

# **UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA**

Sistemas Informáticos y Computación

*Teoría de Autómatas y Compiladores*

**Autor:** Danilo Alejandro Ochoa Hidalgo.

**Fecha:** lunes, 11/02/2019.

**Correo:** [daochoa6@utpl.edu.ec](mailto:daochoa6@utpl.edu.ec)

## **Definición del Lenguaje**

### **Palabras Reservadas**

Las palabras reservadas tienen su nombre debido a que solo pueden ser usadas en ciertas instancias porque tienen una representación para el programa y cuentan con funciones específicas dentro del mismo:

Tabla 1: *Palabras Reservadas*

<b>Palabra Reservada</b>	<b>Descripción</b>
inicio	Señala el inicio del programa.
fin	Señala el final del programa.
print	Presenta información de salida a una interfaz de consola.
if	Señala el inicio una condición.
then	Señala el cuerpo de una condición en caso de que sea verdadera.
int	Indica el tipo de dato para números enteros.

### **Identificadores**

Los identificadores están conformados por una letra como primer carácter y a continuación pueden concatenarse más caracteres o números, por ejemplo:

- a1
- aa
- Ab
- aB
- a1m2n23j

## Números

Los números quedan definidos cuando tienen un dígito al inicio, y a continuación más dígitos ó un punto seguido de más dígitos, por ejemplo:

- 1
- 11
- 1.1
- 1.11
- 11.1

## Operadores Aritméticos

Indican operaciones de matemática (aritméticas) como:

- Suma: +
- Multiplicación: \*

## Operador de Asignación

Como su nombre lo indica sirve para asignar un valor a un identificador.

- Asignación: =

## Operadores Relacionales

Cuando se quiere saber si existe diferencia entre dos componentes se pueden usar estos operadores para obtener un resultado deseado, por ejemplo:

- ¿Son iguales?: ==
- ¿Son diferentes?: !=

## Restricciones

- Todas las sentencias deben terminar con el carácter “;”.

