

## MANUAL TÉCNICO – IDE LFYDP

**Descripción:** El proyecto “IDE – LFYDP” es un IDE que se encuentra en fase de desarrollo, sin embargo, el IDE actualmente es capaz de crear, abrir y editar archivos y proyectos, también es capaz de reconocer expresiones determinadas y colorearlas.

### Notas:

- **IDE:** Visual Studio 2019, Community.
- **Paradigma Orientado a Objetos.**

### INTERFAZ GRÁFICA (Clases):

**AperturaArchivo:** Clase encargada de mostrar un Windows Form, el cual recibirá una ruta de archivo, y creará un proyecto, en base a su ubicación (la cual se puede ingresar de manera manual, a través de un TextBox o se puede seleccionar a través de un FolderBrowserDialog que se mostrará presionando un botón).

**CreacionDeProyecto:** Clase encargada de mostrar un Windows Form, el cual recibirá el nombre del proyecto, la ubicación que tendrá el proyecto (la cual se puede ingresar de manera manual, a través de un TextBox o se puede seleccionar a través de un FolderBrowserDialog que se mostrará presionando un botón), y también le permite al usuario decidir si junto a su proyecto creará un archivo inicial (.gt) a través de un CheckBox.

**NuevoArchivo:** Clase encargada de mostrar un Windows Form, el cual creará un archivo (.gt) dentro de un proyecto ya abierto. El usuario únicamente debe ingresar el nombre del archivo nuevo en un TextBox.

**IDE:** Clase encargada de mostrar un Windows Form, el cuál muestra el IDE, compuesto por:

- **Área de Texto:** RichTextBox en el que el usuario escribirá.
- **Barra de Herramientas:** Menú desplegable, que brinda opciones al usuario.
- **Barra de Acceso Rápido:** Panel que contiene un grupo de objetos tipo Button, los cuales servirán para la facilidad del usuario, también contiene un Label, el cual indica la posición que el cursor tiene dentro del área de texto.
- **Área de Archivos de Código Fuente:** ListBox que muestra el nombre de todos los archivos que el proyecto contiene, si el usuario presiona el nombre de algún archivo, el programa debe mostrar en el área de texto el código del archivo seleccionado.
- **Área Log:** TextBox multilínea, el cuál se encuentra vacío al inicio de la ejecución del programa, este TextBox debe ser llenado cuando el usuario decida compilar su código.

**PantallaInicial:** Clase encargada de mostrar un Windows Form, el cual le permitirá al usuario decidir si crear un proyecto, abrir un proyecto, crear un archivo o abrir un archivo; esto a través de botones.

## ARCHIVOS (Clases):

**Automata:** Clase encargada de manejar todo lo relacionado con la validación o descarte de tokens recibidos. El autómata ha sido programado por medio de métodos, por lo cual, cada método representa cada estado del autómata y las transiciones se dan en las condiciones que cada método posee. Sus métodos a describir:

- **public bool Comprobar(String cadenaIngresada):** Método encargado de comprobar si el resultado final de las transiciones entre estados es verdadero. Cadena ingresada se refiere al token recibido.
- **private void Inicializacion():** Inicializa las palabras reservadas que el lenguaje tiene.
- **Public String Color:** Getter que devuelve el color del cuál se deberá pintar el token ingresado.

**AnalizadorDeToken:** Clase encargada de manejar el token, previo a mandarlo al autómata. Sus métodos a describir:

- **public AnalizadorDeToken(RichTextBox txtArea):** Constructor de la clase. El RichTextBox crea una instancia en la clase del área de texto del IDE.
- **Private void Inicializacion():** Inicializa aquellos tokens que son el final de una cadena de dos caracteres.
- **Public void AnalizarToken():** Este método analiza los diferentes tipos de tokens que el usuario puede ingresar, desde tokens de un carácter, hasta tokens representados por una cadena de caracteres.
- **Private void ActualizarDatos():** Actualiza el número de fila y columna, en el que se encuentra el cursor que está dentro del área de texto.
- **Private void Pintar(int strt, int length):** Pinta la cadena de caracteres según lo establecido por el autómata. Strt marca el inicio de la cadena y length marca la longitud de la cadena.

**AnalizadorLog:** Clase encargada de Analizar los errores que se mostrarán en el área log. Sus métodos a describir:

- **Public void Analizar(RichTextBox txtArea, TextBox txtLog, int index):** Analiza los errores que se encuentran en el área de texto y los escribe en el área log. txtArea crea en la clase una instancia del área de texto del IDE, txtLog crea en la clase una instancia del área Log del IDE, index es la ubicación en la que se encuentra el cursor dentro del área de texto.

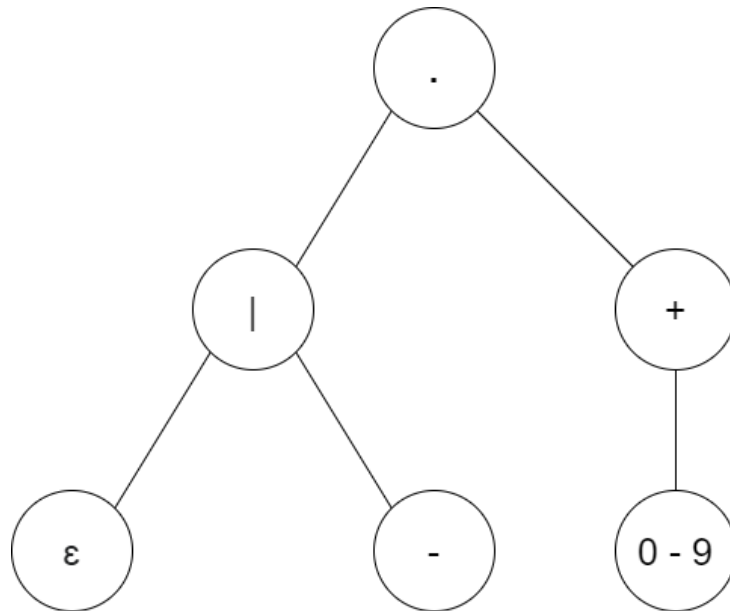
## **DEFINICIÓN DEL LENGUAJE:**

Expresiones Regulares utilizadas para definir el lenguaje:

