Procesador de Lenguaje JavaScript PL

Grupo 46

Sofía Hernández Montero
18M046

Jaime González Delgado
18M048

Fernando Bellido Pazos
18M008

UNIVERSIDAD
POLITÉCNICA
DE MADRID

Universidad Politécnica de Madrid
Grado de Matemáticas e Informática
Procesadores de Lenguajes
2020-2021

Contenido

Objetivos

La Práctica consistirá en el diseño y construcción de un Analizador de una versión del lenguaje JavaScript llamado JavaScript-PDL.

Objetivos comunes

- La estructura general de un programa compuesto por funciones, declaraciones y sentencias.
- Definición de funciones.
- Tipos enteros, lógicos y cadenas.
- Variables y su declaración.
- Constantes enteras y cadenas de caracteres.
- Sentencias: asignación, condicional simple, llamada a funciones y retorno.
- Expresiones.
- Comentarios.
- Operaciones de entrada/salida por terminal:
    - input
    - alert
- Operadores:
    - Aritméticos: +, −
    - Relacionales: ==, ! =
    - Lógicos: &&, ||

Objetivos específicos

- Sentencias: **Sentencia repetitiva (do-while)**
- Operadores especiales: **Asignación con resta (-=)**
- Técnicas de Análisis Sintáctico: **Ascendente**
- Comentarios: **Comentario de bloque (/* */)**
- Cadenas: **Con comillas dobles (" ")**

Analizador Léxico

"Un analizador léxico o analizador lexicográfico es la primera fase de un compilador, consistente en un programa que recibe como entrada el código fuente de otro programa y produce una salida compuesta de tokens"

– Wikipedia

Tokens

<abrirCorchete, ->	<cadena, lexema>	<return, ->
<cerrarCorchete, ->	<restaAsignacion, ->	<input, ->
<abrirParentesis, ->	<opAritmetico, 2>	<alert, ->
<cerrarParentesis, ->	< cteEntera , digito>	<if, ->
< puntoYcoma , ->	<opAritmetico, 1>	<number, ->
<coma, ->	<opLogico, 2>	<boolean, ->
<opRelacional, 2>	<opLogico, 1>	<string, ->
<opRelacional, 1>	<do, ->	<let, ->
<asignacion, ->	<while, ->	<EOF, ->
<identificador, posTS>	<function, ->	

Operador Artimetico	Operador Logico	Operador relacional
1: +	1: &&	1: ==
2: -	2: ||	2: !=

Gramática del Lenguaje

S → $delS \mid lA \mid "C \mid dE \mid -G \mid |i \mid \&J \mid = N \mid !Q \mid /U \mid +\mid c.e$

A → $dA \mid lA \mid o.c \mid \_A$

C → $c_1C \mid "$

E → $dE \mid o.c$

G → $= \mid o.c$

I → $\mid$

J → &

N → $= \mid o.c$

Q → $=$

U → $*Y$

Y → $c_2Y \mid *Z$

Z → $c_3Y \mid *Z \mid /S$

Leyenda

c1 = cualquier carácter | {"}

c2= cualquier carácter | {}*

c3= cualquier carácter | {, /}*

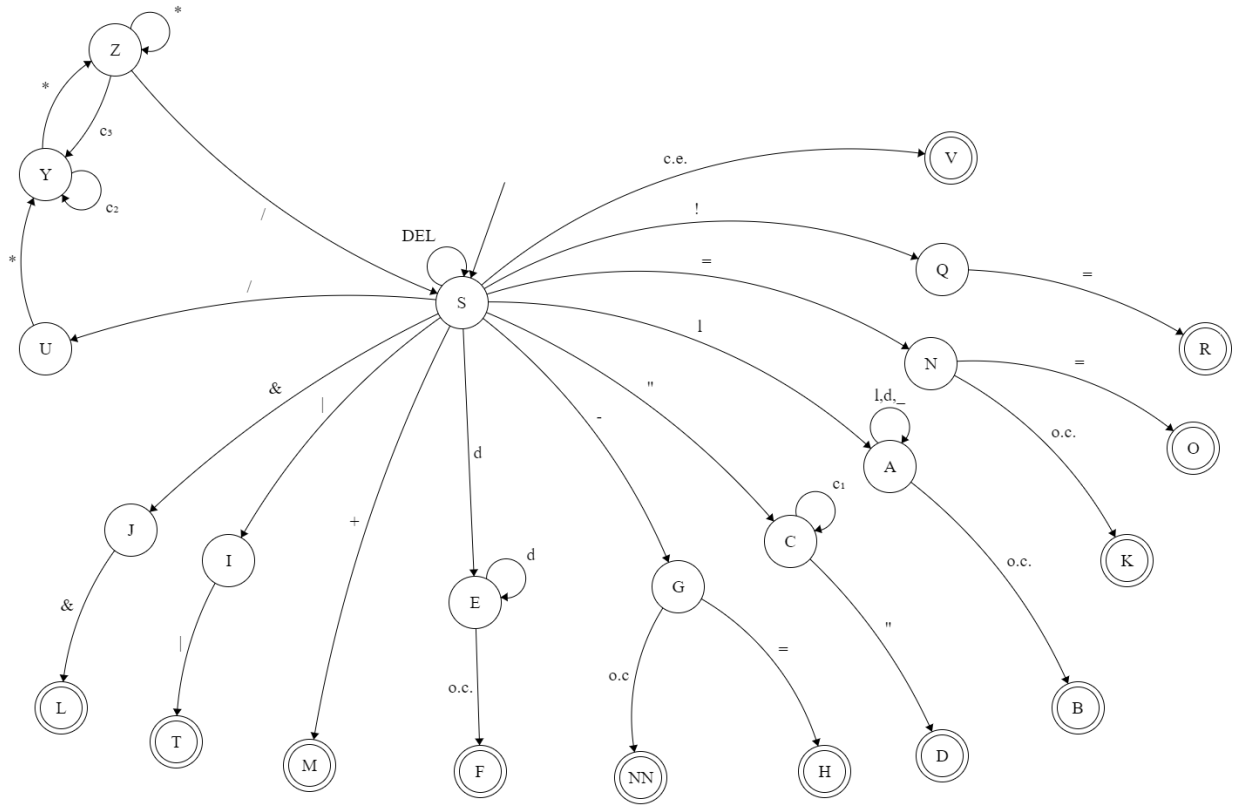
c.e = caracteres especiales: {,},(,), ; , ','

d = dígitos del 0 al 9

l = letras de la a – z, A – Z

del = delimitadores: eol, tab, esp, etc.

Autómata Finito Determinista



Acciones Semánticas

S→V: Comprobar tipo carácter especial: Enviar token correspondiente; Leer;

S→Q: leer;

Q→R: Gen_Token(<opRelacional,2>)

S→N: leer;

N→K: Gen_Token(<asignación, ->)

N→O: Gen_Token(<opRelacional,1>); leer;

S→A: lexema=c;leer; //(Siendo c el carácter leido)

A→A: lexema=lexema ⊕c; leer; //(Siendo c el carácter leído)

A→B:

```
if(isReservada){
    Gen_Token(<lexema,->) // Tomamos el mismo nombre de la palabra
    puesto que es
                                Case-Sensitive JS
}else{
    Gen_Token(<identificador,posTs>); // Siendo posTS =
```

GestorTablaSimbolos.insertar(lexema);

} // El gestor se encarga de los problemas

S→C: lexema= “”; leer();

C→C: lexema= lexema⊕ c_1; leer();

C→D:

if(lexema.length >64){

error

}else{

Gen_Token(<Cadena,lexema>);

}

leer;

S→G: leer();

G→H: Gen_Token(<restaAsignacion,->); leer();

G→NN: Gen_Token(<opAritmetico,2>); leer();

S→E: digito=char2Int(d); leer();

E→E: digito=digito*10+ char2Int(d); leer();

E→F: if(digito > 2^16 -1){

error

}else{

Gen_Token(<Cte-entera,digito>);

}

S→M: Gen_Token(<opAritmetico,1>); leer();

S→I:leer();

I→T: Gen_Token(<opLogico,2>);leer();
S→J: leer();

J→L: Gen_Token(<opLogico,1>)

Autómata Tabular

	del	l	d	-	"	|(or)	&	=	!	/	ce	c1	c2	c3	+	*	o.c	_
S	S	A	E	G	C	I	J	N	Q	U	V				M			
A		A	A														B	A
C					D							C						
E			E														F	
G								H									NN	
I						T												
J							L											
N								O									K	
Q								R										
U																Y		
Y													Y			Z		
Z										S				Y		Z		

Donde la posición (fila, columna) representa al estado que se llega desde el estado <fila> con carácter <columna>

Tabla de Símbolos

Diseño

Lexema	Tipo	Desplazamiento	NºParametros	returnType	eti

La tabla es no homogénea, ya que por cada parámetro de una función se añade una columna “tipo” indicando el tipo de parámetro que es el i-ésimo parámetro de la función

Ejemplo de Output Visual

A pesar de devolver un fichero de texto, según las especificaciones dadas por el profesorado. Se ha decido hacerlo también en un mini interfaz de usuario

