## Problem 3 - X – Expression

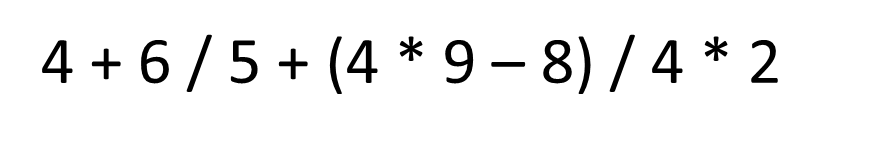
Bonnie is a very powerful witch. Since the power of the nature is not enough to fight against all the vampires and werewolves, she started to master the power of the Expression.

Expression is a very difficult to master, since its spells depend of the **ability to quickly solve complex mathematical expressions**. To cast an expression spell, the witch-caster should know the result in advance.

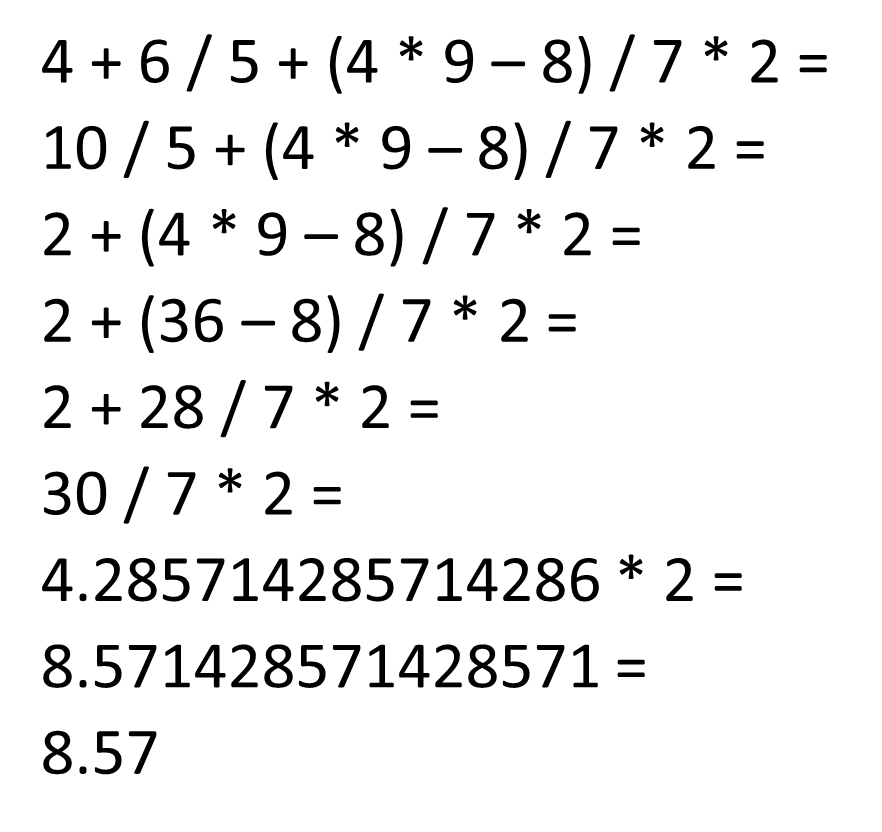
An expression spell consists of set of simple mathematical expressions. Each mathematical expression can contain **SUM**, **SUBTRACT**, **DIVIDE** and/or **MULTIPLY** operations. The expression is solved **like a calculator would do** (example follows) and **can contain brackets**. **Everything inside a 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:



It is solved as follows:



Since Bonnie is quite pretty, but not smart at all, she needs your help to master the art of Expression.

### Input

The input data consists of single line on the console

The line represents **the expression to solve**. The line will **always end with the symbol ‘=’**. The ‘=’ at the end **marks the end of the expression**.

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 expression.

**The result should be rounded to the 2 digit after the decimal point**

### Constraints

* The expression will contain **no more than 2500 characters**
* The numbers in an expression will always be in the range **[1…9]**
* The operators in an expression will always be among **+ (**SUM**)**, **– (**SUBTRACT**)**, **/ (**DIVIDE**)** or **\* (**MULTIPLY**)**
* **The result will always be in the range [-100000.00, 100000.00]**
* Allowed work time for your program: **0.1** seconds.
* Allowed memory: **4 MB**.

### Examples

|  |  |
| --- | --- |
| **Input** | 4+6/5+(4\*9–8)/7\*2= |
| **Output** | 8.57 |
| **Input** | 3+(6/5)+(2\*3/7)\*7/2\*(9/4+4\*1)= |
| **Output** | 110.63 |