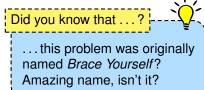
# Problem H: Helpless Humans

When writing text, people sometimes rely on braces to inject ad- Did you know that . ditional details to certain statements. Due to human defectiveness, braces (especially closing ones are prone to be missing. A different race – the compilers – heavily relies on braces to be balanced and well nested. To prevent misapprehensions and ensure proper



interaction between humans and compilers, all humans' texts need to be verified prior to passing them on to the compilers.

The pairs of characters (),  $\{\}$ , [], and  $\{\}$  are used as control symbols in source code and each define a control block. An opening brace (i.e., one of ({ [<) marks the start of a control block, the corresponding closing brace (i.e., the matching brace ) } ] >) marks the end of the control blocks. Control blocks can encapsulate other control blocks, but no two control blocks may intersect each other. A text may also contain string literals, which are character sequences delimited by a pair of quotation marks ". Quotation marks cancel out control characters, which means that after the first quotation mark no character is interpreted as control character until the next quotation mark. Write a source code check that tests if all string literals and control blocks are valid and closed.

## Input

The input consists of:

• One or more lines (separated by \n) made up of whitespace, numbers, alphabetic characters, and all other printable ASCII characters. More precisely, the set of allowed characters is the ASCII range 0x20-0x7e, as well as 0x09 (tab) and 0x0a (newline).

The total number of characters in the input is at most  $1\,048\,576$  (= 1 MiB).

# **Output**

Output correct if all control blocks and strings are valid and closed or incorrect if not.

#### Sample Input 1

### **Sample Output 1**

```
{ [ < "this is a > string" > ]
                                correct
this is not a string }
```

#### Sample Input 2

#### Sample Output 2

```
{ [ < "this is a > string " ]
                                 incorrect
this is not a string } ()
```