Nota: El valor de “error” es porque nos apeteció, no para confundir a nadie

Analizador Sintáctico

Gramática

Aumentada

$Z \rightarrow P$

Inicio

$P \rightarrow BP \mid FP \mid \lambda$

Funciones

$F \rightarrow I\ J\ G$

$I \rightarrow \text{function}\ H\ \text{id}$

$J \rightarrow (\ A\ )$

$G \rightarrow \{\ C\ \}$

$H \rightarrow T \mid \lambda$

$A \rightarrow T\ \text{id}\ K \mid \lambda$

$K \rightarrow ,\ T\ \text{id}\ K \mid \lambda$

$C \rightarrow BC \mid \lambda$

Sentencias:

Simples

$S \rightarrow \text{id} = E\ ; \mid \text{id}\ (\ L\ )\ ; \mid \text{alert}\ (\ E\ )\ ; \mid \text{input}\ (\ \text{id}\ )\ ; \mid \text{return}\ X\ ; \mid \text{id} \text{ -= } L\ ;$

Compuestas

$B \rightarrow \text{if}\ (\ E\ )\ S \mid \text{let}\ T\ \text{id}\ ; \mid S \mid \text{do}\ \{\ C\ \}\ \text{while}\ (\ E\ )$

$T \rightarrow \text{number} \mid \text{boolean} \mid \text{string}$

Expresiones

$E \rightarrow E \mid\mid R \mid R$

$R \rightarrow R \&\& U \mid U$

$U \rightarrow U == V \mid U \mathrel{!}= V \mid V$

$V \rightarrow V + W \mid V - W \mid W$

$W \rightarrow \text{id} \mid (\ E\ ) \mid \text{id}\ (\ L\ ) \mid \text{entero} \mid \text{cadena}$

Argumentos de la función

$L \rightarrow E\ Q \mid \lambda$

$Q \rightarrow ,\ E\ Q \mid \lambda$

Return

$X \rightarrow E \mid \lambda$

LR(1)

A continuación, dejamos las tablas de First´s y Follows , tabla del cálculo de la colección canónica, tabla Acción y tabla GoTo, en ese mismo orden.

FIRST / FOLLOW table		
Nonterminal	**FIRST**	**FOLLOW**
Z	{'',if,let,identificador,alert,input,return,do,function}	{$}
P	{'',if,let,identificador,alert,input,return,do,function}	{$}
F	{function}	{$,if,let,identificador,alert,input,return,do,function}
I	{function}	{abrirParentesis}
J	{abrirParentesis}	{abrirCorchete}
G	{abrirCorchete}	{$,if,let,identificador,alert,input,return,do,function}
H	{'',number,boolean,string}	{identificador}
A	{'',number,boolean,string}	{cerrarParentesis}
K	{coma,''}	{cerrarParentesis}
C	{'',if,let,identificador,alert,input,return,do}	{cerrarCorchete}
S	{identificador,alert,input,return}	{$,if,let,identificador,alert,input,return,do,function,cerrarCorchete}
B	{if,let,identificador,alert,input,return,do}	{$,if,let,identificador,alert,input,return,do,function,cerrarCorchete}
T	{number,boolean,string}	{identificador}
E	{identificador,abrirParentesis,cteEntera,cadena}	{puntoYcoma,cerrarParentesis,opLogico2,coma}
R	{identificador,abrirParentesis,cteEntera,cadena}	{puntoYcoma,cerrarParentesis,opLogico2,opLogico1,coma}
U	{identificador,abrirParentesis,cteEntera,cadena}	{puntoYcoma,cerrarParentesis,opLogico2,opLogico1,opRelacional1,opRelacional2,coma}
V	{identificador,abrirParentesis,cteEntera,cadena}	{puntoYcoma,cerrarParentesis,opLogico2,opLogico1,opRelacional1,opRelacional2,opAritmetico1,opAritmetico2,coma}
W	{identificador,abrirParentesis,cteEntera,cadena}	{puntoYcoma,cerrarParentesis,opLogico2,opLogico1,opRelacional1,opRelacional2,opAritmetico1,opAritmetico2,coma}
L	{'',identificador,abrirParentesis,cteEntera,cadena}	{cerrarParentesis,puntoYcoma}
Q	{coma,''}	{cerrarParentesis,puntoYcoma}
X	{'',identificador,abrirParentesis,cteEntera,cadena}	{puntoYcoma}

