.

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

Additional Examples

Input 1	Output 1	Input 2	Output 2
`10`	`Number ten is alive!`	`8`	`Number eight is alive!`

problem
04

Stonks

3
points

Pokémon card futures are blowin' up!

It's your time to shine and get rich in the stock market! To get rich quick simply look at how much a stock costs and multiply it by the total units you have managed to get enough money to buy!

Get your wallet ready and your 💎🙌!

Input

Each line of the input is two floating point decimal values. The last line of input is two zeros.

```
15.10 9.0
11.19 8.0
0 0
```

Output

Multiply the stock's cost (left most number) against how many units you want to buy (right most number). Output the total cost for each stock, one per line (matching the same order as the input). **Round your answer to 1 decimal place.**

```
135.9
89.5
```

Discussion

You are guaranteed not to have to deal with numbers greater than the value of an int-32, negative numbers, or numbers multiplied by zero.

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

Additional Examples

Input 1	Output 1	Input 2	Output 2
42.60 18.0	766.8	67.34 501.0	33737.3
99.30 52.0	5163.6	88.80 602.0	53457.6
87.32 15.0	1309.8	77.22 700.0	54285.7
0 0		55.90 900.0	50813.1
		1024.76 789.0	808535.6
		0 0	

problem 05

Makin' It Rain

3 points

Tony has a problem. He needs to program the prototype of his Iron Man suit, which means he needs the circuit logic to work, which means he needs to convince his obstinate compiler AI that: `A * (B + C) = A * B + A * C` Help Tony convince the compiler AI with some examples.

Input

You will get exactly 3 lines of input. Line 1 contains the value for A. Line 3 contains the value for C. (The rest of the line definitions Tony leaves as an exercise for the reader ^_-)

```
11
5
4
```

Output

Your program must help Tony convince his obnoxious know-it-all AI (is there any other type?) that his math is correct. So, your program must *show its work.* Print three lines of output. On the first line swap the numbers into the equation given above. On the second line, first solve the parts in parenthesis (on the left side of the equation); then do the multiplication (only on the right side). On the third line show the result of the final evaluations. AIs are tricky little devils, and they are extremely suspicious of *human errors* so follow the output format given below *exactly*.

```
11 * (5 + 4) = 11 * 5 + 11 * 4
11 * 9 = 55 + 44
99 = 99
```

Discussion

This is a substitution mathematics problem. Simply plug in the numbers into the formulas given for all of the stages given, and then calculate the results. You are guaranteed not to have deal with negative numbers or multiplication by zero.

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

Additional Examples

Input 1	Output 1	Input 2	Output 2
15 7 89	15 * (7 + 89) = 15 * 7 + 15 * 89 15 * 96 = 105 + 1335 1440 = 1440	22 105 33	22 * (105 + 33) = 22 * 105 + 22 * 33 22 * 138 = 2310 + 726 3036 = 3036

problem 06

Sir Isaac Newton is the deadliest person in space

4 points

Shepard knows that Newton stood on the shoulders of pioneers such as Johannes Kepler, who derived three mathematical laws for planetary motion in the early 1600's. This work was based on measurements made before the advent of the telescope using instruments called quadrants and sextants. Oh, and before the advent of the computer as well.

Kepler's third law of Planetary Motion captures the relationship between the distance of planets from the Sun and their orbital periods. It states that the square of the orbital period of a planet is proportional to the cube of the semi-major axis of its orbit. If we assume that the period P is measured in Earth-years and the orbital radius R is measured in Astronomical Units (AU), then the equation is simply this:

$P^2 = R^3$

Where P is the period measured in Earth-years, and R is the semi-major axis (orbital radius) in Astronomical Units (AU).

