



ESCUELA TÉCNICA
SUPERIOR DE INGENIEROS
INFORMÁTICOS

UNIVERSIDAD POLITÉCNICA DE MADRID

Procesadores de Lenguajes: Memoria del Proyecto

Jose Luis Prado Sierra- 220070
Alejandro Gragera Serradilla- 22M043
Antonio Bielza Díez- 22M049

1. Analizador Léxico

Durante el desarrollo del Analizador Léxico hemos descrito una serie de objetos y estructuras matemáticas indispensables para que su diseño, desarrollo y función final cumplan con las expectativas de un Procesador de Lenguajes.

1.1. Tokens

Los tokens son duplas que se generan cuando el Analizador Léxico encuentra una concatenación de caracteres que identifica como válida.

Contienen la información necesaria para que las reciba el Analizador Sintáctico y se componen de un código que los identifica y un atributo opcional que puede servir para diferenciarlos de otros tokens con el mismo código o aportar información extra.

Hemos definido los siguientes tokens en función de las necesidades de nuestra práctica:

- Boolean: <BOOLEAN, >
- Else: <ELSE, >
- Float: <FLOAT, >
- Function: <FUNCTION, >
- If: <IF, >
- Int: <INT, >
- Let: <LET, >
- Read: <READ, >
- Return: <RETURN, >
- String: <STRING, >
- Void: <VOID, >
- Write: <WRITE, >
- Constante real: <REALCONST, >
- Constante entera: <INTCONST, >
- Cadena: <STR, c*"
- Identificador: <ID, posTS>
- Suma con asignación (+=): <PLUSEQ, >
- Igual (=): <EQ, >
- Coma (,): <COMMA, >
- Punto y coma (;): <SEMICOLON, >
- Paréntesis abierto (): <OPPAR, >
- Paréntesis cerrado ()): <CLPAR, >
- Llave abierta {}: <OPBRA, >

- Llave cerrada (}): <CLBRA, >
- Suma (+): <SUM, >
- Y Lógico (&): <AND, >
- Menor (<): iMINORTHAN, >
- false: <FALSE, >
- true: <TRUE, >
- EOF: <EOF, >

Todos los tokens anteriores conforman todos los obligatorios, los específicos y los opcionales de la práctica.

1.2. Gramática

La gramática es la estructura matemática que define el lenguaje a generar, en nuestro caso es para una versión reducida de JS. Definir la gramática es una forma de estructurar por tanto el lenguaje y asegurarnos de que vamos a generar solo lo que queremos.

Hemos definido las siguientes reglas:

$$S \rightarrow delS \mid +A \mid \&B \mid 'C \mid dD \mid lF \mid _F \mid /G \mid = \mid , \mid ; \mid < \mid (\mid) \mid \{ \mid \}$$

$$A \rightarrow = \mid \lambda$$

$$B \rightarrow \&$$

$$C \rightarrow cC \mid '$$

$$D \rightarrow dD \mid .E \mid \lambda$$

$$E \rightarrow dE \mid d$$

$$F \rightarrow lF \mid dF \mid _F \mid \lambda$$

$$G \rightarrow /H$$

$$H \rightarrow c_2H \mid \backslash nS$$

Los conjuntos definidos para el desarrollo de la gramática y del resto de la práctica son:

Conjunto l: representa cualquier letra.

Conjunto d: representa cualquier dígito.

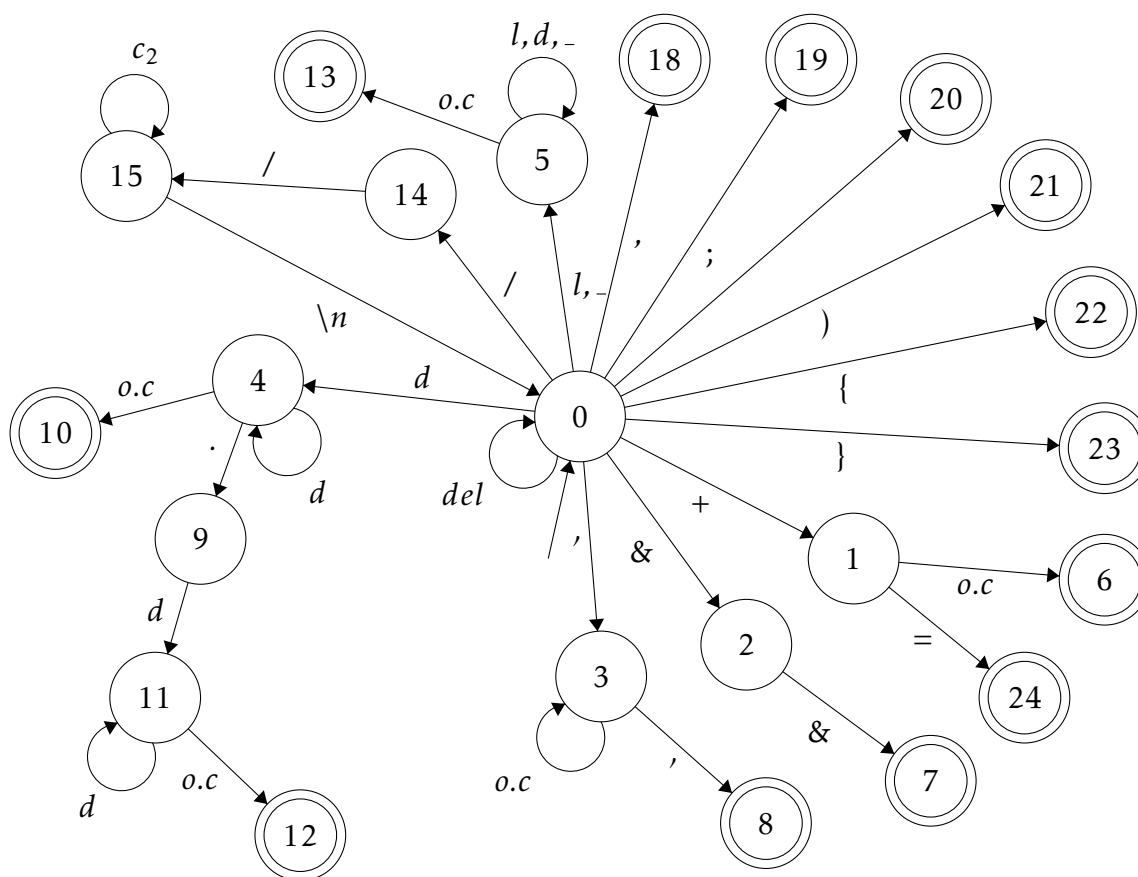
Conjunto c: representa cualquier caracter.