SLR closure table

Goto	Kernel	State	Closure
	{Z -> .P}	0	{Z -> .P; P -> .B P; P -> .F P; P -> .; B -> .if abrirParentesis E cerrarParentesis S; B -> .let T identificador puntoYcoma; B -> .S; B -> .do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; F -> .I J G; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma; I -> .function H identificador}
goto(0, P)	{Z -> P.}	1	{Z -> P.}
goto(0, B)	{P -> B.P}	2	{P -> B.P; P -> .B P; P -> .F P; P -> .; B -> .if abrirParentesis E cerrarParentesis S; B -> .let T identificador puntoYcoma; B -> .S; B -> .do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; F -> .I J G; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma; I -> .function H identificador}
goto(0, F)	{P -> F.P}	3	{P -> F.P; P -> .B P; P -> .F P; P -> .; B -> .if abrirParentesis E cerrarParentesis S; B -> .let T identificador puntoYcoma; B -> .S; B -> .do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; F -> .I J G; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma; I -> .function H identificador}
goto(0, if)	{B -> if.abrirParentesis E cerrarParentesis S}	4	{B -> if.abrirParentesis E cerrarParentesis S}
goto(0, let)	{B -> let.T identificador puntoYcoma}	5	{B -> let.T identificador puntoYcoma; T -> .number; T -> .boolean; T -> .string}
goto(0, S)	{B -> S.}	6	{B -> S.}
goto(0, do)	{B -> do.abrirCorchete C cerrarCorchete while a	7	{B -> do.abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma}
goto(0, I)	{F -> I.J G}	8	{F -> I.J G; J -> .abrirParentesis A cerrarParentesis}
goto(0, identif	{S -> identificador.asignacion E puntoYcoma; S	9	{S -> identificador.asignacion E puntoYcoma; S -> identificador.abrirParentesis L cerrarParentesis puntoYcoma; S -> identificador.restaAsignacion L puntoYcoma}
goto(0, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	{S -> alert.abrirParentesis E cerrarParentesis puntoYcoma}
goto(0, input)	{S -> input.abrirParentesis identificador cerra	11	{S -> input.abrirParentesis identificador cerrarParentesis puntoYcoma}
goto(0, return)	{S -> return.X puntoYcoma}	12	{S -> return.X puntoYcoma; X -> .E; X -> .; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(0, function)	{I -> function.H identificador}	13	{I -> function.H identificador; H -> .T; H -> .; T -> .number; T -> .boolean; T -> .string}
goto(2, P)	{P -> B P.}	14	{P -> B P.}
goto(2, B)	{P -> B.P}	2	
goto(2, F)	{P -> F.P}	3	
goto(2, if)	{B -> if.abrirParentesis E cerrarParentesis S}	4	
goto(2, let)	{B -> let.T identificador puntoYcoma}	5	
goto(2, S)	{B -> S.}	6	
goto(2, do)	{B -> do.abrirCorchete C cerrarCorchete while a	7	
goto(2, I)	{F -> I.J G}	8	
goto(2, identific	{S -> identificador.asignacion E puntoYcoma; S	9	
goto(2, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	
goto(2, input)	{S -> input.abrirParentesis identificador cerra	11	
goto(2, return)	{S -> return.X puntoYcoma}	12	
goto(2, function)	{I -> function.H identificador}	13	
goto(3, P)	{P -> F P.}	15	{P -> F P.}
goto(3, B)	{P -> B.P}	2	
goto(3, F)	{P -> F.P}	3	
goto(3, if)	{B -> if.abrirParentesis E cerrarParentesis S}	4	
goto(3, let)	{B -> let.T identificador puntoYcoma}	5	
goto(3, S)	{B -> S.}	6	
goto(3, do)	{B -> do.abrirCorchete C cerrarCorchete while a	7	
goto(3, I)	{F -> I.J G}	8	
goto(3, identific	{S -> identificador.asignacion E puntoYcoma; S	9	
goto(3, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	
goto(3, input)	{S -> input.abrirParentesis identificador cerra	11	
goto(3, return)	{S -> return.X puntoYcoma}	12	
goto(3, function)	{I -> function.H identificador}	13	
goto(4, abrirPare	{B -> if abrirParentesis.E cerrarParentesis S}	16	{B -> if abrirParentesis.E cerrarParentesis S; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(5, T)	{B -> let T.identificador puntoYcoma}	17	{B -> let T.identificador puntoYcoma}
goto(5, number)	{T -> number.}	18	{T -> number.}
goto(5, boolean)	{T -> boolean.}	19	{T -> boolean.}
goto(5, string)	{T -> string.}	20	{T -> string.}
goto(7, abrirCorc	{B -> do abrirCorchete.C cerrarCorchete while a	21	{B -> do abrirCorchete.C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; C -> .B C; C -> .; B -> .if abrirParentesis E cerrarParentesis S; B -> .let T identificador puntoYcoma; B -> .S; B -> .do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma}
goto(8, J)	{F -> I J.G}	22	{F -> I J.G; G -> .abrirCorchete C cerrarCorchete}
goto(8, abrirPare	{J -> abrirParentesis.A cerrarParentesis}	23	{J -> abrirParentesis.A cerrarParentesis; A -> .T identificador K; A -> .; T -> .number; T -> .boolean; T -> .string}
goto(9, asignacio	{S -> identificador asignacion.E puntoYcoma}	24	{S -> identificador asignacion.E puntoYcoma; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(9, abrirPare	{S -> identificador abrirParentesis.L cerrarPar	25	{S -> identificador abrirParentesis.L cerrarParentesis puntoYcoma; L -> .E Q; L -> .; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(9, restaAsig	{S -> identificador restaAsignacion.L puntoYcom	26	{S -> identificador restaAsignacion.L puntoYcoma; L -> .E Q; L -> .; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(10, abrirPar	{S -> alert abrirParentesis.E cerrarParentesis	27	{S -> alert abrirParentesis.E cerrarParentesis puntoYcoma; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(11, abrirPar	{S -> input abrirParentesis.identificador cerra	28	{S -> input abrirParentesis.identificador cerrarParentesis puntoYcoma}
goto(12, X)	{S -> return X.puntoYcoma}	29	{S -> return X.puntoYcoma}
goto(12, E)	{X -> E.; E -> E.opLogico2 R}	30	{X -> E.; E -> E.opLogico2 R}
goto(12, R)	{E -> R.; R -> R.opLogico1 U}	31	{E -> R.; R -> R.opLogico1 U}
goto(12, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	{R -> U.; U -> U.opRelacional1 V; U -> U.opRelacional2 V}
goto(12, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	{U -> V.; V -> V.opAritmetico1 W; V -> V.opAritmetico2 W}
goto(12, W)	{V -> W.}	34	{V -> W.}
goto(12, identifi	{W -> identificador.; W -> identificador.abrirP	35	{W -> identificador.; W -> identificador.abrirParentesis E cerrarParentesis}
goto(12, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	{W -> abrirParentesis.E cerrarParentesis; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(12, cteEnter	{W -> cteEntera.}	37	{W -> cteEntera.}
goto(12, cadena)	{W -> cadena.}	38	{W -> cadena.}
goto(13, H)	{I -> function H.identificador}	39	{I -> function H.identificador}
goto(13, T)	{H -> T.}	40	{H -> T.}
goto(13, number)	{T -> number.}	18	
goto(13, boolean)	{T -> boolean.}	19	
goto(13, string)	{T -> string.}	20	
goto(16, E)	{B -> if abrirParentesis E.cerrarParentesis S;	41	{B -> if abrirParentesis E.cerrarParentesis S; E -> E.opLogico2 R}
goto(16, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(16, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(16, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(16, W)	{V -> W.}	34	
goto(16, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(16, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(16, cteEnter	{W -> cteEntera.}	37	
goto(16, cadena)	{W -> cadena.}	38	
goto(17, identifi	{B -> let T identificador.puntoYcoma}	42	{B -> let T identificador.puntoYcoma}
goto(21, C)	{B -> do abrirCorchete C.cerrarCorchete while a	43	{B -> do abrirCorchete C.cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma}
goto(21, B)	{C -> B.C}	44	{C -> B.C; C -> .B C; C -> .; B -> .if abrirParentesis E cerrarParentesis S; B -> .let T identificador puntoYcoma; B -> .S; B -> .do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma}
goto(21, if)	{B -> if.abrirParentesis E cerrarParentesis S}	4	
goto(21, let)	{B -> let.T identificador puntoYcoma}	5	
goto(21, S)	{B -> S.}	6	
goto(21, do)	{B -> do.abrirCorchete C cerrarCorchete while a	7	
goto(21, identifi	{S -> identificador.asignacion E puntoYcoma; S	9	
goto(21, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	
goto(21, input)	{S -> input.abrirParentesis identificador cerra	11	
goto(21, return)	{S -> return.X puntoYcoma}	12	
goto(22, G)	{F -> I J G.}	45	{F -> I J G.}
goto(22, abrirCor	{G -> abrirCorchete.C cerrarCorchete}	46	{G -> abrirCorchete.C cerrarCorchete; C -> .B C; C -> .; B -> .if abrirParentesis E cerrarParentesis S; B -> .let T identificador puntoYcoma; B -> .S; B -> .do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma}
goto(23, A)	{J -> abrirParentesis A.cerrarParentesis}	47	{J -> abrirParentesis A.cerrarParentesis}
goto(23, T)	{A -> T.identificador K}	48	{A -> T.identificador K}
goto(23, number)	{T -> number.}	18	
goto(23, boolean)	{T -> boolean.}	19	
goto(23, string)	{T -> string.}	20	
goto(24, E)	{S -> identificador asignacion E.puntoYcoma; E	49	{S -> identificador asignacion E.puntoYcoma; E -> E.opLogico2 R}
goto(24, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(24, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(24, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(24, W)	{V -> W.}	34	
goto(24, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(24, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(24, cteEnter	{W -> cteEntera.}	37	
goto(24, cadena)	{W -> cadena.}	38	
goto(25, L)	{S -> identificador abrirParentesis L.cerrarPar	50	{S -> identificador abrirParentesis L.cerrarParentesis puntoYcoma}
goto(25, E)	{L -> E.Q; E -> E.opLogico2 R}	51	{L -> E.Q; E -> E.opLogico2 R; Q -> .coma E Q; Q -> .}
goto(25, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(25, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(25, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(25, W)	{V -> W.}	34	
goto(25, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(25, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(25, cteEnter	{W -> cteEntera.}	37	
goto(25, cadena)	{W -> cadena.}	38	
goto(26, L)	{S -> identificador restaAsignacion L.puntoYcom	52	{S -> identificador restaAsignacion L.puntoYcoma}
goto(26, E)	{L -> E.Q; E -> E.opLogico2 R}	51	
goto(26, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(26, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(26, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(26, W)	{V -> W.}	34	
goto(26, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(26, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(26, cteEnter	{W -> cteEntera.}	37	
goto(26, cadena)	{W -> cadena.}	38	
goto(27, E)	{S -> alert abrirParentesis E.cerrarParentesis	53	{S -> alert abrirParentesis E.cerrarParentesis puntoYcoma; E -> E.opLogico2 R}
goto(27, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(27, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(27, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(27, W)	{V -> W.}	34	
goto(27, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(27, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(27, cteEnter	{W -> cteEntera.}	37	
goto(27, cadena)	{W -> cadena.}	38	
goto(28, identifi	{S -> input abrirParentesis identificador.cerra	54	{S -> input abrirParentesis identificador.cerrarParentesis puntoYcoma}
goto(29, puntoYco	{S -> return X puntoYcoma.}	55	{S -> return X puntoYcoma.}
goto(30, opLogico	{E -> E opLogico2.R}	56	{E -> E opLogico2.R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(31, opLogico	{R -> R opLogico1.U}	57	{R -> R opLogico1.U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(32, opRelaci	{U -> U opRelacional1.V}	58	{U -> U opRelacional1.V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(32, opRelaci	{U -> U opRelacional2.V}	59	{U -> U opRelacional2.V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(33, opAritme	{V -> V opAritmetico1.W}	60	{V -> V opAritmetico1.W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(33, opAritme	{V -> V opAritmetico2.W}	61	{V -> V opAritmetico2.W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}