Help Shepard write a program to calculate the orbital radius of a planet given its orbital period so she can better map the star system she is in. To solve this problem you may need to know that you can express square roots and cube roots as fractional exponents. For example:

$x^{1/2} = \sqrt[2]{x}$

$x^{1/3} = \sqrt[3]{x}$

Input

Each line of input is the orbital period (P), in years, of a body orbiting the sun. The input ends with a value of zero.

```
1.8808
4.60
0.615198
30.07
0
```

Output

For each period, the program must print the semi-major axis (R) in AU. Round your answers to 4 decimal places. The answer must be accurate to within +/- 0.01 AU.

```
1.5237
2.7659
0.7233
9.6699
```

Discussion

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

Additional Examples

Input 1	Output 1	Input 2	Output 2
347.683	49.4450	123.456789	24.7938
0.640846	0.7433	0.98607534	0.9907
175.3	31.3224	222.82	36.7538
8.549	4.1810	9.555	4.5028
64.017	16.0028	387.321	53.1350
0		0	

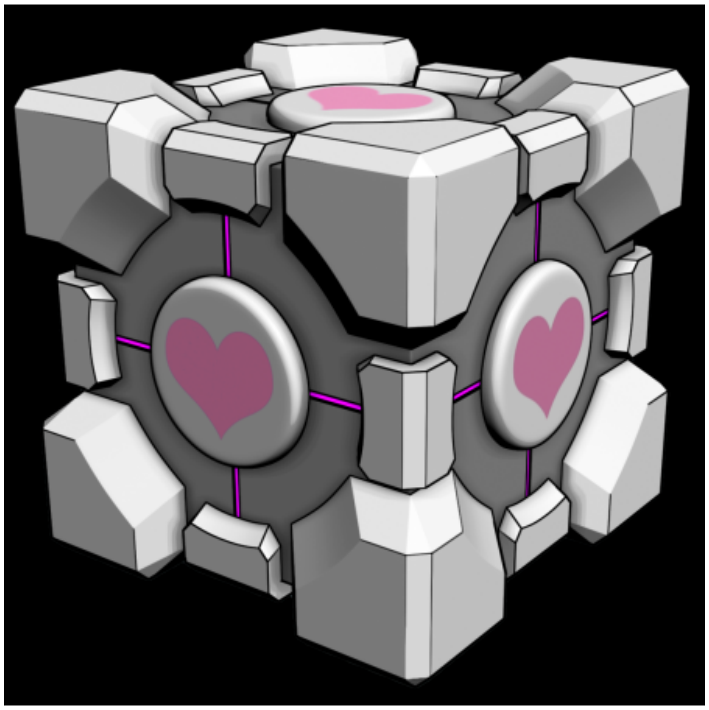
problem 07

Companion Cubes

5 points

GladOS is sorting her testing subjects (again). While she's at it, she's also going through the historical records of *ex-~~employees~~ test subjects* to make sure their assigned companion cubes were properly collected when their tests were *concluded.*

GladOS is especially concerned to find lonely companion cubes (cubes assigned to test subject **Lonely** which is the code for **not assigned** in her system), duplicate assignments for companion cubes (the same cube ID number assigned to more than one person), and test subjects who are missing their cubes (assigned cube ID 0).



Input

GladOS is calling up lists of *test subjects* from the archives. Each list will have, on line one, the total count of people in the list. Every line after that will consist of the subject's first name followed by a space, and then the companion cube ID assigned to that person.

```
11
Chell 42
ATLAS 0
P-body 0
Wheatley 99
Lonely 52
Cave 123
Caroline 123
Rattmann 1
SpaceCore 0
Lonely 12
Rick 9
```

Output

Print 3 total lines. On the first line print the count of Lonely cubes. On the second print the count of duplicate companion cube ID assignments. On the third line print the count of test subjects without a companion cube.

```
Lonely Cubes: 2
Duplicate Cube Assignments: 1
Test Subjects without Cubes: 3
```

Discussion

No names will contain spaces. For the duplicate counts, you are counting the duplicate **cube IDs** assigned, not the total number of times IDs were duplicated. For example, if cube ID 99 shows on every line of the file, that counts as **one** cube ID that was duplicated.

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

Additional Examples

Input 1	Output 1

```
12
Gordon 0
Lonely 971
Ozioma 1
James 3
Lonely 125
Lonely 126
Shapiro 3
Tony 4
Ellen 5
Lonely 124
Rhys 1
Lonely 123
```
	```
	Lonely Cubes: 5
	Duplicate Cube Assignments: 2
	Test Subjects without Cubes: 1
	```