

# Gramática de contexto libre sin recursividad por la izquierda y factorizada por la izquierda:

$P = \{$

(1)  $E \rightarrow T E'$

(2)  $E' \rightarrow + T E'$

(3)  $E' \rightarrow \varepsilon$

(4)  $T \rightarrow F T'$

(5)  $T' \rightarrow * F T'$

(6)  $T' \rightarrow \varepsilon$

(7)  $F \rightarrow ( E )$

(8)  $F \rightarrow \text{identificador}$

(9)  $F \rightarrow \text{número}$

$\}$

## *Análisis sintáctico mediante descenso recursivo (con retroceso)*

**Función E: lógico**

**inicio**

{SIMULACIÓN DE LA REGLA  $E \rightarrow T E'$ }

**si T = VERDADERO**

**entonces**

**si E' = VERDADERO**

**entonces**

**DEVOLVER VERDADERO**

**si no**

*ERROR*

**DEVOLVER FALSO**

**fin si**

**si no**

*ERROR*

**DEVOLVER FALSO**

**fin si**

**fin**

## **Función E': lógico**

### **inicio**

referencia  $\leftarrow$  Componente léxico actual

{SIMULACIÓN DE LA REGLA  $E' \rightarrow + T E'$ }

**si** emparejar( “+” ) = **VERDADERO**

**entonces**

avanzar\_entrada

**si** T = **VERDADERO**

**entonces**

**si** E' = **VERDADERO**

**entonces**

**DEVOLVER VERDADERO**

**si no**

retroceder\_entrada(referencia)

**fin si**

**si no**

retroceder\_entrada(referencia)

**fin si**

**si no**

retroceder\_entrada(referencia)

**fin si**

{SIMULACIÓN DE LA REGLA  $E' \rightarrow \epsilon$ }

**DEVOLVER VERDADERO**

**fin**

## **Función T: lógico**

**inicio**

{SIMULACIÓN DE LA REGLA  $T \rightarrow F T'$ }

**si F = VERDADERO**

**entonces**

**si T' = VERDADERO**

**entonces**

**DEVOLVER VERDADERO**

**si no**

*ERROR*

**DEVOLVER FALSO**

**fin si**

**si no**

*ERROR*

**DEVOLVER FALSO**

**fin si**

**fin**

**Función T': lógico**

**inicio**

referencia  $\leftarrow$  Componente léxico actual

{SIMULACIÓN DE LA REGLA T'  $\rightarrow$  \* F T'}

**si** emparejar( "\*" ) = **VERDADERO**

**entonces**

avanzar\_entrada

**si** F = **VERDADERO**

**entonces**

**si** T' = **VERDADERO**

**entonces**

**DEVOLVER VERDADERO**

**si no**

retroceder\_entrada(referencia)

**fin si**

**si no**

retroceder\_entrada(referencia)

**fin si**

**si no**

retroceder\_entrada(referencia)

**fin si**

{SIMULACIÓN DE LA REGLA T'  $\rightarrow$   $\epsilon$ }

**DEVOLVER VERDADERO**

**fin**

## **Función F: lógico**

### **inicio**

referencia  $\leftarrow$  Componente léxico actual

{SIMULACIÓN DE LA REGLA  $F \rightarrow ( E )$ }

**si** emparejar(“(” ) = **VERDADERO**

**entonces**

avanzar\_entrada

**si** E = **VERDADERO**

**entonces**

**si** emparejar(“)” ) = **VERDADERO**

**entonces**

avanzar\_entrada

**DEVOLVER VERDADERO**

**si no**

retroceder\_entrada(referencia)

**fin si**

**si no**

retroceder\_entrada(referencia)

**fin si**

**si no**

retroceder\_entrada(referencia)

**fin si**

{SIMULACIÓN DE LA REGLA F → **identificador**}

**si** emparejar( "**identificador**" ) = **VERDADERO**

**entonces**

        avanzar\_entrada

**DEVOLVER VERDADERO**

**si no**

        retroceder\_entrada(referencia)

