

Práctica PL

Elena Kaloyanova Popova y Álvaro Borja Velasco García

2018

Índice general

1. Introducción	2
2. Fase 1: Analizador léxico	3
2.1. Clases Léxicas	3
2.2. Especificación Formal	4
2.3. Diseño	6

Capítulo 1

Introducción

Esta práctica consistirá en el desarrollo de un procesador de lenguajes sobre el siguiente lenguaje:

Capítulo 2

Fase 1: Analizador léxico

2.1. Clases Léxicas

Todo programa consta de dos secciones: una para las declaraciones y otra para las instrucciones, separadas por un token «&&». La sección de declaraciones está formada por una serie de declaraciones compuestas por el nombre de tipo y el de variable y separadas por un punto y coma. La sección de instrucciones, por su parte, consta de una serie de asignaciones (variable=expresión), separadas también por un punto y coma. Las clases léxicas que hemos considerado para representar los tokens del lenguaje son las siguientes:

- **SEC:** Representa el seccionador de las dos partes del programa («&&»).
- **NUM:** Palabra reservada «num».
- **BOOL:** Palabra reservada «bool».
- **VAR:** Representa el nombre de la variable. Comienza necesariamente por una letra, seguida por una secuencia de cero o más letras, dígitos o el símbolo «_».
- **ASIG:** Representa el signo igual de las asignaciones.
- **NXT:** Representa el signo «;» que marca el comienzo de la siguiente instrucción.
- **TRUE:** Palabra reservada «true».
- **FALSE:** Palabra reservada «false».

- **NUMR:** Representa un número real. Puede empezar opcionalmente con un signo seguido de una secuencia de uno o más dígitos cualesquiera, pudiendo poner ceros no significativos a la izquierda. Puede opcionalmente estar seguido por una parte decimal y/o una parte exponencial.
- **MAS:** Operador suma ($\backslash +$).
- **MENOS:** Operador resta ($\backslash -$).
- **POR:** Operador multiplicación ($\backslash *$).
- **DIV:** Operador división ($\backslash /$).
- **AND:** Palabra reservada «and».
- **OR:** Palabra reservada «or».
- **NOT:** Palabra reservada «not».
- **MAY:** Operador mayor ($>$).
- **MEN:** Operador menor ($<$).
- **MAYI:** Operador mayor o igual ($>=$).
- **MENI:** Operador menor o igual ($<=$).
- **IGUAL:** Operador igual a ($==$).
- **DIST:** Operador distinto a ($!=$).
- **PAP:** Signo de apertura de paréntesis.
- **PCI:** Signo de cierre de paréntesis.
- **EOF:** Representa el final de fichero.

2.2. Especificación Formal

Las definiciones regulares correspondientes a las clases léxicas definidas son:

- (★) **SEC:** $[\&][\&]$
- (★) **VAR:** $\text{LETRA}([\text{LETRA}|\text{DIG}|\backslash_]*)$
 LETRA: $([a-z,A-Z])$

DIG: ([0-9])
 (★) **NUM:** ([n][u][m])
 (★) **BOOL:** ([b][o][o][l])
 (★) **TRUE:** ([t][r][u][e])
 (★) **FALSE:** ([f][a][l][s][e])
 (★) **NUMR:** SIGNO?(DIG+(DEC)?(EX)?)
 DEC: (\.)DIG+
 EX: [e|E](SIGNO?DIG+)
 SIGNO: [\+, \-]
 DIG: [0-9]
 (★) **AND:** ([a][n][d])
 (★) **OR:** ([o][r])
 (★) **NOT:** ([n][o][t])
 (★) **MAS:** (\+)
 (★) **MENOS:** (\-)
 (★) **DIV:** (/)
 (★) **POR:** (*)
 (★) **MAY:** (>)
 (★) **MEN:** (<)
 (★) **MAYI:** ([>][=])
 (★) **MENI:** ([<][=])
 (★) **IGUAL:** ([=][=])
 (★) **DIST:** (![=])
 (★) **ASIG:** (=)
 (★) **NXT:** (;)
 (★) **PAP:** (\()
 (★) **PCIERRE:** (\))
 (★) **SEP:** [« », \t, \n, \r, \b]

2.3. Diseño

