Adding Words

Psychologists at Wassamatta University believe that humans are able to more easily deal with words than with numbers. So they have devised experiments to find out if this is true. In an interesting twist, one of their experiments deals with using words which represent numbers. Rather than adding numbers, they want to add words. You are a research programmer on the project, and your job is to write a program that demonstrates this ability.

Input

Input is a sequence of up to 2000 commands, one per line, ending at end of file. Each command is a definition, a calculation, or a clear. All tokens within a command are separated by single spaces.

A definition has the format `def x y` where x is a variable name and y is a an integer in the range $[-1\,000, 1\,000]$. No two variables are ever defined to have the same numeric value at the same time. If x had been defined previously, defining it again erases its old definition. Variable names are up to 30 lowercase characters, each character from the range a to z.

A calculation command starts with the word `calc`, and is followed by one to 15 more variable names separated by addition or subtraction operators. The end of a `calc` command is an equals sign. For example:

$$\text{calc foo + bar - car =}$$

The `clear` command instructs your program to forget all of its definitions.

Output

Your program should produce no output for definitions, but for calculations it should produce the value of the calculation. If there is not a word for the result, or some word in the calculation has not been defined, then the result of the calculation should be `unknown`. The word `unknown` is never used as a variable in the input.

Sample Input 1

```
def foo 3
calc foo + bar =
def bar 7
def programming 10
calc foo + bar =
def is 4
def fun 8
calc programming - is + fun =
def fun 1
calc programming - is + fun =
clear
```

Sample Output 1

```
foo + bar = unknown
foo + bar = programming
programming - is + fun = unknown
programming - is + fun = bar
```