

## Problem B. Build the String

Input file:            standard input  
Output file:           standard output  
Time limit:            2 seconds  
Memory limit:         64 megabytes

Let us have some virtual machine. The machine has a memory stack that can infinitely widen and can contain strings of any finite non-zero length.

The machine supports four operations:

- `copy` — copy a string at the stack’s end and place it in the stack’s end;
- `swap` — swap the last string in the stack and the one before the last;
- `roll` — cyclically shift three last strings in the stack away from the stack’s end;
- `fuse` — extract two strings from the stack’s end and then place their concatenation at the stack’s end.

More formally it looks like that: (`[...]` stands for some sequence of strings at the beginning of the stack, perhaps of zero length):

```
copy: [...] x → [...] x x;  
swap: [...] x y → [...] y x;  
roll: [...] x y z → [...] y z x;  
fuse: [...] x y → [...] xy.
```

Program for a given virtual machine is represented by a command sequence; the machine performs the commands one after another. If the stack doesn’t have enough strings to perform the program’s current command, then an event **CRASH** occurs and the machine stops functioning. The machine also stops if the program runs out of commands (in this case the **CRASH** event never occurs).

Initially the virtual machine’s stack contains two strings and has the form of “a b”. You have to write a program for the given machine; the program’s progress should result in an  $s$  string located at the stack’s end (at the end of the program’s progress the stack can have more than one string left). The program should contain no more than  $3 \times |s|$  commands ( $|s|$  — is the  $s$  string’s length). Of course, the program’s progress shouldn’t lead to the **CRASH** event.

### Input

The first line contains the  $s$  string. It only consists of lowercase Latin letters “a” and “b” and has the length from 1 to  $10^5$  characters.

### Output

Print on the first output line number  $k$  the number of commands in the program ( $0 \leq k \leq 3 \times |s|$ ). Print on the next  $k$  lines  $k$  commands, one command per line. The acceptable commands are “`copy`”, “`swap`”, “`roll`” and “`fuse`”. As the result of the program’s progress the last element in the stack should be string  $s$  (the stack can have more than one string left). The program shouldn’t lead to the **CRASH** event. If there are several acceptable decisions, print any of them. See the samples for clarifications.

## Example

standard input	standard output
ababa	9 swap copy roll fuse copy fuse copy roll fuse

## Note

During the sample program's progress the virtual machine's stack changes in the following manner:

a b → b a → b a a → a a b → a ab → a ab ab → a abab → a abab abab → abab abab a → abab ababa