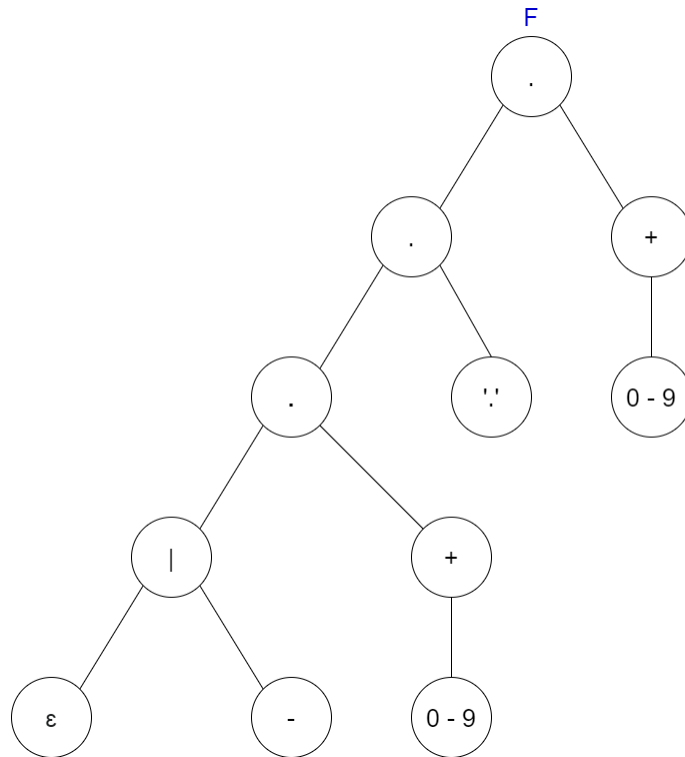
Descripción	Expresión Regular	Comentario
E - ENTERO	(- ? [ 0 - 9 ]+ )	
D - DECIMAL	(- ? [ 0 - 9 ] + '.' [ 0 - 9 ]+)	
C - CADENA	( 34 ( [ 0 - 33]   [ 35 - 255 ] ) * 34 )	Código ASCII para expresar carácter, 34 son comillas, los dos intervalos culaquier otro carácter
OA - OP. ARITMÉTICOS	[ ( ( + ? + )   ( - ? - ) )   '*'   '/' ]	
OR - OP.RELACIONALES	[ ( ( >   < ) = ? )   ( =   ! ) = ]	
OL - OP. LOGICOS	[ ( ( ' ' ' ' )   ( & & )   ! )	
SA - SIG. AGRUPACION	[ ' ( '   ' ) ' ]	
CO - Comment	[ '/' '*' (char)* '*' '/' ]   [ '/' '/' (char)* '/' '\n' ]	char hace referencia a cualquier caracter, con algunas excepciones
PR - PALABRA RESERV.	L   ( L   _ ) +	L hace referencia únicamente a letras
AF - ASIGN. FINAL	( =   ; )	

### DEFINICIÓN DEL AUTÓMATA (MÉTODO DEL ÁRBOL)

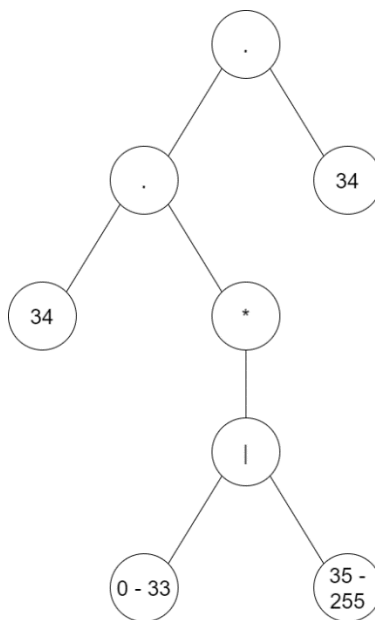
- **Enteros:**  $(- ? [ 0 - 9 ] + )$



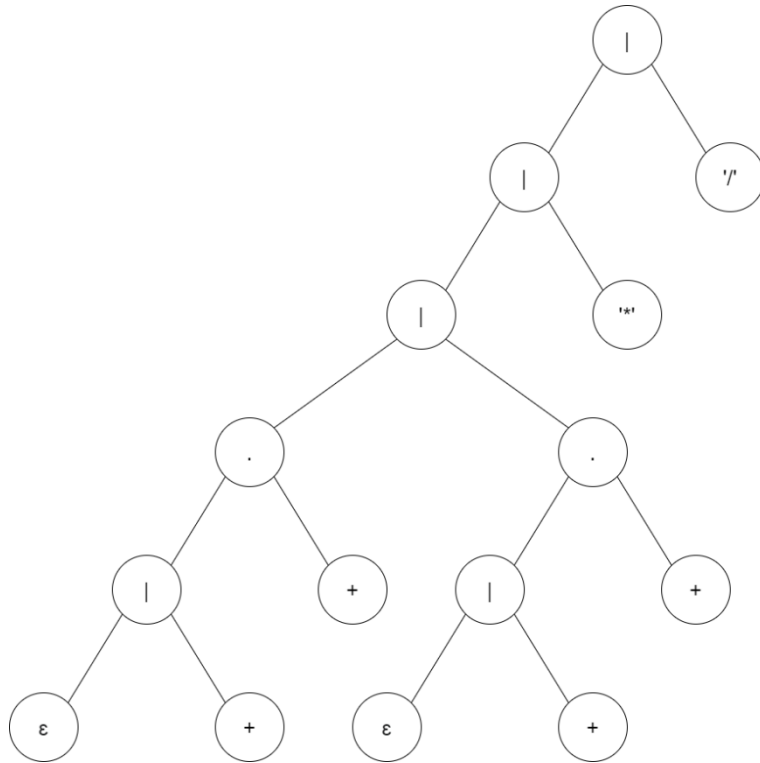
- **Decimal:**  $(- ? [ 0 - 9 ] + '.' [ 0 - 9 ] + )$



