Problem 3 - Enigmanation

Edward Nigma (the Riddler) is one of the most formidable adversaries of the almighty Dark Knight. That is because of the superior intelligence of the Riddler. He has just put Batman into the enigmanation, a state where the Batman's imagination is controlled by The Riddler.

The only way for the Dark Knight to escape the enigmanation is to solve the Riddler's riddle. This time the riddle is to find the exit code. The exit code is the result from a complex mathematical expression.

The code is actually a mathematical expression. Each mathematical expression can contain **SUM**, **SUBTRACT**, **MODULE** and/or **MULTIPLY** operations. The expression is solved **like a calculator would do** (example follows) and **can contain brackets**. **Everything inside an opening and closing bracket is solved first**. An expression **can contain multiple brackets**, but **cannot contain nested brackets (**i.e. expression containing **(… (…)…) is invalid**, but expression containing **(…)… (…) is valid**).

Example:

Given the expression:

(1 + 9) % 6 – (7 % 2) * 8

It is solved as follows:

(1 + 9) % 6 – (7 % 2) * 8 =
10 % 6 – (7 % 2) * 8 =
4 – (7 % 2) * 8 =
4 – 1 * 8 =
3 * 8 =
24 =
24.000

Batman is pretty smart, but he is busy fighting with the monsters in the enigmanation. He needs your help to escape from the enigmanation and capture the Riddler.

Input

The input data consists of single line on the console

The line represents **the code to solve**. The line will **always end with the symbol '='**. The '=' **at the end marks the end** of the code.

The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

The output data should be printed at the console on a single line – the result from the calculated code.

The result should be rounded to the 3 digit after the decimal point

Constraints

- The code will contain **no more than 2500 characters**
- The numbers in an code will always be in the range **[1…9]**
- The operators in an code will always be among **+ (**SUM**)**, **– (**SUBTRACT**)**, **% (**MODULE**)** or *** (**MULTIPLY**)**
- The result will always **be in the range [-100000.000, 100000.000]**
- Allowed work time for your program: 0.1 seconds.
- Allowed memory: 4 MB.

Examples

Input	`(1+9)%6-(7%2)*8=`
Output	`24.000`
Input	`(6%5)*9+(8*3%7)*7%2*(5%3+4*1)=`
Output	`0.000`
Input	`1*2*3*4*5*6*7*8*9%1-9=`
Output	`-9.000`