Problem D. Decompress and Sort

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Your task is to sort the given N strings $s_1, s_2, \ldots, s_N$ which are given in compressed form.

Compression works as follows. An m-times repetition of a non-empty character sequence *seq* can be compressed into "m(*seq*)", where $m \geq 2$ is an integer. When *seq* consists of just one letter `c`, we may omit the parentheses and write "m`c`". Formally, a compressed string is defined by the following BNF grammar:

```
<String> ::= <Letter> | <Number> <Letter> | <Number> '(' <String> ')' | <String> <String>
<Letter> ::= 'A' | 'B' | ... | 'Z'
<Number> ::= <Digit> | <Number> '0' | <Number> <Digit>
<Digit> ::= '1' | '2' | ... | '9'
```

For example, `ACMACMACMICPCICPCACMACMACMICPCICPCXXXXXXXXXX` can be compressed into:

`3(ACM)ICPCICPCACMACMACMICPCICPCXXXXXXXXXX`

by replacing the first occurrence of `ACMACMACM` with its compressed form. Similarly, by replacing the following repetitions of `ICPC`, `ACM`, and `X`, we get:

`3(ACM)2(ICPC)3(ACM)2(ICPC)10X`

Since `X` is a single letter, parentheses are omitted in this compressed representation. Finally, we have:

`2(3(ACM)2(ICPC))10X`

by compressing the repetitions of `3(ACM)2(ICPC)`. As you may notice from this example, parentheses can be nested.

Input

The first line contains an integer N, the number of compressed strings ($1 \leq N \leq 50$).

The i-th of the following N lines contains the compressed string s_i. The length of the given compressed form is between 1 and 2000 characters, inclusive. Each decompressed string consists of uppercase English letters, and its length is at most 10^{10}.

Output

Print N lines. If the decompressed s_x is the i-th lexicographically smallest string, print x on the i-th line. Equal strings must be sorted in the order in which they appear in the input.

Examples

standard input	*standard output*
4 3(IC)PC I3(CI) ICICICPC 2(5I)	2 1 3 4
5 2(2(2(2X))) 4(4X) 16X 8X8X 8XX	5 1 2 3 4