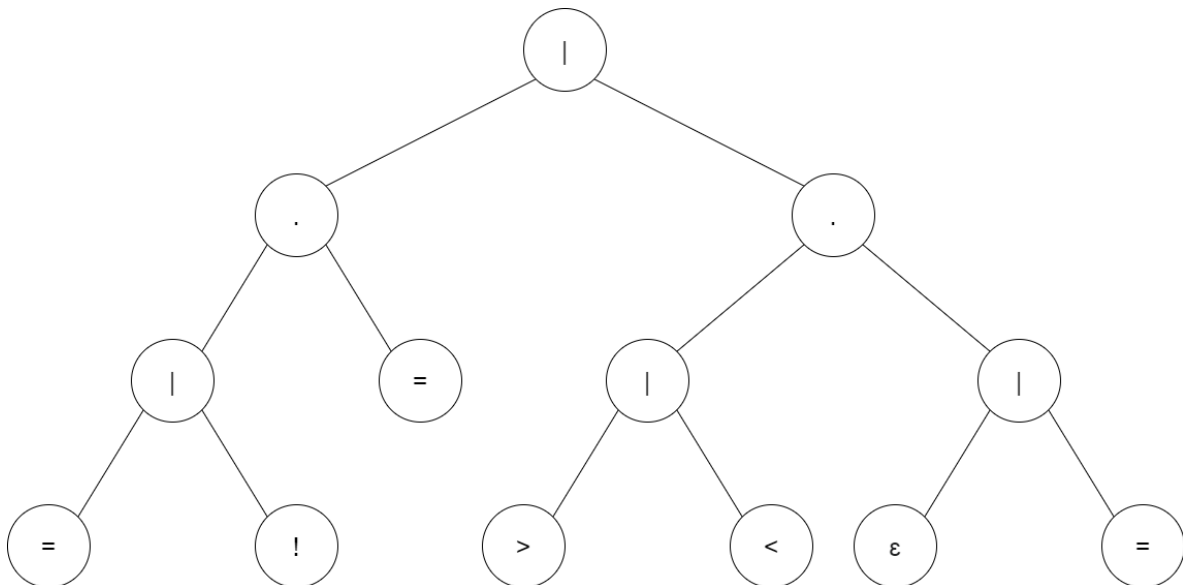
- Cadena: ( 34 ( [ 0 - 33] | [ 35 - 255 ] ) \* 34 )



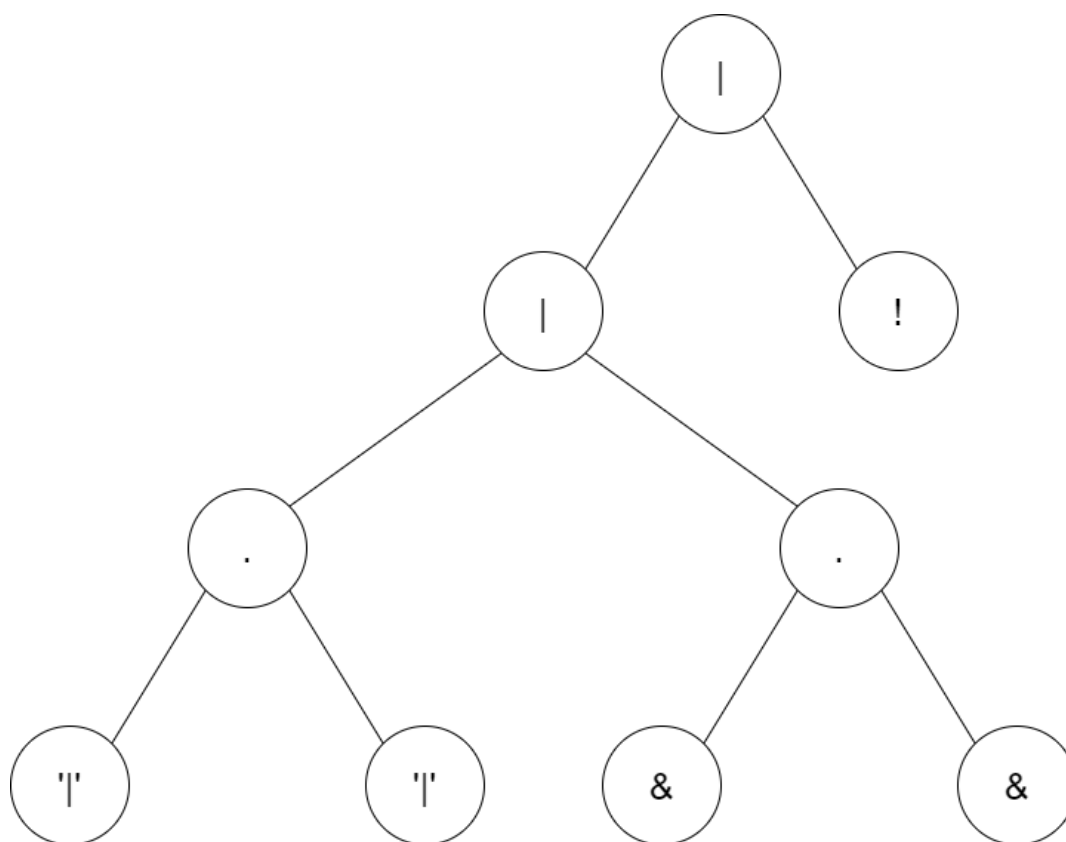
- **Operadores Aritméticos:** [ ( (+?+) | (-?-) ) | '\*' | '/' ]



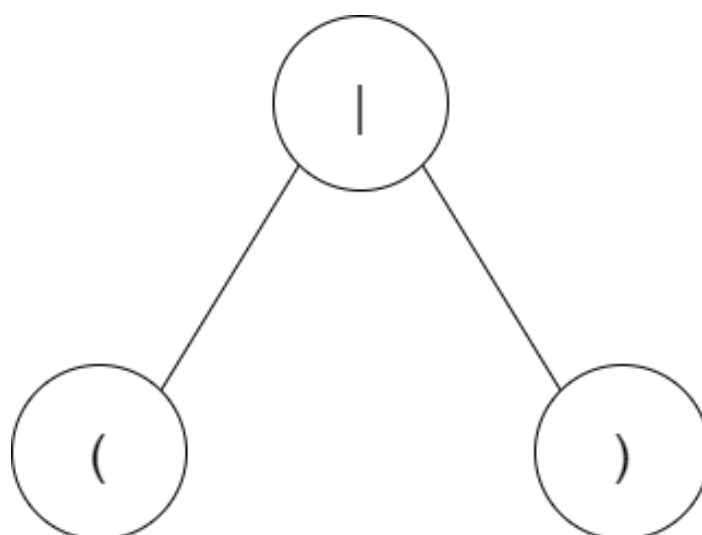
- **Operadores Relacionales:** [ ( > | < ) = ? ) | (=|!)= ]



