COCI '15 Contest 7 #1 Nizovi

Time limit: 1.0s **Memory limit:** 64M

An **array** in a certain programming language is defined in the following way:

- The beginning and the end of an array is denoted by an open and closed curly bracket, respectively.
- Inside the curly braces, there are (possibly even 0) **values** separated by a comma (there is no comma after the last value in the array).
- Each value can be a **word** (an array of lowercase letters of the English alphabet) or another array.
- Examples of correctly defined arrays are: `{}`, `{a,b,c}`, `{abc,znj,{novi,niz},pozz}`.

Recently, you've noticed that the programming language does not throw an error if you do not place the appropriate number of spaces and new lines before or after the curly braces or the commas. Given the fact that you too mind the values being so "squished" together, you've decided to get to work and modify the shape of the array in the following way:

- Each value that is not an array or denotes the beginning or the end of the array (curly braces) will be in its own **line**.
- The commas are "connected" with the value located directly before them and there is a new line after each comma.
- After the curly bracket denoting the beginning of the array, the indentation of the content **increases** (shifting the output to the right) for **2 spaces**.
- Before the curly bracket denoting the end of the array `}`, the indentation of the content **decreases** (shifting the output to the left) for **2 spaces**.

Input Specification

The first line of input contains an array of characters S $(1 \leq |S| \leq 1500)$, representing the array from the task.

Output Specification

The output must consist of the modified version of the array from the task.

Scoring

In test cases worth 50% of total points, all values in the array are going to be words (in other words, it won't be the case that a value is another array).

Sample Input 1

```
{abc,ono,sto}
```

Sample Output 1

```
{
  abc,
  ono,
  sto
}
```

Sample Input 2

```
{}
```

Sample Output 2

```
{
}
```

Sample Input 3

```
{znakovi}
```

Sample Output 3

```
{
  znakovi
}
```

Sample Input 4

```
{a,b,{c,d},e,{}}
```

Sample Output 4

```
{
  a,
  b,
  {
    c,
    d
  },
  e,
  {
  }
}
```

Explanation for Sample Output 4

In the beginning, there is no indentation, it is 0. After the first curly bracket, there is a new line and the indentation increases by 2 spaces. After that, the word `a` is printed, as well as the comma right after it, then the new line with the same indentation of 2 spaces. The same procedure is repeated for the word `b`.

The third value in the array is a new array (let's call it array X). It begins with a curly bracket, so we need to print a new line and increase the indentation for 2 spaces. The indentation is now a total of 4 spaces. Using that indentation, we print the words `c` and `d` each in its own line. After the word `d`, there is no comma because it is the last value in the array X.

Before we print the curly bracket that denotes the end of array X, we need to decrease the indentation for 2. After the curly bracket, we print the comma and a new line and continue printing the values from the main array.