# MEMORIA PRÁCTICA ANALIZADOR LÉXICO

Grupo 54:

* Marcos Carnerero Blanco
* Rodrigo Elola Torrijos
* Miguel López García

# Diseño del Analizador Léxico

Para esta práctica hemos implementado los siguientes Tokens:

<EOF,-> End Of File <INPUT,-> Input

<VAR,-> Var <AUTOINCREMENTO,-> ++

<BOOLEAN,-> Boolean <INTEGER,-> Integer

<VOID,-> Void <COMA,-> ,

<BREAK,-> Break <OUTPUT,-> Output

<CteENTERA, Numero> Constante entera <DosPUNTOS,-> :

<FUNCTION,-> Function <STRING,-> String

<ID,Numero> Identificador <EOS,-> ;

<CASE,-> Case <RETURN,-> Return

<CteCADENA,-> Cadena “ <ParentesisABRE,-> (

<ParentesisCIERRA,-> ) <IF,-> if

<SWITCH,-> Switch <ASIG,-> Asignación

<LlaveABRE,-> { <LlaveCIERRA,-> }

<NOT,-> ! <MAYOR,-> >

<MULT,-> Multiplicación \* <DEFAULT,-> Default

<FALSE,-> False <TRUE,-> True

Tras decidir el diseño de los Tokens anteriormente citados, hemos procedido a crear la gramática de la siguiente manera:

S 🡪 del S | /A | lB | dC | “D | +E | ( | ) | { | } | , | ; | : | ! | = | > | ‘ | \* | EOF

A 🡪 \*A’ l: letra

A’ 🡪 c1A’ | \*A’’ d: dígito

A’’ 🡪 /S | c2A’ | \*A’’ c1: Cualquier carácter exceptuando \*

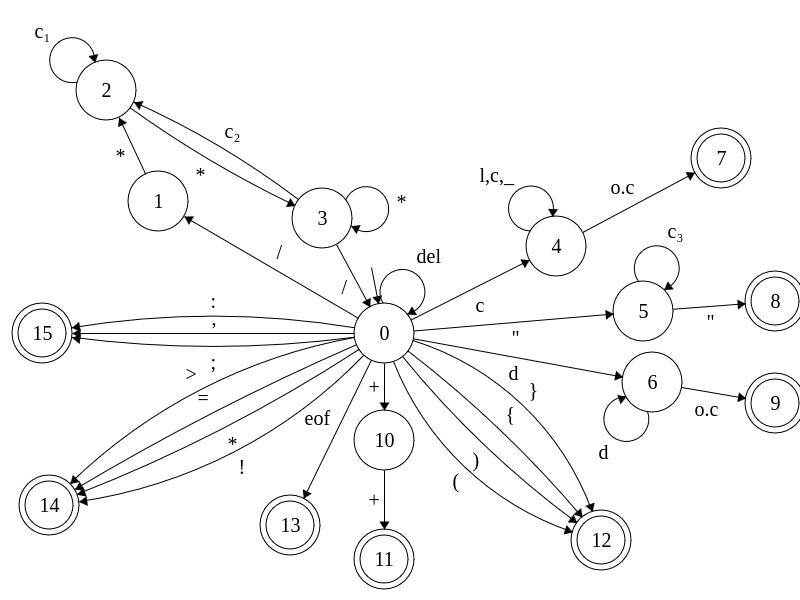
B 🡪 lB | dB | \_B | λ c2: Cualquier carácter menos \* o /

C 🡪 dC | λ c3: Cualquier carácter menos “

D 🡪 c3D | “

E 🡪 +

Con la gramática anterior, construimos el siguiente Autómata Finito Determinista:



Como se puede apreciar, muchos caracteres podrían transitar hacia el mismo estado final, pero los hemos separado para favorecer la limpieza, claridad y comprensión del autómata.

También destacamos las siguientes acciones semánticas:

* Leer🡪 L car := leer() //en todas menos 6:7, 10:11
* Concatenar 🡪 C lex := car //0:6

C’ lex := lex + car //6:6, 8:8

C\* lex := “ //0:8

* Número 🡪 N num := valor (car)

N’ num := num \* 10 + valor

* Generar tokens 🡪 Gen\_token (CteENTERA,num)

Gen\_token (CteCADENA, lex)