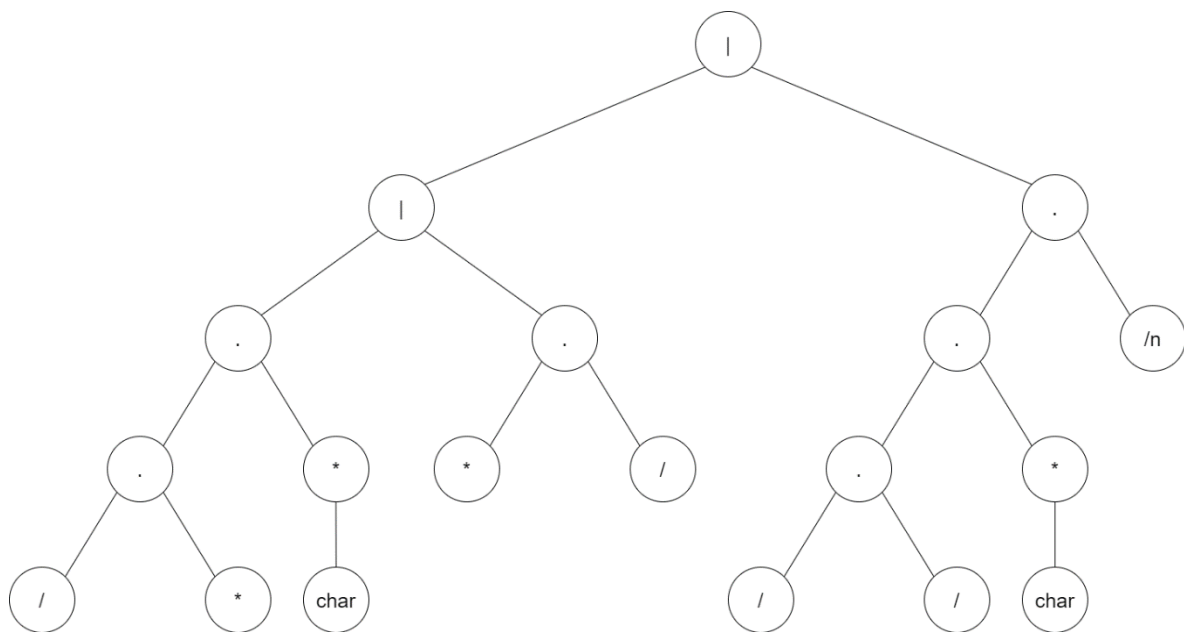
- **Operadores Lógicos:** ( ( ' | ' ) | ( & & ) | ! )



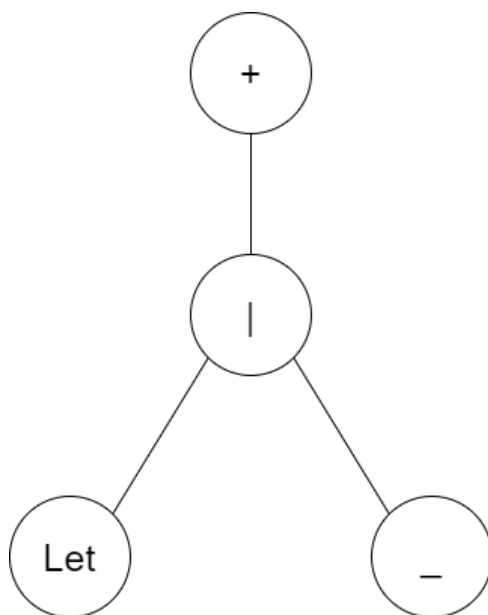
- **Signos de Agrupación:** [ '(' ')' ]



- **Comentario:** [ '/' '\*' (char)\* '\*' '/' ] [ '/' '/' (char)\* '\n' ]

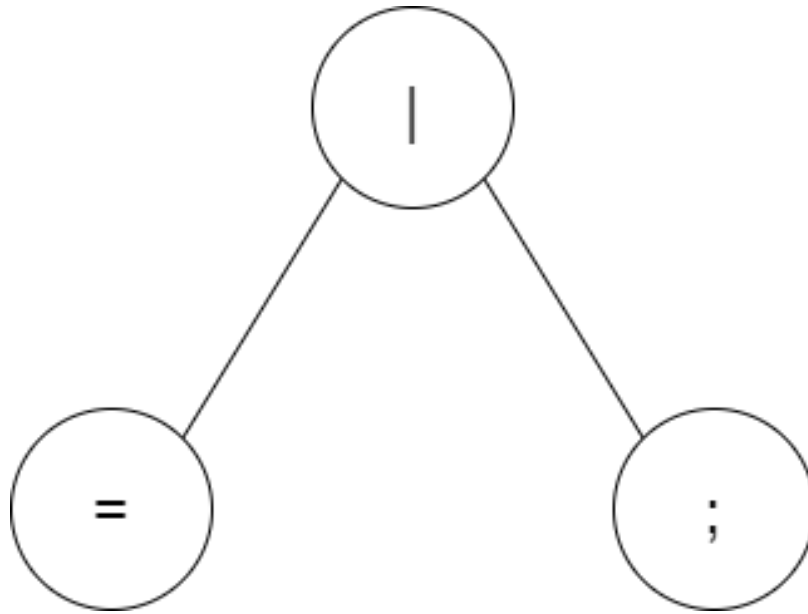


- Palabras:  $L | (L | \_ ) +$



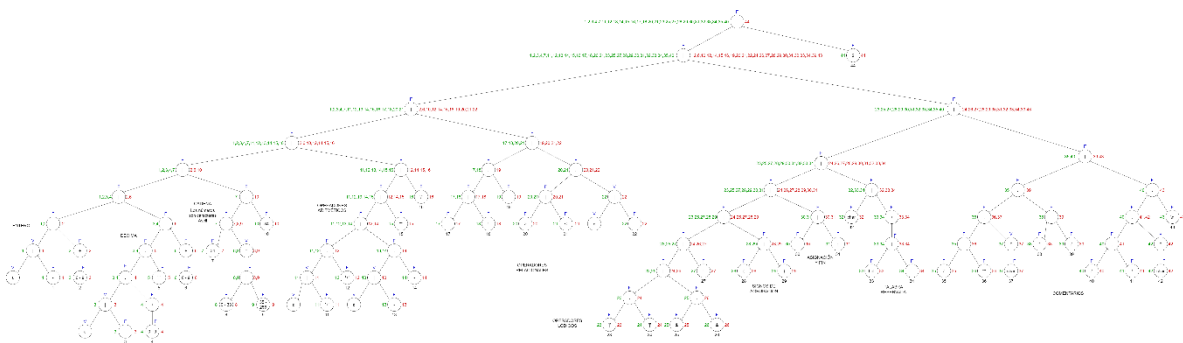


- **Asignación y Fin de Sentencia: ( = | ; )**



### **RESOLUCIÓN:**

- Se crea el árbol, se determinó cada nodo era anulable o no era anulable y se hizo el cálculo de siguientes:



(Debido a la baja calidad de la imagen en este documento, la imagen se adjunta a la carpeta del proyecto en mejor calidad)

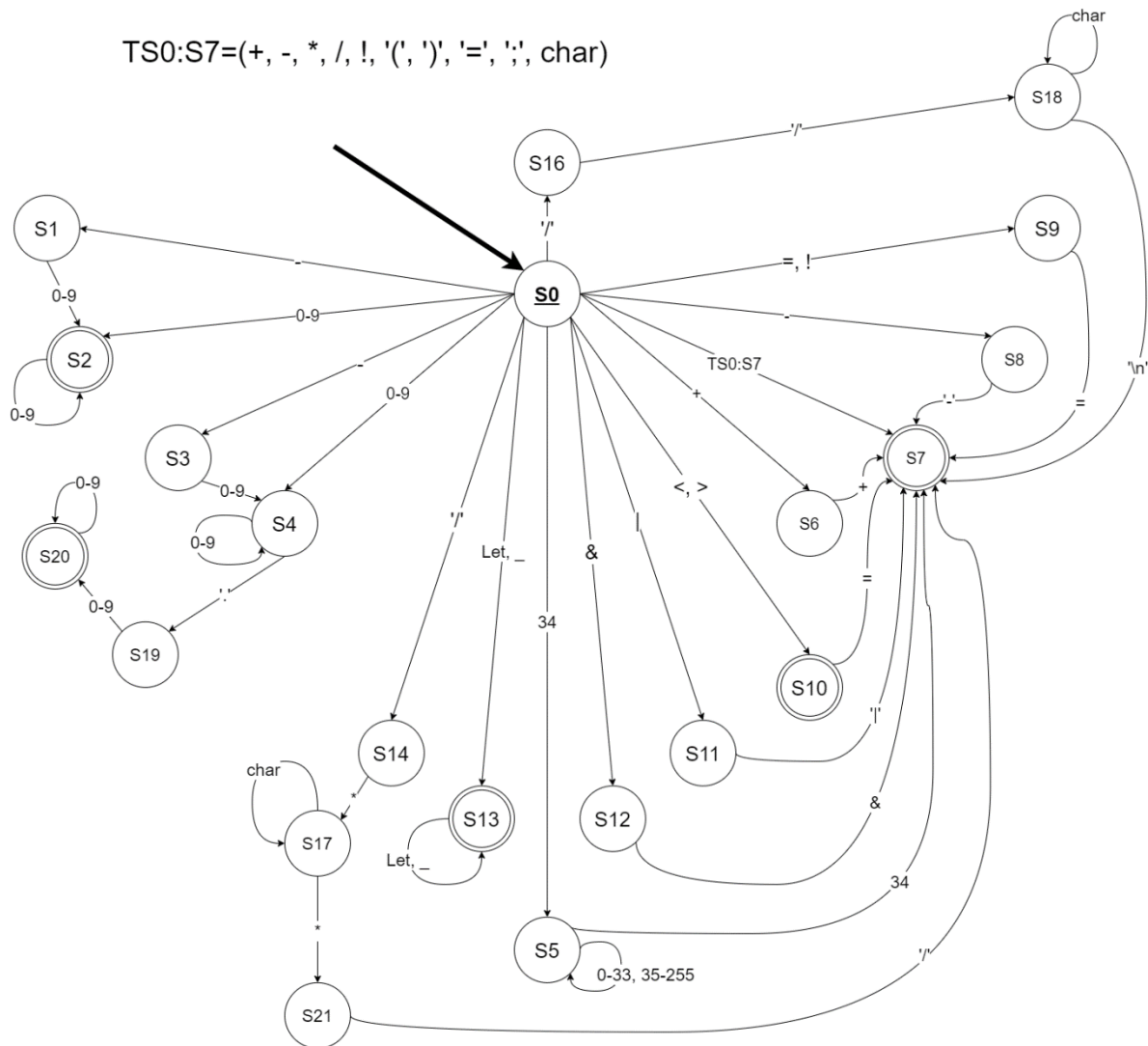
Se hizo la tabla de siguientes y transiciones:

No	$\Sigma$	Siguiente
1	-	2
2	0 - 9	2,44
3	-	4
4	0 - 9	4,5
5	.	6
6	0 - 9	6,44
7	34	8,9,10
8	0 - 33	8,9,10
9	35 - 255	8,9,10
10	34	44
11	+	12
12	+	44
13	-	14
14	-	44
15	*	44
16	/	44
17	=	19
18	!	19
19	=	44
20	>	22,44
21	<	22,44
22	=	44
23		24
24		44
25	&	26
26	&	44
27	!	44
28	(	44
29	)	44
30	=	44
31	;	44
32	char	44
33	Let	33,34,44
34	_	33,34,44
35	/	36
36	*	37,38
37	char	37,38
38	*	39
39	/	44
40	/	41
41	/	42,43
42	char	42,43
43	\n	44
44	\$	

