Problema B

Fórmulas Proposicionais Bem Formadas

O objectivo deste problema é a da análise de fórmulas proposicionais e da sua boa formação.

Por exemplo as strings

```
(X -> Y)
(False & ((True & Y) | (X <-> Y)))
((False) & ((True & Y) | (X <-> Y)))
```

representam fórmulas proposicionais bem formadas (bem parenteseadas, todos os operadores lógicos são usados devidamente). Em contraste, as strings seguintes não representam fórmulas bem formadas:

```
( & ((True & Y) | (X <-> Y)))
((False & (!(True & Y) | (X <-> Y)))
```

Na primera há um uso desadequado da conjunção, e a segunda fórmula está mal parenteseada.

Tarefa

Implemente um programa que, recebendo uma fórmula em entrada, verifique se está bem formada ou não.

Input

A fórmula por analisar é dada de uma forma faseada, elemento por elemento, um por linha.

`True` representa o valor de verdade, `False` o valor falso, qualquer outra string de caracteres é uma variável proposicional (assuma que usaremos somente letras maiúsculas para simplificar). Usamos igualmente as parêntesis habituais `( )`, o caractere `&` para a conjunção, `!` para a negação, `|` para a disjunção, `->` para a implicação e finalmente `<->` para a equivalência.

O input é então organizado da seguinte forma. Na primeira linha indica-se com um inteiro N o tamanho da fórmula (quantos elementos sintácticos constituem a fórmula - ver exemplo).

As N linhas introduzem na ordem e um a um os elementos da fórmula.

Output

Uma linha com a palavra `YES` se a fórmula em entrada estiver bem formada ou a a palavra `NO`, caso contrário.

Constraints

$0 \leq N \leq 100$

Input 1

```
17
(
False
&
(
(
True
&
Y
)
|
(
X
<->
Y
)
)
)
```

Output 1

```
YES
```

Input 2

```
18
(
(
False
&
(
(
True
&
Y
)
|
(
X
<->
Y
)
)
)
```

Output 2

```
NO
```