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
  - EstructHasta
  - EstructIncremento
  - desde
  - hasta
- **T:**
  - PRprincipal
  - (
  -)
  - {
  - }
  - PRDatoPrimi
  - TokId
  - ++
  - --
  - ,
  - ;
  - cadena
  - entero
  - decimal
  - booleano
  - char
  - NumE
  - 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 = TipoValor ; TokId Incrementable ;
Incrementable	→	++ | --
VariablesL	→	PRDatoPrimi idVar id
id	→	, idVar id | ;
idVar	→	TokId valor
valor	→	= TipoValor | e
TipoValor	→	cadena | Suma | decimal | 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
ValImprimir	(
Imprimir	TokId, cadena, Num, char
Append	+,)
Lectura	PRLeer
ValLeer	(
Condicion	PRSI
DescCondicion	(
CuerpoDescCond	{
Cond2	PRSINO_SI, PRSINO, e
ValCondicion	(
DescValCondicion	!, 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. }
ValImprimir	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	{, PRDatoPrimi, TokId, PRImprimir, PRLeer, PRSI, PRMIENTRAS, PRHACER, PRDESDE. }
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