**fin si**

{SIMULACIÓN DE LA REGLA F → **número**}

**si** emparejar( "**número**" ) = **VERDADERO**

**entonces**

        avanzar\_entrada

**DEVOLVER VERDADERO**

**si no**

        retroceder\_entrada(referencia)

**fin si**

{SI NINGUNA REGLA ES CORRECTA, SE DEVUELVE FALSO}

**DEVOLVER FALSO**

**fin**

## CONJUNTOS PRIMERO Y SIGUIENTE

	PRIMERO	SIGUIENTE
<b>E</b>	(, identificador, número	\$, )
<b>E'</b>	+ , $\epsilon$	\$, )
<b>T</b>	(, identificador, número	+ , \$, )
<b>T'</b>	* , $\epsilon$	+ , \$, )
<b>F</b>	(, identificador, número	+ , * , \$, )

## TABLA PREDICTIVA

	identificador	número	+	*	(	)	\$
<b>E</b>	1	1			1		
<b>E'</b>			2			3	3
<b>T</b>	4	4			4		
<b>T'</b>			6	5		6	6
<b>F</b>	8	9			7		

## *Algoritmo de construcción de la tabla predictiva*

```
para cada  $A \rightarrow \alpha \in P$  hacer  
  si  $a \in \text{primero}(\alpha)$   
    entonces  $A \rightarrow \alpha \in M[A,a]$   
  fin si  
  
  si  $\varepsilon \in \text{primero}(\alpha)$   
    entonces  
      para cada  $b \in \text{siguiente}(A)$  hacer  
         $A \rightarrow \alpha \in M[A,b]$   
      fin para  
    fin si  
fin para
```

- Las celdas vacías de  $M$  han de ser tratadas con procedimientos de control de errores.
- Las celdas de  $M$  con valores múltiples indican que la gramática no admite un análisis descendente predictivo.

## *Análisis sintáctico descendente predictivo y recursivo*

**Procedimiento E;**

**inicio**

{SIMULACIÓN DE LA REGLA  $E \rightarrow T E'$ }

**si** (token = “**identificador**”) o (token = “**número**”) o (token = “(“)  
**entonces**

T;  
E’;

**si no**

*ERROR*

**fin si**

**fin**

## Procedimiento E'

**inicio**

{SIMULACIÓN DE LA REGLA  $E' \rightarrow + T E'$ }

**si** (token = “+” )

**entonces**

avanzar\_entrada(token)

T;

E';

**si no**

{SIMULACIÓN DE LA REGLA  $E' \rightarrow \epsilon$ }

**si** (token = “)”) o (token = “\$”)

**entonces**

{SENTENCIA NULA}

**si no**

*ERROR*

**fin si**

**fin si**

**fin**

**Procedimiento T;**

**inicio**

{SIMULACIÓN DE LA REGLA  $T \rightarrow F T'$ }

**si** (token = “**identificador**”) o (token = “**número**”) o (token = “(“)

**entonces**

F;

T’;

**si no**

*ERROR*

**fin si**

**fin**

## Procedimiento T'

inicio

{SIMULACIÓN DE LA REGLA  $T' \rightarrow * F T'$ }

si (token = “\*” )

entonces

avanzar\_entrada(token)

F;

T';

si no

{SIMULACIÓN DE LA REGLA  $T' \rightarrow \epsilon$ }

si (token = “+”) o (token = “)”) o (token = “\$”)

entonces

{SENTENCIA NULA}

si no

*ERROR*

fin si

fin si

fin

## Procedimiento F inicio

{SIMULACIÓN DE LA REGLA  $F \rightarrow ( E )$ }

```
si (token = “(” )
  entonces
    avanzar_entrada(token)
    E;
    si (token = “)” )
      entonces
        avanzar_entrada(token)
      si no
        ERROR
    fin si
  si no
```

{SIMULACIÓN DE LA REGLA F → **identificador**}

**si** (token = “**identificador**”)

**entonces**

        avanzar\_entrada(token)

