
PROCESADORES DE LENGUAJES

MEMORIA DE PROYECTO - HITO 1: ANALIZADOR LÉXICO

Grupo 10

SERGIO COLET GARCÍA
LAURA MARTÍNEZ TOMÁS
RODRIGO SOUTO SANTOS
LI JIE CHEN CHEN

*GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID*



Índice general

1. Introducción	5
2. Clases léxicas	7
2.1. Palabras reservadas	7
2.2. Literales	7
2.3. Identificadores	7
2.4. Símbolos de operación y puntuación	7
3. Especificación formal del léxico	9
3.1. Definiciones auxiliares.	9
3.2. Definiciones de cadenas ignorables.	9
3.3. Definiciones léxicas.	9
4. Diseño de un analizador léxico	11
Índice de figuras	13

1 | Introducción

2 | Clases léxicas

2.1. Palabras reservadas

Para poder analizar de manera correcta, será necesario establecer una clase léxica por cada palabra reservada. En el lenguaje de esta práctica, *Tiny(0)*, contamos con 3 palabras reservadas, utilizadas para definir el tipo de las variables. Tendremos pues, una palabra para las variables de tipo booleano, otra para las de tipo entero y una última para las reales. Las palabras son las definidas a continuación, contando cada una con una clase léxica.

- *bool* → Variables booleanas.
- *int* → Variables enteras.
- *real* → Variables reales.
- *and* → Conjunción lógica.
- *or* → Disyunción lógica.
- *not* → Negación lógica.

2.2. Literales

- **Literales booleanos.** Toma como valor las palabras reservadas *true* o *false*. Su clase léxica será *literalBooleano*.
- **Literales enteros.** Opcionalmente empiezan con un signo más (+) o menos (-), y después debe aparecer una secuencia (que empieza por un número distinto de 0) de 1 o más dígitos. Su clase léxica será *literalEntero*.
- **Literales reales.**

2.3. Identificadores

Los identificadores nos sirven para poder ponerle un nombre a las variables. Éstos deben comenzar por un subrayado (_) o una letra, seguida de una secuencia de 0 o más subrayados, dígitos o letras. Su clase léxica será *identificador*.

2.4. Símbolos de operación y puntuación

Cada uno de ellos tendrá su propia clase léxica. En el subconjunto del lenguaje en el que trabajamos, *Tiny(0)*, contamos con las siguientes clases:

- **Suma.** Se representa con el símbolo más (+). Su clase léxica será *operadorSuma*.
- **Resta.** Se representa con el símbolo menos (-). Su clase léxica será *operadorResta*.
- **Multiplicación.** Se representa con el símbolo asterisco (*). Su clase léxica será *operadorMul*.
- **División.** Se representa con el símbolo barra (/). Su clase léxica será *operadorDiv*.
- **Menor.** Se representa con el símbolo menor que (<). Su clase léxica será *operadorMenor*.
- **Mayor.** Se representa con el símbolo mayor que (>). Su clase léxica será *operadorMayor*.
- **Igual.** Se representa con el dos símbolos de igualdad seguidos (==). Su clase léxica será *operadorIgual*.
- **Menor o igual.** Se representa con el símbolo menor que seguido del símbolo de igualdad (<=). Su clase léxica será *operadorMenIgual*.

- **Mayor o igual.** Se representa con el símbolo mayor que seguido del símbolo de igualdad (\geq). Su clase léxica será *operadorMayIgual*.
- **Asignación.** Se representa con el símbolo un símbolo de igualdad ($=$). Su clase léxica será *operadorAsig*.
- **Paréntesis de apertura.** Se representa con el símbolo del paréntesis de apertura (" $($ ", sin comillas). Su clase léxica será *parentesisAp*.
- **Paréntesis de cierre.** Se representa con el símbolo del paréntesis de cierre (" $)$ ", sin comillas). Su clase léxica será *parentesisCi*.
- **Punto y coma.** Se representa con el símbolo punto y coma ($;$). Su clase léxica será *puntoYComa*.

3 | Especificación formal del léxico

3.1. Definiciones auxiliares.

$letra \rightarrow A|B|\dots|Z|a|b|\dots|z$
 $digitoPositivo \rightarrow 1|\dots|9$
 $digito \rightarrow digitoPositivo|0$
 $parteEntera \rightarrow digitoPositivodigito^*$
 $parteDecimal \rightarrow digito^*digitoPositivo$
 $parteExponencial \rightarrow (e|E)(\backslash+|\backslash-|)parteEntera$

3.2. Definiciones de cadenas ignorables.

3.3. Definiciones léxicas.

$bool \rightarrow \mathbf{bool}$
 $int \rightarrow \mathbf{int}$
 $real \rightarrow \mathbf{real}$
 $and \rightarrow \mathbf{and}$
 $or \rightarrow \mathbf{or}$
 $not \rightarrow \mathbf{not}$
 $literalBooleano \rightarrow \mathbf{true}|\mathbf{false}$
 $literalEntero \rightarrow [\backslash+|\backslash-]parteEntera$
 $literalReal \rightarrow [\backslash+|\backslash-]parteEntera(\cdot parteDecimal|parteExponencial|parteDecimalparteExponencial)$
 $identificador \rightarrow (_|letra)(letra|digito|_)^*$
 $operadorSuma \rightarrow \backslash+$
 $operadorResta \rightarrow \backslash-$
 $operadorMul \rightarrow \backslash*$
 $operadorDiv \rightarrow \backslash/$
 $operadorMenor \rightarrow \backslash<$
 $operadorMayor \rightarrow \backslash>$
 $operadorIgual \rightarrow \backslash==$
 $operadorMenIgual \rightarrow \backslash<=$
 $operadorMayIgual \rightarrow \backslash>=$
 $operadorAsig \rightarrow \backslash=$
 $parentesisAp \rightarrow \backslash($
 $parentesisCi \rightarrow \backslash)$
 $puntoYComa \rightarrow \backslash;$

4 | Diseño de un analizador léxico

Índice de figuras