goto(35, abrirPar	{W -> identificador abrirParentesis.L cerrarPar	62	{W -> identificador abrirParentesis.L cerrarParentesis; L -> .E Q; L -> .; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(36, E)	{W -> abrirParentesis E.cerrarParentesis; E ->	63	{W -> abrirParentesis E.cerrarParentesis; E -> E.opLogico2 R}
goto(36, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(36, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(36, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(36, W)	{V -> W.}	34	
goto(36, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(36, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(36, cteEnter	{W -> cteEntera.}	37	
goto(36, cadena)	{W -> cadena.}	38	
goto(39, identifi	{I -> function H identificador.}	64	{I -> function H identificador.}
goto(41, cerrarPa	{B -> if abrirParentesis E cerrarParentesis.S}	65	{B -> if abrirParentesis E cerrarParentesis.S; S -> .identificador asignacion E puntoYcoma; S -> .identificador abrirParentesis L cerrarParentesis puntoYcoma; S -> .alert abrirParentesis E cerrarParentesis puntoYcoma; S -> .input abrirParentesis identificador cerrarParentesis puntoYcoma; S -> .return X puntoYcoma; S -> .identificador restaAsignacion L puntoYcoma}
goto(41, opLogico	{E -> E opLogico2.R}	56	
goto(42, puntoYco	{B -> let T identificador puntoYcoma.}	66	{B -> let T identificador puntoYcoma.}
goto(43, cerrarCo	{B -> do abrirCorchete C cerrarCorchete.while a	67	{B -> do abrirCorchete C cerrarCorchete.while abrirParentesis E cerrarParentesis puntoYcoma}
goto(44, C)	{C -> B C.}	68	{C -> B C.}
goto(44, B)	{C -> B.C}	44	
goto(44, if)	{B -> if.abrirParentesis E cerrarParentesis S}	4	
goto(44, let)	{B -> let.T identificador puntoYcoma}	5	
goto(44, S)	{B -> S.}	6	
goto(44, do)	{B -> do.abrirCorchete C cerrarCorchete while a	7	
goto(44, identifi	{S -> identificador.asignacion E puntoYcoma; S	9	
goto(44, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	
goto(44, input)	{S -> input.abrirParentesis identificador cerra	11	
goto(44, return)	{S -> return.X puntoYcoma}	12	
goto(46, C)	{G -> abrirCorchete C.cerrarCorchete}	69	{G -> abrirCorchete C.cerrarCorchete}
goto(46, B)	{C -> B.C}	44	
goto(46, if)	{B -> if.abrirParentesis E cerrarParentesis S}	4	
goto(46, let)	{B -> let.T identificador puntoYcoma}	5	
goto(46, S)	{B -> S.}	6	
goto(46, do)	{B -> do.abrirCorchete C cerrarCorchete while a	7	
goto(46, identifi	{S -> identificador.asignacion E puntoYcoma; S	9	
goto(46, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	
goto(46, input)	{S -> input.abrirParentesis identificador cerra	11	
goto(46, return)	{S -> return.X puntoYcoma}	12	
goto(47, cerrarPa	{J -> abrirParentesis A cerrarParentesis.}	70	{J -> abrirParentesis A cerrarParentesis.}
goto(48, identifi	{A -> T identificador.K}	71	{A -> T identificador.K; K -> .coma T identificador K; K -> .}
goto(49, puntoYco	{S -> identificador asignacion E puntoYcoma.}	72	{S -> identificador asignacion E puntoYcoma.}
goto(49, opLogico	{E -> E opLogico2.R}	56	
goto(50, cerrarPa	{S -> identificador abrirParentesis L cerrarPar	73	{S -> identificador abrirParentesis L cerrarParentesis.puntoYcoma}
goto(51, Q)	{L -> E Q.}	74	{L -> E Q.}
goto(51, opLogico	{E -> E opLogico2.R}	56	
goto(51, coma)	{Q -> coma.E Q}	75	{Q -> coma.E Q; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(52, puntoYco	{S -> identificador restaAsignacion L puntoYcom	76	{S -> identificador restaAsignacion L puntoYcoma.}
goto(53, cerrarPa	{S -> alert abrirParentesis E cerrarParentesis.	77	{S -> alert abrirParentesis E cerrarParentesis.puntoYcoma}
goto(53, opLogico	{E -> E opLogico2.R}	56	
goto(54, cerrarPa	{S -> input abrirParentesis identificador cerra	78	{S -> input abrirParentesis identificador cerrarParentesis.puntoYcoma}
goto(56, R)	{E -> E opLogico2 R.; R -> R.opLogico1 U}	79	{E -> E opLogico2 R.; R -> R.opLogico1 U}
goto(56, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(56, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(56, W)	{V -> W.}	34	
goto(56, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(56, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(56, cteEnter	{W -> cteEntera.}	37	
goto(56, cadena)	{W -> cadena.}	38	
goto(57, U)	{R -> R opLogico1 U.; U -> U.opRelacional1 V; U	80	{R -> R opLogico1 U.; U -> U.opRelacional1 V; U -> U.opRelacional2 V}
goto(57, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(57, W)	{V -> W.}	34	
goto(57, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(57, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(57, cteEnter	{W -> cteEntera.}	37	
goto(57, cadena)	{W -> cadena.}	38	
goto(58, V)	{U -> U opRelacional1 V.; V -> V.opAritmetico1	81	{U -> U opRelacional1 V.; V -> V.opAritmetico1 W; V -> V.opAritmetico2 W}
goto(58, W)	{V -> W.}	34	
goto(58, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(58, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(58, cteEnter	{W -> cteEntera.}	37	
goto(58, cadena)	{W -> cadena.}	38	
goto(59, V)	{U -> U opRelacional2 V.; V -> V.opAritmetico1	82	{U -> U opRelacional2 V.; V -> V.opAritmetico1 W; V -> V.opAritmetico2 W}
goto(59, W)	{V -> W.}	34	
goto(59, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(59, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(59, cteEnter	{W -> cteEntera.}	37	
goto(59, cadena)	{W -> cadena.}	38	
goto(60, W)	{V -> V opAritmetico1 W.}	83	{V -> V opAritmetico1 W.}
goto(60, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(60, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(60, cteEnter	{W -> cteEntera.}	37	
goto(60, cadena)	{W -> cadena.}	38	
goto(61, W)	{V -> V opAritmetico2 W.}	84	{V -> V opAritmetico2 W.}
goto(61, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(61, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(61, cteEnter	{W -> cteEntera.}	37	
goto(61, cadena)	{W -> cadena.}	38	
goto(62, L)	{W -> identificador abrirParentesis L.cerrarPar	85	{W -> identificador abrirParentesis L.cerrarParentesis}
goto(62, E)	{L -> E.Q; E -> E.opLogico2 R}	51	
goto(62, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(62, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(62, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(62, W)	{V -> W.}	34	
goto(62, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(62, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(62, cteEnter	{W -> cteEntera.}	37	
goto(62, cadena)	{W -> cadena.}	38	
goto(63, cerrarPa	{W -> abrirParentesis E cerrarParentesis.}	86	{W -> abrirParentesis E cerrarParentesis.}
goto(63, opLogico	{E -> E opLogico2.R}	56	
goto(65, S)	{B -> if abrirParentesis E cerrarParentesis S.}	87	{B -> if abrirParentesis E cerrarParentesis S.}
goto(65, identifi	{S -> identificador.asignacion E puntoYcoma; S	9	
goto(65, alert)	{S -> alert.abrirParentesis E cerrarParentesis	10	
goto(65, input)	{S -> input.abrirParentesis identificador cerra	11	
goto(65, return)	{S -> return.X puntoYcoma}	12	
goto(67, while)	{B -> do abrirCorchete C cerrarCorchete while.a	88	{B -> do abrirCorchete C cerrarCorchete while.abrirParentesis E cerrarParentesis puntoYcoma}
goto(69, cerrarCo	{G -> abrirCorchete C cerrarCorchete.}	89	{G -> abrirCorchete C cerrarCorchete.}
goto(71, K)	{A -> T identificador K.}	90	{A -> T identificador K.}
goto(71, coma)	{K -> coma.T identificador K}	91	{K -> coma.T identificador K; T -> .number; T -> .boolean; T -> .string}
goto(73, puntoYco	{S -> identificador abrirParentesis L cerrarPar	92	{S -> identificador abrirParentesis L cerrarParentesis puntoYcoma.}
goto(75, E)	{Q -> coma E.Q; E -> E.opLogico2 R}	93	{Q -> coma E.Q; E -> E.opLogico2 R; Q -> .coma E Q; Q -> .}
goto(75, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(75, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(75, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(75, W)	{V -> W.}	34	
goto(75, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(75, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(75, cteEnter	{W -> cteEntera.}	37	
goto(75, cadena)	{W -> cadena.}	38	
goto(77, puntoYco	{S -> alert abrirParentesis E cerrarParentesis	94	{S -> alert abrirParentesis E cerrarParentesis puntoYcoma.}
goto(78, puntoYco	{S -> input abrirParentesis identificador cerra	95	{S -> input abrirParentesis identificador cerrarParentesis puntoYcoma.}
goto(79, opLogico	{R -> R opLogico1.U}	57	
goto(80, opRelaci	{U -> U opRelacional1.V}	58	
goto(80, opRelaci	{U -> U opRelacional2.V}	59	
goto(81, opAritme	{V -> V opAritmetico1.W}	60	
goto(81, opAritme	{V -> V opAritmetico2.W}	61	
goto(82, opAritme	{V -> V opAritmetico1.W}	60	
goto(82, opAritme	{V -> V opAritmetico2.W}	61	
goto(85, cerrarPa	{W -> identificador abrirParentesis L cerrarPar	96	{W -> identificador abrirParentesis L cerrarParentesis.}
goto(88, abrirPar	{B -> do abrirCorchete C cerrarCorchete while a	97	{B -> do abrirCorchete C cerrarCorchete while abrirParentesis.E cerrarParentesis puntoYcoma; E -> .E opLogico2 R; E -> .R; R -> .R opLogico1 U; R -> .U; U -> .U opRelacional1 V; U -> .U opRelacional2 V; U -> .V; V -> .V opAritmetico1 W; V -> .V opAritmetico2 W; V -> .W; W -> .identificador; W -> .abrirParentesis E cerrarParentesis; W -> .identificador abrirParentesis L cerrarParentesis; W -> .cteEntera; W -> .cadena}
goto(91, T)	{K -> coma T.identificador K}	98	{K -> coma T.identificador K}
goto(91, number)	{T -> number.}	18	
goto(91, boolean)	{T -> boolean.}	19	
goto(91, string)	{T -> string.}	20	
goto(93, Q)	{Q -> coma E Q.}	99	{Q -> coma E Q.}
goto(93, opLogico	{E -> E opLogico2.R}	56	
goto(93, coma)	{Q -> coma.E Q}	75	
goto(97, E)	{B -> do abrirCorchete C cerrarCorchete while a	100	{B -> do abrirCorchete C cerrarCorchete while abrirParentesis E.cerrarParentesis puntoYcoma; E -> E.opLogico2 R}
goto(97, R)	{E -> R.; R -> R.opLogico1 U}	31	
goto(97, U)	{R -> U.; U -> U.opRelacional1 V; U -> U.opRela	32	
goto(97, V)	{U -> V.; V -> V.opAritmetico1 W; V -> V.opArit	33	
goto(97, W)	{V -> W.}	34	
goto(97, identifi	{W -> identificador.; W -> identificador.abrirP	35	
goto(97, abrirPar	{W -> abrirParentesis.E cerrarParentesis}	36	
goto(97, cteEnter	{W -> cteEntera.}	37	
goto(97, cadena)	{W -> cadena.}	38	
goto(98, identifi	{K -> coma T identificador.K}	101	{K -> coma T identificador.K; K -> .coma T identificador K; K -> .}
goto(100, cerrarP	{B -> do abrirCorchete C cerrarCorchete while a	102	{B -> do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis.puntoYcoma}
goto(100, opLogic	{E -> E opLogico2.R}	56	
goto(101, K)	{K -> coma T identificador K.}	103	{K -> coma T identificador K.}
goto(101, coma)	{K -> coma.T identificador K}	91	
goto(102, puntoYc	{B -> do abrirCorchete C cerrarCorchete while a	104	{B -> do abrirCorchete C cerrarCorchete while abrirParentesis E cerrarParentesis puntoYcoma.}

