Etnetera Brevity Challenge

`brevity.c`, `brevity.cpp`, `Brevity.java`, `brevity.py`

It became a tradition for the practice round of this competition to include a playful problem brought to you by Etnetera. The last year or so, everyone seems to be talking about *Artifical Intelligence* (AI), so we decided to use this topical subject. By the way, it is the main theme not only for this problem, but also for our internal challenge for employees of Etnetera Group.

Today's AI is capable of producing surprising results when generating text, images, music, or even computer programs. Many people use AI-assisted development tools. However, the output of AI is generally unreliable and often incorrect (speaking politely). With AI-generated programs, having very good test cases gains even more importance than with human developers. If the test cases have high coverage, we can validate programs generated by AI and have reasonable confidence in the results. And who knows – maybe one day, programmers will *only* need to write tests, and AI will then produce corresponding programs.

You know what? Let's try that approach, right away! You will be given a bunch of test cases and your task is to automatically produce a program that satisfies all of them.

The programming language used in this problem is a simple arithmetic language. Each program has the form of an expression, which may use the following values and operators:

- Integer values can be specified using decimal notation.
- "`a`" and "`b`" denote two input variables available to the program.
- "`(`" and "`)`" are parentheses that alter priority of other operators in the usual way.
- "`*`" means multiplication.
- "`+`" and "`-`" means addition and subtraction.
- "`>`", "`<`", and "`=`" are comparison operators. Their result is 1 if the left operand is greater than, less than, or equal to the right operand, respectively. Otherwise the result is 0.
- "`&`" and "`|`" are meant as operators for logical conjunction and disjunction. "`&`" returns zero if the value of the left operand is zero, otherwise it returns the value of the right operand. "`|`" returns the value of the left operand, unless it is zero, in which case it returns the value of the right operand.
- "`?`" and "`:`" together form the standard ternary conditional operator. `x?y:z` will return `z` if the value of `x` is zero, and `y` otherwise.

Operators above are listed in the decreasing order of their priority. Operators in the same item have equal priority and are associative from left to right (`x-y+z` means `(x-y)+z`), except for the ternary operator, which is associative from right to left (`u?v:x?y:z` can be interpreted as `u?v:(x?y:z)`). Note that there are no two-character operators (such as `<=`) and no division.

Input Specification

The input contains at least 1 and at most 20 test cases. Each test case is specified on one line by three integers: A, B, and R. These numbers mean that if we set input variables `a` and `b` to values A and B, respectively, the program must then yield the output value R.

No integers in the input will be negative or higher than 1000. No two test cases will contain the same combination of values for both input variables.

Output Specification

Print one line with any valid program that satisfies all the given test cases. The program may contain spaces between values and operators. The line length cannot exceed 10 000 characters.

All values processed by the program must be integers with their absolute value at most 10^9. If a higher value should be produced at any point, the results are undefined.

Sample Input 1

```
1 2 4
1 3 5
1 4 5
4 0 1
```

Output for Sample Input 1

```
a=1 & b=2 ? 4 : a=1 ? 5 : 1
```

Sample Input 2

```
1 2 3
0 7 0
4 5 24
30 30 930
8 9 80
```

Output for Sample Input 2

```
a*(b+1)
```

---

etnetera
core

While any valid solution will be accepted for the purposes of the Practice Session and its ranking, our **Etnetera Brevity Challenge** always seeks for the *shortest* solution, which in this case means the minimal number of characters forming the resulting program (the output line). The winning team will be awarded a special prize from Etnetera: a paid subscription plan for a real AI-assistant, GitHub Copilot, so you may try this technique for real.

Note that there are multiple test scenarios and you should optimize as much of them as possible. You may submit your solutions repeatedly — the best result counts. Unfortunately, the current evaluation system will not tell you how long your programs really were, so you need to do your best even without having such feedback.

Please remember that it is forbidden to “cheat” in any way, in particular, to abuse the system by sending too many submissions in order to find out details about the real test data.

Good Luck with the **Etnetera Brevity Challenge!**