Q/Σ	Siguientes	Transición
S0 = 1,2,3,4,7,11,12,13,14,15,16,17,18,20,21,23,25,27,28,29,30,31,32,33,34,35,40	Sig(1)={2}=S1	δ(S0,-)=S1
	Sig(2)={2,44}=S2	δ(S0,0-9)=S2
	Sig(3)={4}=S3	δ(S0,-)=S3
	Sig(4)={4,5}=S4	δ(S0,0-9)=S4
	Sig(7)={8,9,10}=S5	δ(S0,34)=S5
	Sig(11)={12}=S6	δ(S0,+)=S6
	Sig(12)={44}=S7	δ(S0,+)=S7
	Sig(13)={14}=S8	δ(S0,-)=S8
	Sig(14)={44}=S7	δ(S0,-)=S7
	Sig(15)={44}=S7	δ(S0,*)=S7
	Sig(16)={44}=S7	δ(S0,/)=S7
	Sig(17)={19}=S9	δ(S0,=)=S9
	Sig(18)={19}=S9	δ(S0,!)=S9
	Sig(20)={22,44}=S10	δ(S0,>)=S10
	Sig(21)={22,44}=S10	δ(S0,<)=S10
	Sig(23)={24}=S11	δ(S0, )=S11
	Sig(25)={26}=S12	δ(S0,&)=S12
	Sig(27)={44}=S7	δ(S0,!)=S7
	Sig(28)={44}=S7	δ(S0,'')=S7
	Sig(29)={44}=S7	δ(S0,')=S7
	Sig(30)={44}=S7	δ(S0,=)=S7
	Sig(31)={44}=S7	δ(S0,;)=S7
	Sig(32)={44}=S7	δ(S0,char)=S7
	Sig(33)={33,34,44}=S13	δ(S0,Let)=S13
	Sig(34)={33,34,44}=S13	δ(S0,_)=S13
	Sig(35)={36}=S14	δ(S0,/)=S14
	Sig(40)={41}=S16	δ(S0,/)=S16
S1 = {2}	Sig(2)={2,44}=S2	δ(S1,0-9)=S2
S2 = {2,44}	Sig(2)={2,44}=S2	δ(S2,0-9)=S2
S3 = {4}	Sig(4)={4,5}=S4	δ(S3,0-9)=S4
S4 = {4,5}	Sig(4)={4,5}=S4	δ(S4,0-9)=S4
	Sig(5)={6}=S19	δ(S4,.)=S19
S5 = {8,9,10}	Sig(8)={8,9,10}=S5	δ(S5,0-33)=S5
	Sig(9)={8,9,10}=S5	δ(S5,35-255)=S5
	Sig(10)={44}=S7	δ(S5,34)=S7
S6 = {12}	Sig(12)={44}=S7	δ(S6,+)=S7
S7 = {44}		
S8 = {14}	Sig(14)={44}=S7	δ(S8,-)=S7
S9 = {19}	Sig(19)={44}=S7	δ(S9,=)=S7
S10 = {22,44}	Sig(22)={44}=S7	δ(S10,=)=S7
S11 = {24}	Sig(24)={44}=S7	δ(S11, )=S7
S12 = {26}	Sig(26)={44}=S7	δ(S12,&)=S7
S13 = {33,34,44}	Sig(33)={33,34,44}=S13	δ(S13,Let)=S13
	Sig(34)={33,34,44}=S13	δ(S13,_)=S13
S14 = {36}	Sig(36)={37,38}=S17	δ(S14,*)=S17
S16 = {41}	Sig(41)={42,43}=S18	δ(S16,/)=S18
S17 = {37,38}	Sig(37)={37,38}=S17	δ(S17,char)=S17
	Sig(38)={39}=S21	δ(S17,*)=S21
S18 = {42,43}	Sig(42)={42,43}=S18	δ(S18,char)=S18
	Sig(43)={44}=S7	δ(S18,\n)=S7
S19 = {6}	Sig(6)={6,44}=S20	δ(S19,0-9)=S20
S20 = {6,44}	Sig(6)={6,44}=S20	δ(S20,0-9)=S20
S21 = {39}	Sig(39)={4}=S7	δ(S21,/)=S7

- Se obtuvo el autómata (En este caso un Autómata Finito No Determinista):

TS0:S7=(+, -, \*, /, !, '(', ')', '=', ';', char)



(La imagen original del autómata se adjunta en la carpeta del proyecto)

- Se hizo una tabla para convertir el AFnD en AFD

Q/2		0-9		34	0-33	35-255	+	*	/	=		>	<		&	(	)	:	char	Let		\n
Q0	S1,S3,S7,S8-Q3	S2,S4-Q2		S5-Q3			S6,S7-Q4	S7-Q5	S7,S14,S16-Q6	S7,S9-Q7	S7,S9-Q7	S1-Q8	S10-Q8	S11-Q9	S12-Q10	S7+Q5	S7+Q5	S7+Q5	S7+Q5	S13-Q11	S13-Q11	
Q1-S1,S1,S7,S8	S7+Q5	S2,S4+Q2																				
Q2-S2,S4		S2,S4+Q2	S19-Q12																			
Q3-S5				S7-Q5	S5-Q3	S5-Q3																
Q4-S6,S7							S7-Q5															
Q5-S7																						
Q6-S7,S14,S16								S17-Q13	S18-Q17													
Q7-S7,S9									S21-Q16													
Q8-S10									S7-Q5													
Q9-S11														S7-Q5								
Q10-S12															S7-Q5							
Q11-S13																S7-Q5				S13-Q11	S13-Q11	
Q12-S19		S20-Q15																				
Q13-S17							S21-Q16												S17-Q13			
Q14-S20		S20-Q15																				
Q15-S21								S7-Q5														
Q16-S23																			S18-Q17		S7-Q5	
Q17-S18																						