	function	entificad	brirParentesi	errarParentesi	abrirCorchete	cerrarCorchete	coma	asignacion	puntoYcoma	alert	input	return	taAsignac	if	let	do	while	number	boolean	string	opLogico2	opLogico1	Relaciona	Relaciona	Aritmetic	Aritmetic	cteEntera	cadena	$
0	s13	s9								s10	s11	s12		s4	s5	s7													r3
1																													acc
2	s13	s9								s10	s11	s12		s4	s5	s7													r1
3	s13	s9								s10	s11	s12		s4	s5	s7													r1
4			s16																										
5																		s18	s19	s20									
6	r24	r24				r24				r24	r24	r24		r24	r24	r24													r24
7					s21																								
8			s23																										
9			s25					s24					s26																
10			s27																										
11			s28																										
12		s35	s36						r49																		s37	s38	
13		r9																s18	s19	s20									
14																													r1
15																													r2
16		s35	s36																								s37	s38	
17		s42																											
18		r26																											
19		r27																											
20		r28																											
21		s9				r15				s10	s11	s12		s4	s5	s7													
22					s46																								
23				r11														s18	s19	s20									
24		s35	s36																								s37	s38	
25		s35	s36	r45					r45																		s37	s38	
26		s35	s36	r45					r45																		s37	s38	
27		s35	s36																								s37	s38	
28		s54																											
29									s55																				
30									r48												s56								
31				r30			r30		r30												r30	s57							
32				r32			r32		r32												r32	r32	s58	s59					
33				r35			r35		r35												r35	r35	r35	r35	s60	s61			
34				r38			r38		r38												r38	r38	r38	r38	r38	r38			
35			s62	r39			r39		r39												r39	r39	r39	r39	r39	r39			
36		s35	s36																								s37	s38	
37				r42			r42		r42												r42	r42	r42	r42	r42	r42			
38				r43			r43		r43												r43	r43	r43	r43	r43	r43			
39		s64																											
40		r8																											
41				s65																	s56								
42									s66																				
43						s67																							
44		s9				r15				s10	s11	s12		s4	s5	s7													
45	r4	r4								r4	r4	r4		r4	r4	r4													r4
46		s9				r15				s10	s11	s12		s4	s5	s7													
47				s70																									
48		s71																											
49									s72												s56								
50				s73																									
51				r47			s75		r47												s56								
52									s76																				
53				s77																	s56								
54				s78																									
55	r20	r20				r20				r20	r20	r20		r20	r20	r20													r20
56		s35	s36																								s37	s38	
57		s35	s36																								s37	s38	
58		s35	s36																								s37	s38	
59		s35	s36																								s37	s38	
60		s35	s36																								s37	s38	
61		s35	s36																								s37	s38	
62		s35	s36	r45					r45																		s37	s38	
63				s86																	s56								
64			r5																										
65		s9								s10	s11	s12																	
66	r23	r23				r23				r23	r23	r23		r23	r23	r23													r23
67																	s88												
68						r14																							
69						s89																							
70					r6																								
71				r13			s91																						
72	r16	r16				r16				r16	r16	r16		r16	r16	r16													r16
73									s92																				
74				r44					r44																				
75		s35	s36																								s37	s38	
76	r21	r21				r21				r21	r21	r21		r21	r21	r21													r21
77									s94																				
78									s95																				
79				r29			r29		r29												r29	s57							
80				r31			r31		r31												r31	r31	s58	s59					
81				r33			r33		r33												r33	r33	r33	r33	s60	s61			
82				r34			r34		r34												r34	r34	r34	r34	s60	s61			
83				r36			r36		r36												r36	r36	r36	r36	r36	r36			
84				r37			r37		r37												r37	r37	r37	r37	r37	r37			
85				s96																									
86				r40			r40		r40												r40	r40	r40	r40	r40	r40			
87	r22	r22				r22				r22	r22	r22		r22	r22	r22													r22
88			s97																										
89	r7	r7								r7	r7	r7		r7	r7	r7													r7
90				r10																									
91																		s18	s19	s20									
92	r17	r17				r17				r17	r17	r17		r17	r17	r17													r17
93				r47			s75		r47												s56								
94	r18	r18				r18				r18	r18	r18		r18	r18	r18													r18
95	r19	r19				r19				r19	r19	r19		r19	r19	r19													r19
96				r41			r41		r41												r41	r41	r41	r41	r41	r41			
97		s35	s36																								s37	s38	
98		s101																											
99				r46					r46																				
100				s102																	s56								
101				r13			s91																						
102									s104																				
103				r12																									
104	r25	r25				r25				r25	r25	r25		r25	r25	r25													r25

Z	P	F	I	J	G	H	A	K	C	S	B	T	E	R	U	V	W	L	Q	X
	1	3	8							6	2									
	14	3	8							6	2									
	15	3	8							6	2									
												17								
				22																
													30	31	32	33	34			29
						39						40								
													41	31	32	33	34			
									43	6	44									
					45															
							47					48								
													49	31	32	33	34			
													51	31	32	33	34	50		
													51	31	32	33	34	52		
													53	31	32	33	34			
									68	6	44									
									69	6	44									
																			74	
														79	32	33	34			
															80	33	34			
																81	34			
																82	34			
																	83			
																	84			
													51	31	32	33	34	85		
										87										
								90												
													93	31	32	33	34			
												98								
																			99	
													100	31	32	33	34			
								103												

Justificación de LR(1)

Debido a que en la tabla acción solo existe una opción por celda, ya sea desplazar(S), reducir(R) o aceptar (ACC), podemos asegurar que nuestra gramática es LR(1).

Además, se puede justificar, a partir del análisis de la colección canónica

Calculo de tablas

Debido a la inmensa cantidad de iteraciones del algoritmo para el desarrollo de tablas, hemos optado por usar una herramienta online:
http://jsmachines.sourceforge.net/machines/slr.html

La gramática para usarla con el programa:

```
Z -> P

P -> B P
P -> F P
P -> ''

F -> I J G
I -> function H identificador
J -> abrirParentesis A cerrarParentesis
G -> abrirCorchete C cerrarCorchete
H -> T
H -> ''
A -> T identificador K
A -> ''
K -> coma T identificador K
K -> ''
C -> B C
C -> ''

S -> identificador asignacion E puntoYcoma
S -> identificador abrirParentesis L cerrarParentesis puntoYcoma
S -> alert abrirParentesis E cerrarParentesis puntoYcoma
S -> input abrirParentesis identificador cerrarParentesis puntoYcoma
S -> return X puntoYcoma
S -> identificador restaAsignacion L puntoYcoma

B -> if abrirParentesis E cerrarParentesis S
B -> let T identificador puntoYcoma
B -> S
B -> do abrirCorchete C cerrarCorchete while abrirParentesis E
cerrarParentesis puntoYcoma

T -> number
T -> boolean
T -> string

E -> E opLogico2 R
E -> R
R -> R opLogico1 U
R -> U
U -> U opRelacional1 V
U -> U opRelacional2 V
U -> V
V -> V opAritmetico1 W
V -> V opAritmetico2 W
```

```
V -> W
W -> identificador
W -> abrirParentesis E cerrarParentesis
W -> identificador abrirParentesis L cerrarParentesis
W -> cteEntera
W -> cadena

L -> E Q
L -> ''
Q -> coma E Q
Q -> ''
X -> E
X -> ''
```

Únicamente se ha usado para obtener las tablas. Para la extracción de estas tablas, se ha procedido a exportarlas a un excel, a partir del cual, se ha desarrollado un Script en Python que genere el código necesario para importarlo a Java

Errores

0	Carácter No Valido
1	Lexema ya en tabla de símbolos
-1	Error de Programador
2	Integer Out Of Bounds
3	
4	Variable Name Out Of Bounds
5	Error Sintáctico