Conjunto c_2 : representa cualquier caracter sin $\backslash n$.

1.3. Autómata

Un autómata es otro tipo de estructura matemática capaz de, al contrario que una gramática que se centra en generar un lenguaje, un autómata se encarga de comprenderlo.

El autómata que hemos planteado capaz de entender todas las palabras válidas de nuestro lenguaje es el siguiente:



El autómata tiene que ir acompañado de una serie de acciones que realizar mientras se recorre para poder generar los tokens mientras vamos leyendo el archivo, que vamos a plantear en el siguiente punto.

1.4. Acciones Semánticas

Las acciones semánticas son una serie de acciones que se realizan entre las transiciones del autómata que permiten realizar diferentes funciones como por ejemplo ir generando un string para cuando detectamos una cadena.

Las acciones semánticas que hemos definido para nuestro proyecto son:

0:0 Leer
0:18 Leer, gentoken(iCOMMA, $\dot{\iota}$)
0:19 Leer, gentoken(iSEMICOLON, $\dot{\iota}$)
0:20 Leer, gentoken(iOPPAR, $\dot{\iota}$)
0:21 Leer, gentoken(iCLPAR, $\dot{\iota}$)
0:22 Leer, gentoken(iOPBRA, $\dot{\iota}$)
0:23 Leer, gentoken(iCLBRA, $\dot{\iota}$)
0:1 Leer
1:6 gentoken(iSUM, $\dot{\iota}$)
1:24 Leer, gentoken(iEQ, $\dot{\iota}$)
0:2 Leer
2:7 Leer, gentoken(iAND, $\dot{\iota}$)
0:3 Leer, $str = ''$
3:3 Leer, $str = str + c$
3:8 Leer, gentoken(iSTRING, $str\dot{\iota}$)
0:4 Leer, $num = d$
4:4 Leer, $num = num * 10 + d$
4:10 gentoken(iINTCONST, $num\dot{\iota}$)
4:9 Leer, $cont = 1$
9:11 Leer, $num = num * 10 + d$, $cont++$
11:11 Leer, $num = num * 10 + d$, $cont++$
11:12 $num = num * 10^{-cont}$, gentoken(iREALCONST, $num\dot{\iota}$)
0:14 Leer
0:15 Leer
15:15 Leer
15:0 Leer
0:5 Leer, $id = c$
5:5 Leer, $id = id + c$

5:13

```

if(id in palabrasReservadas)
    postTS = insertTS(id)
    gentoken(<toUpper(id), postTS>)
else
    postTS = insertTS(id)
    gentoken(<ID, postTS>)

```

1.5. Errores

Todo buen procesador de lenguajes debe de avisar al usuario de si existe algún error y además se capaz de tratar todos los posibles errores para una ejecución robusta y comprensible.

Hemos detectado los siguientes casos capaces de generar algún error y que por tanto tratamos en nuestro código:



Figura 1: A goose.

The radius of a bubble under a varying pressure field is defined by the Rayleigh-Plesset equation. This equation is derived using standard conservation law under a number of simplifying assumptions:

$$\rho_l \left(R\ddot{R} + \frac{3}{2}\dot{R}^2 \right) = p_{gas} - P(t) - P_0 + \frac{R}{c} \frac{dp_{gas}}{dt} - 4\mu \frac{\dot{R}}{R} - \frac{2\sigma}{R} \quad (1)$$

In the above equation $R(t)$ [m] represents the bubble radius. The other terms are: ρ_l [$kg \cdot m^3$] the density of the liquid, p_{gas} [Pa] the pressure of the gas inside the bubble, $P(t)$ [Pa] the imposed oscillatory pressure field, P_0 [Pa] the baseline pressure, c [m/s] speed of sound in the liquid, μ [$Pa \cdot s$] molecular viscosity of the liquid and σ [$kg \cdot s^{-2}$] the surface tension at the bubble-water interface.

For more contextual information on the bubble dynamic phenomena, please see the following sources: [?], [?], and [?].

2. Tabla de Símbolos

This section answers the individual questions of the project description. For each question, provide an answer and short analysis.

Question 1: Forced bubble oscillation

(a)

References to equations can be written out in latex (1). Similarly, figures 1 and sections 2.1 may also be referenced.

(b)

Citations require the mybib.tex file to be extended with the desired references. Students can use JabRef (<http://www.jabref.org>) to construct the mybib.bib file. Students are invited to link their Mendelay, CiteULike and Zotero account directly to Overleaf.

2.1. Question 2: Bubble evolution

3. Analizador Sintáctico

4. Analizador Semántico

Students' contributions

Mr. Goose and Mrs. Goose worked together to understand the problem and write the numerical codes. The summary of the problem and Q2 were written up by Mr. Goose, Q1 and Q3 were completed by Mrs. Goose. Both students corrected the final report.