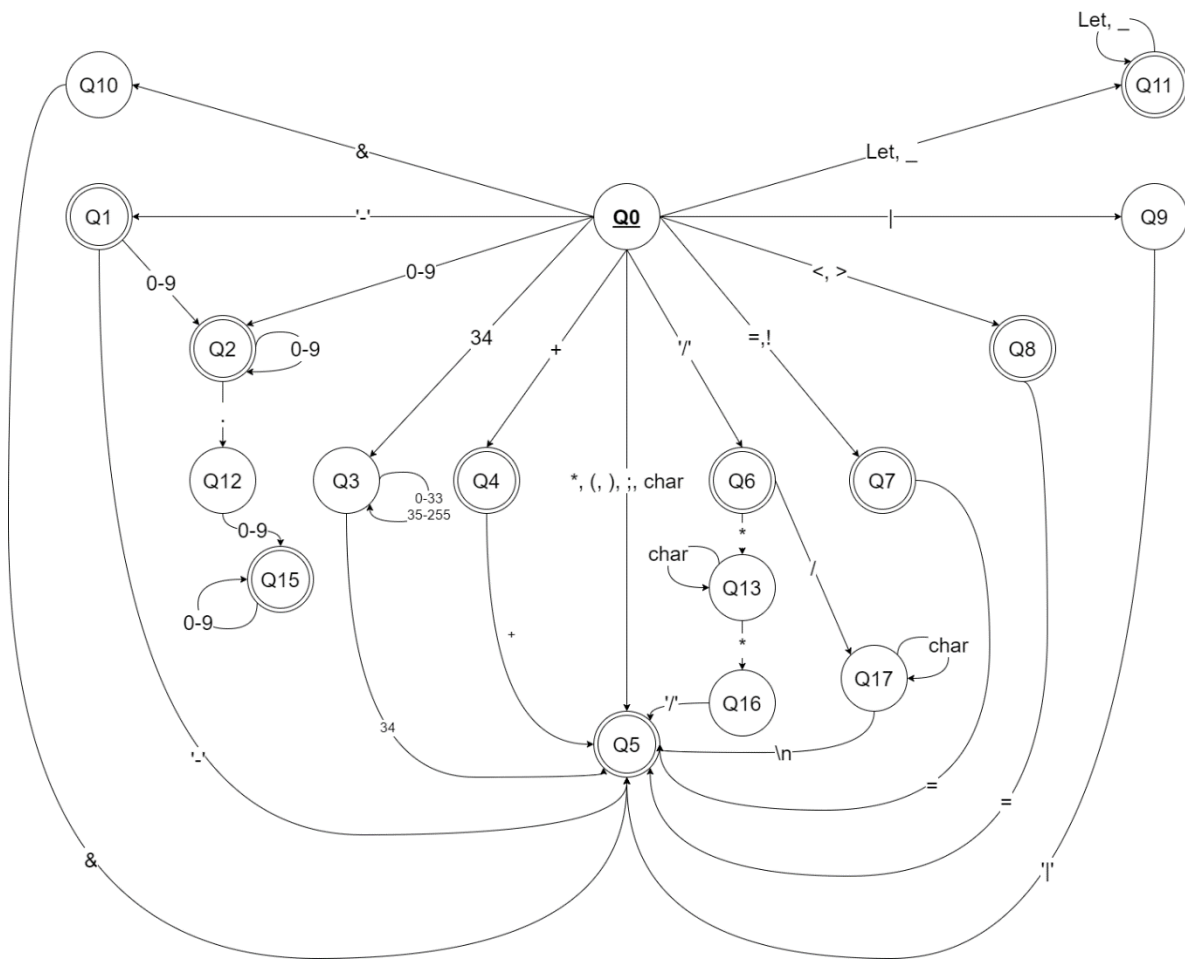
(Debido al tamaño de la tabla, la misma se adjunta a la carpeta del proyecto)

- Se hizo una tabla de transiciones:

	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17
Q0		.	0-9	34	+	*, (, ), :, char	/	'', ,	<, >		&	Let, _						
Q1			0-9			.												
Q2			0-9															
Q3				0-33, 35-255														
Q4						34												
Q5						+												
Q6																		
Q7						=								*				/
Q8						=												
Q9																		
Q10						&												
Q11												Let, _						
Q12																0-9		
Q13														char			*	
Q15																0-9		
Q16						/												
Q17						\n												char

(Debido al tamaño de la tabla, la misma se adjunta a la carpeta del proyecto)

- Se obtuvo el Autómata Finito Determinista



(La imagen original del autómata se adjunta en la carpeta del proyecto)

### **DEFINICIÓN FORMAL**

$$A = \{Q, \Sigma, \delta, q_0, F\}$$

$$Q = \{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6, Q_7, Q_8, Q_9, Q_{10}, Q_{11}, Q_{12}, Q_{13}, Q_{15}, Q_{16}\}$$

$$\Sigma = \{-, +, ' ', 0-9, \&, |, <, >, =, !, /, *, (, ), :, \text{CHAR}, 34, 0-33, 35-255, \text{Let}, \_ \}$$

$$Q_0 = q_0$$

$$F = \{Q_1, Q_2, Q_4, Q_5, Q_6, Q_7, Q_8, Q_{11}, Q_{15}\}$$

Función de Transición Extendida:

$\delta(Q_0, .) = Q_1$	$\delta(Q_4, +) = Q_5$
$\delta(Q_0, 0-9) = Q_2$	$\delta(Q_6, *) = Q_{13}$
$\delta(Q_0, 34) = Q_3$	$\delta(Q_6, /) = Q_{17}$
$\delta(Q_0, +) = Q_4$	$\delta(Q_7, =) = Q_5$
$\delta(Q_0, *) = Q_5$	$\delta(Q_8, =) = Q_5$
$\delta(Q_0, ' ') = Q_5$	$\delta(Q_9,  ) = Q_5$
$\delta(Q_0, ' ') = Q_5$	$\delta(Q_{10}, \&) = Q_5$
$\delta(Q_0, :) = Q_5$	$\delta(Q_{11}, \text{Let}) = Q_{11}$
$\delta(Q_0, \text{char}) = Q_5$	$\delta(Q_{11}, \_) = Q_{11}$
$\delta(Q_0, /) = Q_6$	$\delta(Q_{12}, 0-9) = Q_{15}$
$\delta(Q_0, =) = Q_7$	$\delta(Q_{13}, \text{char}) = Q_{13}$
$\delta(Q_0, '!') = Q_7$	$\delta(Q_{13}, *) = Q_{16}$
$\delta(Q_0, <) = Q_8$	$\delta(Q_{15}, 0-9) = Q_{15}$
$\delta(Q_0, >) = Q_8$	$\delta(Q_{16}, /) = Q_5$
$\delta(Q_0,  ) = Q_9$	$\delta(Q_{17}, \backslash n) = Q_5$
$\delta(Q_0, \&) = Q_{10}$	$\delta(Q_{17}, \text{char}) = Q_{17}$
$\delta(Q_0, \text{Let}) = Q_{11}$	
$\delta(Q_0, ' ') = Q_{11}$	
$\delta(Q_1, 0-9) = Q_2$	
$\delta(Q_1, -) = Q_5$	
$\delta(Q_2, 0-9) = Q_2$	
$\delta(Q_2, .) = Q_{12}$	
$\delta(Q_3, 0-33) = Q_3$	
$\delta(Q_3, 35-255) = Q_3$	



## GRAMÁTICA LL(1)

**G(N, T, S, P):**

❖ **N:**

- Codigo
- SimboloInicial
- Texto
- DeclaVariable
- Incrementable
- VariablesL
- id
- idVar
- valor
- TipoValor
- Suma
- Suma'
- Multi
- Multi'
- Pot
- Pot'
- Unar
- Elem
- Impresión
- ValImprimir
- Imprimir
- Append
- Lectura
- ValLeer
- Condicion
- DescCondicion
- CuerpoDescCond
- Cond2
- ValCondicion
- DescValCondicion
- Condicionante
- SigDif
- CondicionantePlus
- CicloM
- CicloHM
- DescCicloHM
- CicloF
- EstructCicloF
- EstructDesde