Observaciones al corrector

Incluimos el código usado para Vast

```
Terminales = { function identificador  cerrarParentesis
abrirParentesis   cerrarCorchete  abrirCorchete coma asignacion
puntoYcoma alert input return restaAsignacion if let do while number
boolean string opLogico2 opLogico1 opRelacional1 opRelacional2
opAritmetico1 opAritmetico2 cteEntera cadena }

NoTerminales = { Z P F I J G H A K C S B T E R U V W L Q X }

Axioma = Z
Producciones = {
    Z -> P

    P -> B P
    P -> F P
    P -> lambda

    F -> I J G
    I -> function H identificador
    J -> abrirParentesis A cerrarParentesis
    G -> abrirCorchete C cerrarCorchete
    H -> T
    H -> lambda
    A -> T identificador K
    A -> lambda
    K -> coma T identificador K
    K -> lambda
    C -> B C
    C -> lambda

    S -> identificador asignacion E puntoYcoma
    S -> identificador abrirParentesis L cerrarParentesis puntoYcoma
    S -> alert abrirParentesis E cerrarParentesis puntoYcoma
    S -> input abrirParentesis identificador cerrarParentesis
puntoYcoma
    S -> return X puntoYcoma
    S -> identificador restaAsignacion L puntoYcoma

    B -> if abrirParentesis E cerrarParentesis S
    B -> let T identificador puntoYcoma
    B -> S
    B -> do abrirCorchete C cerrarCorchete while abrirParentesis E
cerrarParentesis puntoYcoma

    T -> number
    T -> boolean
    T -> string

    E -> E opLogico2 R
    E -> R
    R -> R opLogico1 U
    R -> U
    U -> U opRelacional1 V
    U -> U opRelacional2 V
    U -> V
    V -> V opAritmetico1 W
    V -> V opAritmetico2 W
    V -> W
```

```
    W -> identificador
    W -> abrirParentesis E cerrarParentesis
    W -> identificador abrirParentesis L cerrarParentesis
    W -> cteEntera
    W -> cadena

    L -> E Q
    L -> lambda
    Q -> coma E Q
    Q -> lambda
    X -> E
    X -> lambda
}
```

Anexo

Prueba 1

```
let number n1;
let boolean l1;
let string cad;
let number n2;
let boolean l2;

alert ("PdL");
input (esto_es_un_nombre_de_variable_global_de_tipo_entero);
input (n1);
l1 = l2;
if (l1&& l2) cad = "hello";
n2 = n1 - 378;

alert(     33
           -
           n1
           -
           n2);
function boolean ff(boolean ss)
{
      l2 =  l1;
      if (l2) l1 = ff (ss);
      varglobal = 8888;
      return (ss);
}
if (ff(l1)) alert (varglobal);
```

Tokens Autogenerados

```
<let, >
<number, >
<identificador, 0>
<puntoYcoma, >
<let, >
<boolean, >
<identificador, 1>
<puntoYcoma, >
<let, >
<string, >
<identificador, 2>
<puntoYcoma, >
<let, >
<number, >
<identificador, 3>
<puntoYcoma, >
<let, >
<boolean, >
<identificador, 4>
<puntoYcoma, >
<alert, >
<abrirParentesis, >
<cadena, "PdL">
<cerrarParentesis, >
<puntoYcoma, >
<input, >
<abrirParentesis, >
```

<identificador, 5>
<cerrarParentesis, >
<puntoYcoma, >
<input, >
<abrirParentesis, >
<identificador, 0>
<cerrarParentesis, >
<puntoYcoma, >
<identificador, 1>
<asignacion, >
<identificador, 4>
<puntoYcoma, >
<if, >
<abrirParentesis, >
<identificador, 1>
<opLogico, 1>
<identificador, 4>
<cerrarParentesis, >
<identificador, 2>
<asignacion, >
<cadena, "hello">
<puntoYcoma, >
<identificador, 3>
<asignacion, >
<identificador, 0>
<opAritmetico, 2>
<cteEntera, 378>
<puntoYcoma, >
<alert, >
<abrirParentesis, >
<cteEntera, 33>
<opAritmetico, 2>
<identificador, 0>
<opAritmetico, 2>
<identificador, 3>
<cerrarParentesis, >
<puntoYcoma, >
<function, >
<boolean, >
<identificador, 6>
<abrirParentesis, >
<boolean, >
<identificador, 7>
<cerrarParentesis, >
<abrirCorchete, >
<identificador, 4>
<asignacion, >
<identificador, 1>
<puntoYcoma, >
<if, >
<abrirParentesis, >
<identificador, 4>
<cerrarParentesis, >
<identificador, 1>
<asignacion, >
<identificador, 6>
<abrirParentesis, >
<identificador, 7>

```
<cerrarParentesis, >
<puntoYcoma, >
<identificador, 8>
<asignacion, >
<cteEntera, 8888>
<puntoYcoma, >
<return, >
<abrirParentesis, >
<identificador, 7>
<cerrarParentesis, >
<puntoYcoma, >
<cerrarCorchete, >
<if, >
<abrirParentesis, >
<identificador, 6>
<abrirParentesis, >
<identificador, 1>
<cerrarParentesis, >
<cerrarParentesis, >
<alert, >
<abrirParentesis, >
<identificador, 8>
<cerrarParentesis, >
<puntoYcoma, >
<EOF, >
```

Tabla Simbolos

```
CONTENIDO DE LA TABLA TSMAIN #1 :

      * LEXEMA : 'n1'
------------------------------------------------------------
      * LEXEMA : 'l1'
------------------------------------------------------------
      * LEXEMA : 'cad'
------------------------------------------------------------
      * LEXEMA : 'n2'
------------------------------------------------------------
      * LEXEMA : 'l2'
------------------------------------------------------------
      * LEXEMA :
'esto_es_un_nombre_de_variable_global_de_tipo_entero'
------------------------------------------------------------
      * LEXEMA : 'ff'
------------------------------------------------------------
      * LEXEMA : 'ss'
------------------------------------------------------------
      * LEXEMA : 'varglobal'
------------------------------------------------------------
```

Parse y Arbol Sintactico

Parse

Ascendente 27 24 28 24 29 24 27 24 28 24 44 39 36 33 31 19 25 20 25 20 25 40 39 36 33 31 17 25 40 39 36 33 40 39 36 32 31 44 39 36 33 31 17 23 40 39 43 38 36 33 31 17 25 43 39 40 38 40 38 36 33 31 19 25 28 9 6 28 14 11 7 40 39 36 33 31 17 25 40 39 36 33 31 40 39 36 33 31 48 45 42 39 36 33 31 17 23 43 39 36 33 31 17 25 40 39 36 33 31 41 39 36 33 31 49 21 25 16 15 15 15 15 8 5 40 39 36 33 31 48 45 42 39 36 33 31 40 39 36 33 31 19 23 4 2 3 2 2 2 2 2 2 2 2 2 2 2 2 1

Árbol resultado de:

Gramática: D:\OneDrive - Universidad Politécnica de Madrid\Universidad\3º\Procesadores de Lenguajes\Practica\ProcesadoresLenguajes\Pruebas\ASintactico-Tests\Herramienta Vast\aSintax\GramaticaVast.txt

Parse: D:\OneDrive - Universidad Politécnica de Madrid\Universidad\3º\Procesadores de Lenguajes\Practica\ProcesadoresLenguajes\docs\Analizador Sintactico\Pruebas realizadas\Prueba1\Parse.txt

- Z (1)
  - P (2)
    - B (24)
      - let
      - T (27)
        - number
      - identificador
      - puntoYcoma
    - P (2)
      - B (24)
        - let
        - T (28)
          - boolean
        - identificador
        - puntoYcoma
      - P (2)
        - B (24)
          - let
          - T (29)
            - string
          - identificador
          - puntoYcoma
        - P (2)
          - B (24)
            - let
            - T (27)
              - number
            - identificador
            - puntoYcoma
          - P (2)
            - B (24)
              - let
              - T (28)
                - boolean
              - identificador
              - puntoYcoma
            - P (2)
              - B (25)
                - S (19)
                  - alert
                  - abrirParentesis
                  - E (31)
                    - R (33)
                      - U (36)
                        - V (39)
                          - W (44)

cadena

cerrarParentesis

puntoYcoma

P (2)

B (25)

S (20)

input

abrirParentesis

identificador

cerrarParentesis

puntoYcoma

P (2)

B (25)

S (20)

input

abrirParentesis

identificador

cerrarParentesis

puntoYcoma

P (2)

B (25)

S (17)

identificador

asignacion

E (31)

R (33)

U (36)

V (39)

W (40)

identificador

puntoYcoma

P (2)

B (23)

if

abrirParentesis

E (31)

R (32)

R (33)

U (36)

V (39)

W (40)

identificador

opLogico1

U (36)

V (39)