**si no**

        {SIMULACIÓN DE LA REGLA F → **número**}

**si** (token = “**número**”)

**entonces**

                avanzar\_entrada(token)

**si no**

*ERROR*

**fin si**

**fin si**

**fin si**

**fin**

## *Análisis sintáctico descendente predictivo y no recursivo*

{Se apila el delimitador derecho - \$ - y el símbolo inicial – S -}

apilar (\$)

apilar (S)

Token  $\leftarrow$  primer símbolo de la entrada

**mientras** (final-entrada= **FALSO**) **hacer**

**si** Cima  $\in V_T$

**entonces**

**si** (Token = Cima)

**entonces**

**si** (Token = \$)

**entonces** *“La cadena de entrada es aceptada”*

**si no**

{ Token  $\neq$  \$

Se empareja la cima de la pila con el componente léxico de la entrada

}

desapilar (Cima)

avanzar (Token)

**fin si**

**si no**

{Token  $\neq$  Cima}

Llamar a una función de error

**fin si**

**si no**

{Cima  $\notin V_T$ , por tanto Cima  $\in V_N$ }

**si** (M[Cima, Token] = Cima  $\rightarrow X_1 X_2 \dots X_N$ )

**entonces**

{Se extrae el símbolo actual de la cima de la pila}

desapilar (Cima)

{ Se introducen los símbolos de la alternativa en orden inverso }

**para** i = N **hasta** 1 **hacer**

    apilar ( $X_i$ )

**fin para**

**si no**

{Cima  $\in V_N$  y M[Cima, Token] = **vacío** }

Llamar a una función de error

**fin si**

**fin si**

**fin si**

**fin mientras**

## TABLA PREDICTIVA Y EL MODO DE PÁNICO DE RECUPERACIÓN DE ERRORES

	identificador	número	+	*	(	)	\$
<b>E</b>	1	1			1	Sincronización	Sincronización
<b>E'</b>	3*	3*	2	3*	3*	3	3
<b>T</b>	4	4	Sincronización		4	Sincronización	Sincronización
<b>T'</b>	6*	6*	6	5	6*	6	6
<b>F</b>	8	9	Sincronización	Sincronización	7	Sincronización	Sincronización

- **Producciones épsilon**
  - Se han introducido producciones **épsilon** marcadas con **asterisco** en las celdas vacías de los símbolos no terminales que contienen a épsilon en su conjunto primero. Esta acción posterga la detección del error.
- **Sincronización:**
  - Se han utilizado como símbolos de sincronización del símbolo no terminal A a los símbolos de siguiente(A).
  - Se eliminaría el símbolo no terminal de la pila del análisis sintáctico.

## TABLA PREDICTIVA Y EL MÉTODO DE NIVEL DE FRASE

	identificador	número	+	*	(	)	\$
<b>E</b>	1	1	E3	E3	1	E2	E3
<b>E'</b>	3*	3*	2	3*	3*	3	3
<b>T</b>	4	4	E3	E3	4	E2	E3
<b>T'</b>	6*	6*	6	5	6*	6	6
<b>F</b>	8	9	E3	E3	7	E2	E3
<b>identificador</b>	Emparejar						
<b>número</b>		Emparejar					
+			Emparejar				
*				Emparejar			
(					Emparejar		
)	E1	E1	E1	E1	E1	Emparejar	E1
\$	E2	E2	E2	E2	E2	E2	ACEPTAR

- Las producciones **épsilon** marcadas con **asterisco** postergan la detección del error.
- **E1:**
  - **Diagnóstico:** falta el paréntesis derecho.
  - **Posible acción:** extraer el paréntesis derecho de la pila.
- **E2:**
  - **Diagnóstico:** símbolo inesperado.
  - **Posible acción:** eliminar el símbolo de la entrada.
- **E3:**
  - **Diagnóstico:** falta operando.
  - **Posible acción:** se inserta en la entrada “identificador”.