

## Old Typewriter (typewriter)

Edoardo found a very old typewriter that is undoubtedly fashionable, but not particularly efficient. In particular, when you type a string, the following errors may (randomly) happen:

- **Stuck key:** when you type the same character two times consecutively (e.g. “aa”), only one of them is written (e.g. obtaining “a”).
- **Wild key:** when you type a character (e.g. “a”), it may be repeated twice (e.g. obtaining “aa”).

Furthermore, these errors may happen multiple times for the same string, and even multiple times for the same characters: for instance, a doubly stuck key would change “aaa” into “a” and a doubly wild key would do the opposite change.



Before the first round, Edoardo wrote the password of the contest server three times with the typewriter, obtaining strings  $S_1, S_2, S_3$ . Edoardo does not remember the password any more and thus he needs to reconstruct it from these strings. Since he trusts his old typewriter, he decided that the most likely password  $C$  is the string such that  $S_1, S_2, S_3$  may be generated from it *with the least number of errors*.

Help Edoardo find the password  $C$  minimising the number of errors. If multiple passwords exist for the same minimal number of errors, you can choose any one of them. If no password exist which may generate  $S_1, S_2, S_3$ , you should notify Edoardo that he must have made mistakes when typing the strings.

Among the attachments of this task you may find a template file `typewriter.*` with a sample incomplete implementation.

### Input

There are three lines in input, respectively  $S_1, S_2, S_3$ .

### Output





You need to write a single line with the new common string  $C$  that minimizes the total number of operations required to convert all three initial strings to  $C$ . If such a string cannot be found, the program should print “IMPOSSIBLE” without quotes and in capital letters.

## Constraints

- Each string is formed by lowercase English letters.
- The length of each string does not exceed 100 000 characters.

## Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

- **Subtask 1** (0 points)      Examples.  

- **Subtask 2** (30 points)      The length of each string does not exceed 50.  

- **Subtask 3** (50 points)      The length of each string does not exceed 1000.  

- **Subtask 4** (20 points)      No additional limitations.  


## Examples

input	output
aaaza aazzaa azzza	aazza
xy xxyy yx	IMPOSSIBLE

## Explanation

In the **first sample case**, the first string can be obtained from  $C = \text{“aazza”}$  through 2 errors. The second can be obtained through 1 error, and the last through 2 errors. The total is 5 errors, and it’s not possible to achieve a lower number of errors with a different  $C$ .

In the **second sample case**, there isn’t a string  $C$  such that all three string can be obtained from it through errors.