W (40)

identificador

cerrarParentesis

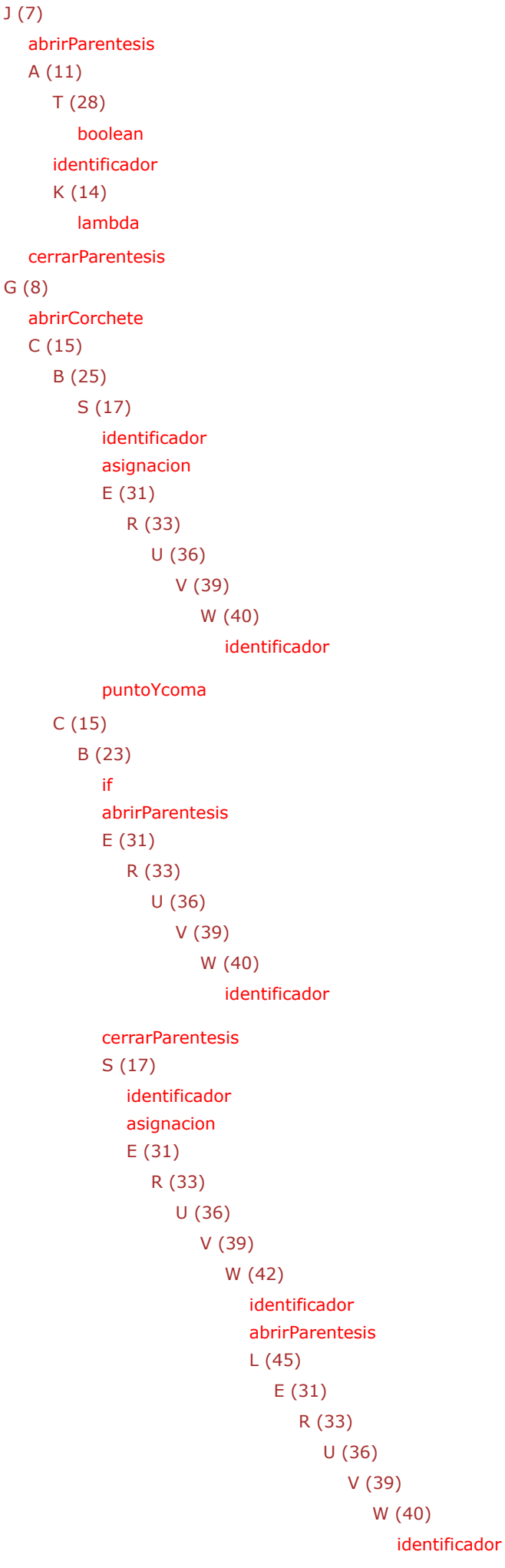
S (17)

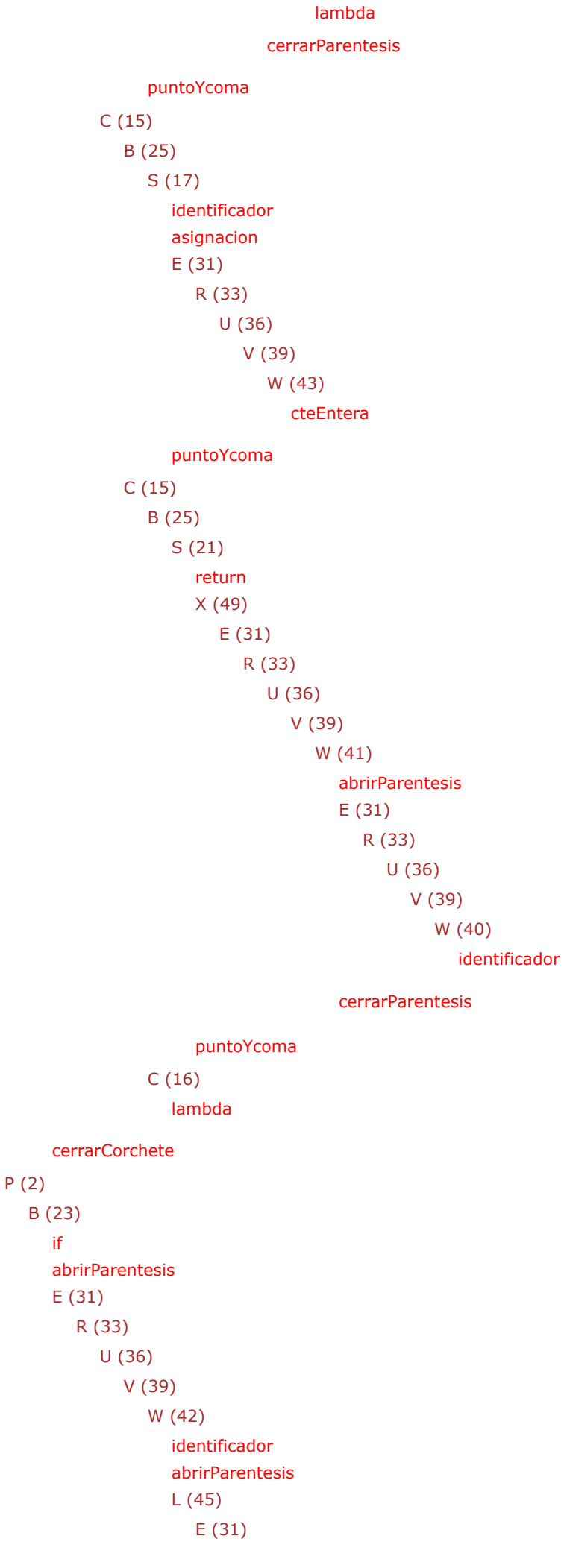
identificador

asignacion

E (31)

R (33)
U (36)
V (39)
W (44)
cadena
puntoYcoma
P (2)
B (25)
S (17)
identificador
asignacion
E (31)
R (33)
U (36)
V (38)
V (39)
W (40)
identificador
opAritmetico2
W (43)
cteEntera
puntoYcoma
P (2)
B (25)
S (19)
alert
abrirParentesis
E (31)
R (33)
U (36)
V (38)
V (38)
V (39)
W (43)
cteEntera
opAritmetico2
W (40)
identificador
opAritmetico2
W (40)
identificador
cerrarParentesis
puntoYcoma
P (3)
F (5)
I (6)
function
H (9)
T (28)
boolean
identificador


J (7)
abrirParentesis
A (11)
T (28)
boolean
identificador
K (14)
lambda
cerrarParentesis
G (8)
abrirCorchete
C (15)
B (25)
S (17)
identificador
asignacion
E (31)
R (33)
U (36)
V (39)
W (40)
identificador
puntoYcoma
C (15)
B (23)
if
abrirParentesis
E (31)
R (33)
U (36)
V (39)
W (40)
identificador
cerrarParentesis
S (17)
identificador
asignacion
E (31)
R (33)
U (36)
V (39)
W (42)
identificador
abrirParentesis
L (45)
E (31)
R (33)
U (36)
V (39)
W (40)
identificador
Q (48)


lambda
cerrarParentesis
puntoYcoma
C (15)
B (25)
S (17)
identificador
asignacion
E (31)
R (33)
U (36)
V (39)
W (43)
cteEntera
puntoYcoma
C (15)
B (25)
S (21)
return
X (49)
E (31)
R (33)
U (36)
V (39)
W (41)
abrirParentesis
E (31)
R (33)
U (36)
V (39)
W (40)
identificador
cerrarParentesis
puntoYcoma
C (16)
lambda
cerrarCorchete
P (2)
B (23)
if
abrirParentesis
E (31)
R (33)
U (36)
V (39)
W (42)
identificador
abrirParentesis
L (45)
E (31)

- R (33)
  - U (36)
    - V (39)
      - W (40)
        - identificador
- Q (48)
  - lambda
- cerrarParentesis

- cerrarParentesis
- S (19)
  - alert
  - abrirParentesis
  - E (31)
    - R (33)
      - U (36)
        - V (39)
          - W (40)
            - identificador
  - cerrarParentesis
  - puntoYcoma

P (4)
- lambda

Prueba 2

```
let string texto;
function print (string msg)
{
      alert (msg);
}
function pideTexto ()
{
      alert ("Introduce un texto");
      input (texto);
}
pideTexto();
let string textoAux;
textoAux = texto;
print (textoAux);
```

Tokens

```
<let, >
<string, >
<identificador, 0>
<puntoYcoma, >
<function, >
<identificador, 1>
<abrirParentesis, >
<string, >
<identificador, 2>
<cerrarParentesis, >
<abrirCorchete, >
<alert, >
<abrirParentesis, >
<identificador, 2>
<cerrarParentesis, >
<puntoYcoma, >
<cerrarCorchete, >
<function, >
<identificador, 3>
<abrirParentesis, >
<cerrarParentesis, >
<abrirCorchete, >
<alert, >
<abrirParentesis, >
<cadena, "Introduce un texto">
<cerrarParentesis, >
<puntoYcoma, >
<input, >
<abrirParentesis, >
<identificador, 0>
<cerrarParentesis, >
<puntoYcoma, >
<cerrarCorchete, >
<identificador, 3>
<abrirParentesis, >
<cerrarParentesis, >
<puntoYcoma, >
<let, >
<string, >
<identificador, 4>
<puntoYcoma, >
```

```
<identificador, 4>
<asignacion, >
<identificador, 0>
<puntoYcoma, >
<identificador, 1>
<abrirParentesis, >
<identificador, 4>
<cerrarParentesis, >
<puntoYcoma, >
<EOF, >
```

Tabla símbolos

```
CONTENIDO DE LA TABLA TSMAIN #1 :

      * LEXEMA : 'texto'
-------------------------------------------------------------
      * LEXEMA : 'print'
-------------------------------------------------------------
      * LEXEMA : 'msg'
-------------------------------------------------------------
      * LEXEMA : 'pideTexto'
-------------------------------------------------------------
      * LEXEMA : 'textoAux'
-------------------------------------------------------------
```