❖ **T:**

- EstructHasta
- EstructIncremento
- desde
- hasta
- PRprincipal
- (
- )
- {
- }
- PRdatoPrimi
- TokId
- ++
- --
- ,
- ;
- cadena
- entero
- decimal
- booleano
- char
- Num
- PRImprimir
- +
- -
- \*
- /
- ^
- PRLeer
- PRSI
- =
- PRSINO\_SI
- PRSINO
- Numero
- OpRel
- PRBooleana
- !
- OpLogico
- PRHACER
- PRMIENTRAS
- PRDESDE
- PRHASTA
- PRINCREMENTO
- NumE

❖ **S:** Codigo

❖ **P: Producción**

Codigo	→	PRprincipal SimboloInicial Texto
SimboloInicial	→	( ) {
Texto	→	VariablesL Texto   DeclaVariable Texto   Impresión Texto   Lectura Texto   Condicion Texto   CicloM Texto   CicloHM Texto   CicloF Texto   }
DeclaVariable	→	TokId Incrementable ;
Incrementable	→	++   --   = TipoValor
VariablesL	→	PRDatoPrimi idVar id
id	→	, idVar id   ;
idVar	→	TokId valor
valor	→	= TipoValor   e
TipoValor	→	cadena   Suma   booleano   char
Suma	→	Multi Suma'

Suma'	→	+ Multi Suma'   - Multi Suma'   e
Multi	→	Pot Multi'
Multi'	→	* Pot Multi'   / Pot Multi'   e
Pot	→	Unar Pot'
Pot'	→	^ Pot   e
Unar	→	- Elem   Elem
Elem	→	TokId   Num   (Suma)
Impresión	→	PRImprimir ValImprimir
ValImprimir	→	( Imprimir Append
Imprimir	→	TokId   cadena   Num   char
Append	→	+ Imprimir Append   ) ;
Lectura	→	PRLeer ValLeer ;
ValLeer	→	( TokId )
Condicion	→	PRSI DescCondicion Cond2

DescCondicion	→	ValCondicion CuerpoDescCond
CuerpoDescCond	→	{ Texto
Cond2	→	PRSINO_SI DescCondicion Cond2   PRSINO CuerpoDescCond   e
ValCondicion	→	( DescValCondicion )
DescValCondicion	→	SigDif Condicionante CondicionantePlus
Condicionante	→	Suma OpRel Suma   PRBooleana
SigDif	→	! SigDif   e
CondicionantePlus	→	OpLogico DescValCondicion   e
CicloM	→	PRMIENTRAS ValCondicion CuerpoDescCond
CicloHM	→	DescCicloHM ValCondicion
DescCicloHM	→	PRHACER CuerpoDescCond PRMIENTRAS
CicloF	→	EstructCicloF CuerpoDescCond
EstructCicloF	→	EstructDesde EstructHasta EstructIncremento

EstructDesde	→	PRDESDE desde
EstructHasta	→	PRHASTA hasta
EstructIncremento	→	PRINCREMENTO NumE
desde	→	TokId = NumE
hasta	→	TokId OpRel NumE

## Primeros:

No Terminal	Primeros
Codigo	PRprincipal
SimboloInicial	(
Texto	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
DeclaVariable	TokId
Incrementable	++, --, =
VariablesL	PRDatoPrimi
id	",", ;
idVar	TokId
valor	=, e
TipoValor	cadena, -, TokId, Num, (, booleano, char
Suma	-, TokId, Num, (
Suma'	+, -, e
Multi	-, TokId, Num, (
Multi'	*, /, e
Pot	-, TokId, Num, (
Pot'	^, e
Unar	-, TokId, Num, (
Elem	TokId, Num, (
Impresión	PRImprimir
VallImprimir	(
Imprimir	TokId, cadena, Num, char
Append	+, )
Lectura	PRLeer
ValLeer	(
Condicion	PRSI
DescCondicion	(
CuerpoDescCond	{
Cond2	PRSINO_SI, PRSINO, e
ValCondicion	(
DescValCondicion	!, -, TokId, Num, (, e
Condicionante	-, TokId, Num, (
SigDif	!, e
CondicionantePlus	OpLogico, e
CicloM	PRMIENTRAS
CicloHM	PRHACER
DescCicloHM	PRHACER
CicloF	PRDESDE
EstructCicloF	PRDESDE
EstructDesde	PRDESDE
EstructHasta	PRHASTA
EstructIncremento	PRINCREMENTO
desde	TokId
hasta	TokId

## Siguientes:

No Terminal	Siguientes
Codigo	\$
SimboloInicial	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
Texto	\$, PRSINO_SI, PRSINO, PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE, }, PRMIENTRAS, e
DeclaVariable	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
Incrementable	;
VariablesL	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
id	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
idVar	" " ;
valor	" " ;
TipoValor	;; " " ;
Suma	;; " " ; , OpRel, OpLogico, e
Suma'	;; " " ; , OpRel, OpLogico, e
Multi	+, -, e, ';; " " ; , OpRel, OpLogico, e
Multi'	+, -, e, ';; " " ; , OpRel, OpLogico, e
Pot	*, /, e, +, -, e, ';; " " ; , OpRel, OpLogico, e
Pot'	*, /, e, +, -, e, ';; " " ; , OpRel, OpLogico, e
Unar	*, /, e, +, -, e, ';; " " ; , OpRel, OpLogico, e
Elem	*, /, e, +, -, e, ';; " " ; , OpRel, OpLogico, e
Impresión	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
Vallmpimir	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
Imprimir	+, )
Append	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
Lectura	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
ValLeer	;
Condicion	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
DescCondicion	PRSINO_SI, PRSINO, PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }, e
CuerpoDescCond	PRSINO_SI, PRSINO, PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE, }, PRMIENTRAS, e
Cond2	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
ValCondicion	{, pSig(CicloHM)
DescValCondicion	}
Condicionante	OpLogico, ), e
SigDif	-, TokId, Num, (
CondicionantePlus	)
CicloM	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
CicloHM	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
DescCicloHM	(
CicloF	PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
EstructCicloF	{
EstructDesde	PRHASTA
EstructHasta	PRINCREMENTO
EstructIncremento	{
desde	PRHASTA
hasta	PRINCREMENTO

### Tabla de Análisis Sintáctico:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------

(Debido al Tamaño de la tabla, la misma es ilegible, por lo tanto se añade a la carpeta “Documentación Fase 2” un archivo de excel el cuál contiene la tabla).



## Diagrama de Clases

