Gramática

```
<configuración> ::= "configuración" <nombre> "{" <secciones> "}"
<secciones>     ::= { <sección> }
<sección>       ::= "sección" <nombre> "{" <parámetros> "}"
<parámetros>    ::= { <parámetro> }
<parámetro>     ::= <nombre> "=" <valor> ";"
<valor>         ::= <número> | <cadena> | <booleano>
<número>        ::= <dígito> { <dígito> }
<cadena>        ::= "" { <carácter> } ""
<booleano>      ::= "verdadero" | "falso"
<nombre>        ::= <letra> { <letra> | <dígito> }
<carácter>      ::= <letra> | <dígito> | " " | <símbolo>
<letra>         ::= "a" | "b" | ... | "z" | "A" | "B" | ... | "Z"
<dígito>        ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
<símbolo>       ::= "." | "," | "!" | "?" | ":" | ";" | "'"
```

```
grammar LuciaLuconi;

// ========================

// Reglas del Parser

// ========================

configuracion : CONFIGURACION NOMBRE LLAVE_ABRE secciones
LLAVE_CIERRA ;

secciones : seccion* ;

seccion : SECCION NOMBRE LLAVE_ABRE parametros LLAVE_CIERRA ;

parametros : parametro* ;

parametro : NOMBRE IGUAL valor PUNTO_Y_COMA ;

valor : NUMERO

| CADENA

| BOOLEANO ;

// ========================

// Tokens con nombre simbólico
```

```
// ========================

CONFIGURACION : 'configuración' ;

SECCION : 'sección' ;

IGUAL : '=' ;

LLAVE_ABRE : '{' ;

LLAVE_CIERRA : '}' ;

PUNTO_Y_COMA : ';' ;

// ========================

// Tokens generales

// ========================

NOMBRE : [a-zA-Z] [a-zA-Z0-9]* ;

NUMERO : [0-9]+ ;

CADENA : '\'\'\'' .*? '\'\'\'' ;

BOOLEANO : 'verdadero' | 'falso' ;

// ========================

// Ignorar espacios

// ========================

WS : [ \t\r\n]+ -> skip ;
```