Parse y Arbol Sintactico

Parse

Ascendente 29 24 10 6 29 14 11 7 40 39 36 33 31 19 25 16 15 8 5 10 6 12 7 44 39 36 33 31 19 25 20 25 16 15 15 8 5 46 18 25 29 24 40 39 36 33 31 17 25 40 39 36 33 31 48 45 18 25 4 2 2 2 2 3 3 2 1

Arbol Sintactico

Árbol resultado de:

Gramática: D:\OneDrive - Universidad Politécnica de Madrid\Universidad\3º\Procesadores de Lenguajes\Practica\ProcesadoresLenguajes\Pruebas\ASintactico-Tests\Herramienta Vast\aSintax\GramaticaVast.txt

Parse: D:\OneDrive - Universidad Politécnica de Madrid\Universidad\3º\Procesadores de Lenguajes\Practica\ProcesadoresLenguajes\docs\Analizador Sintactico\Pruebas realizadas\Prueba2\Parse.txt

- Z (1)
  - P (2)
    - B (24)
      - let
      - T (29)
        - string
      - identificador
      - puntoYcoma
    - P (3)
      - F (5)
        - I (6)
          - function
          - H (10)
            - lambda
          - identificador
        - J (7)
          - abrirParentesis
          - A (11)
            - T (29)
              - string
            - identificador
            - K (14)
              - lambda
          - cerrarParentesis
        - G (8)
          - abrirCorchete
          - C (15)
            - B (25)
              - S (19)
                - alert
                - abrirParentesis
                - E (31)
                  - R (33)
                    - U (36)
                      - V (39)
                        - W (40)
                          - identificador
                - cerrarParentesis
                - puntoYcoma
            - C (16)
              - lambda
          - cerrarCorchete
      - P (3)
        - F (5)
          - I (6)

function
H (10)
lambda
identificador
J (7)
abrirParentesis
A (12)
lambda
cerrarParentesis
G (8)
abrirCorchete
C (15)
B (25)
S (19)
alert
abrirParentesis
E (31)
R (33)
U (36)
V (39)
W (44)
cadena
cerrarParentesis
puntoYcoma
C (15)
B (25)
S (20)
input
abrirParentesis
identificador
cerrarParentesis
puntoYcoma
C (16)
lambda
cerrarCorchete
P (2)
B (25)
S (18)
identificador
abrirParentesis
L (46)
lambda
cerrarParentesis
puntoYcoma
P (2)
B (24)
let
T (29)
string
identificador
puntoYcoma
P (2)
B (25)

- S (17)
  - identificador
  - asignacion
  - E (31)
    - R (33)
      - U (36)
        - V (39)
          - W (40)
            - identificador
  - puntoYcoma
- P (2)
  - B (25)
    - S (18)
      - identificador
      - abrirParentesis
      - L (45)
        - E (31)
          - R (33)
            - U (36)
              - V (39)
                - W (40)
                  - identificador
        - Q (48)
          - lambda
      - cerrarParentesis
      - puntoYcoma
  - P (4)
    - lambda

Prueba 3

```
let number x;
let number z;
let boolean b;
input (x);
alert (x);
input (z);
alert (x+z);
b=x!=z;if (b)
x =
  x + 6
    + z
    - 1
    - (2
    - y
    - 6);
```

Tokens

```
<let, >
<number, >
<identificador, 0>
<puntoYcoma, >
<let, >
<number, >
<identificador, 1>
<puntoYcoma, >
<let, >
<boolean, >
<identificador, 2>
<puntoYcoma, >
<input, >
<abrirParentesis, >
<identificador, 0>
<cerrarParentesis, >
<puntoYcoma, >
<alert, >
<abrirParentesis, >
<identificador, 0>
<cerrarParentesis, >
<puntoYcoma, >
<input, >
<abrirParentesis, >
<identificador, 1>
<cerrarParentesis, >
<puntoYcoma, >
<alert, >
<abrirParentesis, >
<identificador, 0>
<opAritmetico, 1>
<identificador, 1>
<cerrarParentesis, >
<puntoYcoma, >
<identificador, 2>
<asignacion, >
<identificador, 0>
<opRelacional, 2>
<identificador, 1>
```

```
<puntoYcoma, >
<if, >
<abrirParentesis, >
<identificador, 2>
<cerrarParentesis, >
<identificador, 0>
<asignacion, >
<identificador, 0>
<opAritmetico, 1>
<cteEntera, 6>
<opAritmetico, 1>
<identificador, 1>
<opAritmetico, 2>
<cteEntera, 1>
<opAritmetico, 2>
<abrirParentesis, >
<cteEntera, 2>
<opAritmetico, 2>
<identificador, 3>
<opAritmetico, 2>
<cteEntera, 6>
<cerrarParentesis, >
<puntoYcoma, >
<EOF, >
```

Tabla Simbolos

```
CONTENIDO DE LA TABLA TSMAIN #1 :

      * LEXEMA : 'x'
------------------------------------------------------------
      * LEXEMA : 'z'
------------------------------------------------------------
      * LEXEMA : 'b'
------------------------------------------------------------
      * LEXEMA : 'y'
------------------------------------------------------------
```

Parse y Arbol Sintactico

Parse

Ascendente 27 24 27 24 28 24 20 25 40 39 36 33 31 19 25 20 25 40 39 40 37 36 33 31 19 25 40 39 36 40 39 35 33 31 17 25 40 39 36 33 31 40 39 43 37 40 37 43 38 43 39 40 38 43 38 36 33 31 41 38 36 33 31 17 23 4 2 2 2 2 2 2 2 2 2 1

Arbol Sintactico

Árbol resultado de:

Gramática: D:\OneDrive - Universidad Politécnica de Madrid\Universidad\3º\Procesadores de Lenguajes\Practica\ProcesadoresLenguajes\Pruebas\ASintactico-Tests\Herramienta Vast\aSintax\GramaticaVast.txt

Parse: D:\OneDrive - Universidad Politécnica de Madrid\Universidad\3º\Procesadores de Lenguajes\Practica\ProcesadoresLenguajes\docs\Analizador Sintactico\Pruebas realizadas\Prueba3\Parse.txt

- Z (1)
  - P (2)
    - B (24)
      - let
      - T (27)
        - number
      - identificador
      - puntoYcoma
    - P (2)
      - B (24)
        - let
        - T (27)
          - number
        - identificador
        - puntoYcoma
      - P (2)
        - B (24)
          - let
          - T (28)
            - boolean
          - identificador
          - puntoYcoma
        - P (2)
          - B (25)
            - S (20)
              - input
              - abrirParentesis
              - identificador
              - cerrarParentesis
              - puntoYcoma
          - P (2)
            - B (25)
              - S (19)
                - alert
                - abrirParentesis
                - E (31)
                  - R (33)
                    - U (36)
                      - V (39)
                        - W (40)
                          - identificador
                - cerrarParentesis
                - puntoYcoma
            - P (2)
              - B (25)
                - S (20)

input
abrirParentesis
identificador
cerrarParentesis
puntoYcoma
P (2)
B (25)
S (19)
alert
abrirParentesis
E (31)
R (33)
U (36)
V (37)
V (39)
W (40)
identificador
opAritmetico1
W (40)
identificador
cerrarParentesis
puntoYcoma
P (2)
B (25)
S (17)
identificador
asignacion
E (31)
R (33)
U (35)
U (36)
V (39)
W (40)
identificador
opRelacional2
V (39)
W (40)
identificador
puntoYcoma
P (2)
B (23)
if
abrirParentesis
E (31)
R (33)
U (36)
V (39)
W (40)
identificador
cerrarParentesis
S (17)
identificador

asignacion
E (31)
R (33)
U (36)
V (38)
V (38)
V (37)
V (37)
V (39)
W (40)
identificador
opAritmetico1
W (43)
cteEntera
opAritmetico1
W (40)
identificador
opAritmetico2
W (43)
cteEntera
opAritmetico2
W (41)
abrirParentesis
E (31)
R (33)
U (36)
V (38)
V (38)
V (39)
W (43)
cteEntera
opAritmetico2
W (40)
identificador
opAritmetico2
W (43)
cteEntera
cerrarParentesis
puntoYcoma
P (4)
lambda

Prueba 4

```
let numero x;
```

Errores

Error Sintactico en la linea:1

@Usuario: Se esperaba: number,boolean,string

@Internal: Error en el token: <identificador, 0>

Prueba 5

```
alert (msg)
```

Errores

Error Sintactico en la linea:1

@Usuario: Se esperaba: ;

@Internal: Error en el token: <EOF, >

Prueba 6

```
do{
    alert(mensaje);
}while(valor = 1);
```

Errores

Error Sintactico en la linea:3

@Usuario: Se esperaba: (,||,&&,==,!=,+,- o nada

@Internal: Error en el token: <asignacion, >

Webgrafía

Agradecimientos especiales a las siguientes fuentes de información

- Clase Pair: https://www.techiedelight.com/implement-pair-class-java/
- Oracle “SimpleTableDemo.java: https://docs.oracle.com/javase/tutorial/uiswing/examples/components/SimpleTableDemoProject/src/components/SimpleTableDemo.java
- Pagina web que nos genera las tablas: http://jsmachines.sourceforge.net/machines/slr.html