CyberChallenge.IT 2024
Programming Test

Emulation [60 points]

Problem Statement

Bob Prof. Alan, I think I got infected by malware...

Alan Fear not, dear Bob, it's not malware. It's just a CTF challenge printing amusing things on your terminal. But hey, we can leverage it to dive into some reverse engineering!

Charlie Reverse engineering? What's that all about?

Alan In simple terms, it's dissecting a piece of software to understand its inner workings. In Bob's case, we've got a virtual machine with straightforward custom instructions. One of the initial steps while facing these challenges is usually crafting an emulator. Think you're up for the challenge?

Bob A... What?

Alan Just some code to run arbitrary programs in the given language! I've set up a small table with the available instructions for you. Your mission: give me the result of a given piece of code. Do you think you can print the sum of the variables `a, b, c, d, e, f` at the end of the execution?

Bob starts screaming and runs away...

VM Details

The VM has 6 variables, namely `a, b, c, d, e` and `f` that store integer values. Operations are sums, subtractions, multiplications and conditional jumps to labels. Labels are always identified by 2-characters long name, in lowercase ascii. The syntax is given in the following table.

At the beginning all the 6 variables are set to 0. Operations are always performed between integer values.

Instruction	Syntax	Explanation
Addition	`add <var> <num>`	Adds the value of `num` to the variable `var`. Example: `add a 5` adds 5 to the variable `a`.
Subtraction	`sub <var> <num>`	Subtracts the value of `num` from the variable `var`. Example: `sub a 5` subtracts 5 from the variable `a`.
Multiplication	`mul <var> <num>`	Substitute the value in `var` with its product with `num`. Example: `mul a 5` stores in `a` the value 5`a`.
Labeling	`lab <name>`	Sets a label named `name`. Label names are always 2-characters long. Labels do not affect variables, but can be jumped to with the `jmp` instruction.
Jumping	`jmp <var> <num> <name>`	Jumps at the label `name` if the value stored in the variable `var` is equal to `num`.

Problem Details

Input

The input consists of $N + 1$ lines:

- Line 1: an integer N, the number of lines the program contains
- Lines $2, \ldots, N + 1$: the instructions of the program, one per line

Output

The output is a single integer, representing the sum `a+b+c+d+e+f`.

Scoring

Your program will be tested on a number of testcases grouped in subtasks. In order to obtain the score associated to a subtask, you need to correctly solve all its testcases.

For both subtasks, N will not exceed 500. It is also guaranteed that no infinite loops will be given in the testcases (i.e., all the programs terminate). It is also guaranteed that in any point of the computations required, the variables will not exceed 2^{60} or go below -2^{60}.

- **Subtask 1** [**30 points**]: the program only contains `add, sub, mul` instructions.
- **Subtask 2** [**30 points**]: no restrictions on the used instructions.

Examples

INPUT	OUTPUT
10 lab xm add e 33 lab ik add d 79 mul a 48 jmp d 79 xm add c 43 lab ex lab ne jmp c 43 ik	389

Explanation

Let's go through the code step by step:

- The first line creates the label `xm`
- The second line sets `e` to 33
- The third creates the label `ik`
- Fourth and fifth lines set `d` to 79 and `a` to 0
- The sixth line jumps to he label `xm`, since `d` is 79
- Then `e` goes to 66, `d` to 158 and the jump condition is not true anymore, so we go to line 7
- Line 7 sets `c` to 43
- Lines 8 and 9 create two labels, `ex` and `ne`
- Since `c` is 43 we execute the jump at line 10 to label `ik`
- At this point, `d` goes to 237, `a` stays at 0, the jump to `xm` is not executed and in the end `c` goes to 86
- The jump to `ik` is not executed since `c` is `86`
- The final result is $0 + 0 + 86 + 237 + 66 = 389$