Shipping Robot

The Idea

You are working for a company (Amazing Shipping) that is automating their shipping structure with robots. The robot you are coding is given stacks of small, medium, and large boxes. This robot needs to sort the boxes by size.

Your Goal

Your goal is to simulate a robot organizing the stacks of boxes. You will be given each stack as a string of boxes containing ‘S’, ‘M’, and ‘L’ boxes. You need to show the steps your robot takes to move the boxes into stacks containing only one box size each. Your robot can only move one box at a time. Also note that the stacks behave like the stack data structure, i.e. last in, first out (LIFO).

Input

Your program should expect the input to be represented as if it were from a user at the keyboard. You can safely assume no malformed input will be provided.

- You will receive 3 strings containing a variable number of ‘S’, ‘M’, and ‘L’ characters, representing a stack of boxes. The “rightmost” character represents the top of the stack.

Output

The output should be a string consisting of a list of numbers, grouped in pairs of two. The first number in each pair indicates the stack to remove the item from, and the second number indicates the stack to add the item to. The two items in the pair should be comma separated, and each pair should be followed by a semicolon. There are only 3 stacks, and they are identified by the numbers 1, 2, and 3.

For example, “1,2;2,3;” contains two pairs of instructions. The first instruction tells the robot to take the item from the top of stack 1 and place it on top of stack 2. The second instruction tells it to remove the item from the top of stack 2 and place it on top of stack 3.

Example Input/Output

Input:

SM

ML

LS

Output:

2,3;1,2;3,2;3,1;2,3;

Explanation:

Original stacks: SM ML LS

2,3; SM M LSL

1,2; S MM LSL

3,2; S MML LS

3,1; SS MML L

2,3; SS MM LL

Evaluation

To evaluate your program, the judges will first compile your program (if your language requires compiling, that is). For the sake of example, let's assume you submit a source file called prog.c. The judges will perform the most basic compile unless you inform them more is required.

$ gcc -o prog prog.c

They will then execute the program using the following command (or similar, depending on the language and test file used):

$ cat 0.dat | ./prog | python3 ./judge.py 0.dat

This will provide the test data to your program as if it were being entered by someone at the console. It will also capture the output of your program and provide it to the judging

program as if it were being entered by someone at the terminal. For your convenience, the judging program's source and data files used for judging are provided.

Runtime

While there is no official time limit for how long your program runs, it will be at the discretion of the judges to terminate your program whenever they see fit. Generally, we will let it run for a reasonably long time unless the testing machine gets too bogged down with concurrently running submissions.