## Diagrama del Autómata

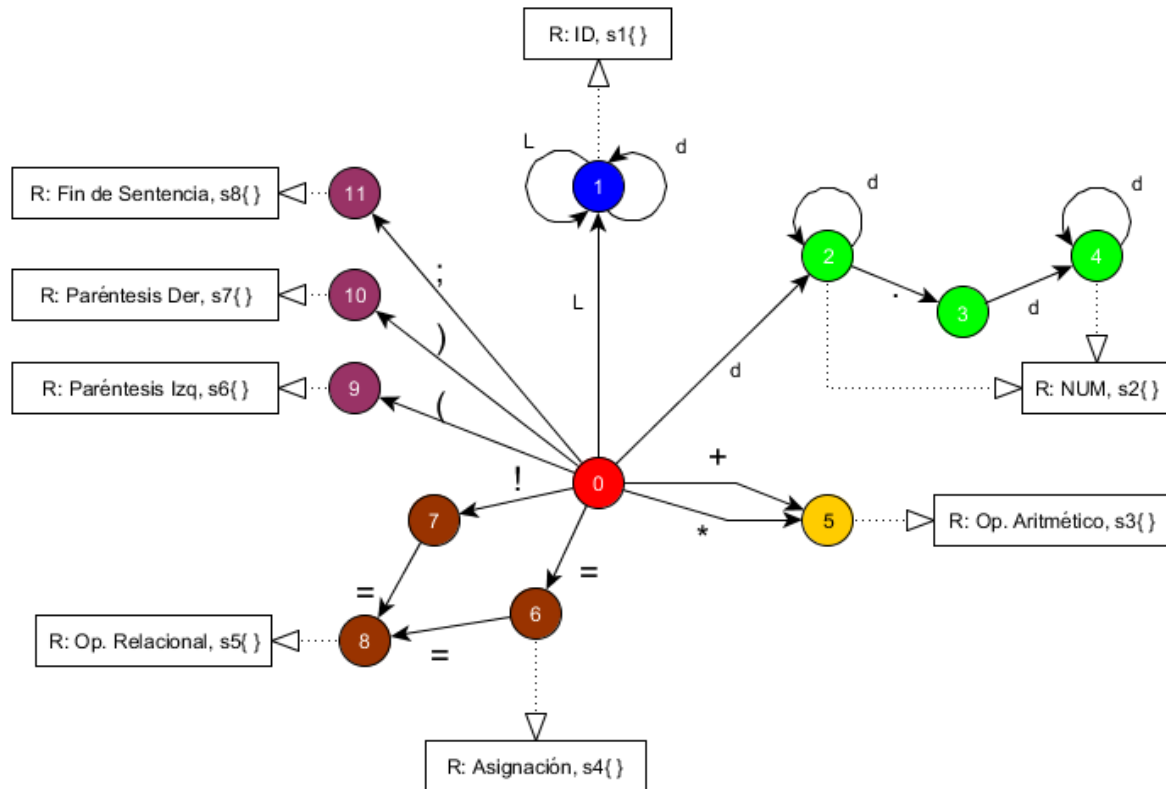


Figura 1: Autómata del Lenguaje Definido.

### Separadores

Son caracteres que ayudan a determinar en qué momento se debe identificar una palabra o carácter tomado desde el código fuente para la construcción de un Token, por ejemplo:

- L -> letra
- d -> dígito
- ( ) -> paréntesis
- + -> más
- \* -> asterisco
- ; -> punto y coma
- ' ' -> espacio
- \n -> salto de línea

Los separadores se utilizan en los Estados Finales por lo que se presentan ocho separadores para:

Tabla 2: Separadores para cada Estado Final del Autómata

Separadores	Símbolos	Función
S1, p. ej.: inicio, a1	+ * = , ! ) ; ' , \n	Palabras Reservadas e Identificadores

S2, p. ej.: 1, 130.5	+ * = , ! ) ; ' , \n	Números
S3, p. ej.: a + b * 2	( ' , \n L	Operadores Aritméticos
S4, p. ej.: a = 12.3;	( ' , L d	Operador de Asignación
S5, p. ej.: if (a == b)	' , L d	Operadores Relacionales
S6, p. ej.: (	' , L d	Paréntesis Izquierdo
S7, p. ej.: )	+ * ; ' , L d	Paréntesis Derecho
S8, p. ej.: ;	' , \n L	Fin de Sentencia

## Analizador Léxico

### Código Fuente Para Ejecutar

```

1  inicio
2  int a = 1;
3  int b = 2;
4  int c = 0;
5  if (a == b) then c = a + b;
6  print c;
7  if (a != b) then c = a * b;
8  print c;
9  fin

```

```

*****
***** INICIALIZADO *****
*****
-----
LINEA 1
-----
Res: inicio
-----
LINEA 2
-----
Res: int
Iden: a
Asig: =
Num: 1
-----
LINEA 3
-----
Res: int
Iden: b
Asig: =
Num: 2
-----
LINEA 4
-----
Res: int
Iden: c
Asig: =
Num: 0
-----
LINEA 5
-----
Res: if
Iden: a
OpRe: ==
Iden: b
Res: then
Iden: c
Asig: =
Iden: a
OpAr: +
Iden: b
-----
LINEA 6
-----
Res: print
Iden: c
-----
LINEA 7
-----
Res: if
Iden: a
OpRe: !=
Iden: b
Res: then
Iden: c
Asig: =
Iden: a
OpAr: *
Iden: b
-----
LINEA 8
-----
Res: print
Iden: c
-----
LINEA 9
-----
Res: fin
*****
***** FINALIZADO *****
*****
Process returned 0 (0x0)   execution time : 0.899 s
Press any key to continue.

```

Figura 2: Ejecución del Código Fuente para Análisis Léxico.

# Analizador Sintáctico

## Producciones (Reglas de la Gramática)

P' -> P  
 P -> inicio B fin  
 B -> S ;  
 B -> S ; B  
 B -> CTRL  
 B -> CTRL B  
 S -> TV UV = VAL  
 S -> UV = VAL  
 S -> print id  
 TV -> int  
 UV -> id  
 VAL -> E  
 E -> E + T  
 E -> T  
 T -> T \* F  
 T -> F  
 F -> ( E )  
 F -> id  
 F -> num  
 CTRL -> if ( COND ) then S ;  
 COND -> id OR id  
 OR -> ==  
 OR -> !=

## Tabla del Analizador Sintáctico

Esta tabla se realiza mediante un proceso que tiene como primera instancia la construcción de los CONJUNTOS DE ELEMENTOS y una tabla con los PRIMEROS Y SIGUIENTES dando como resultado las tablas de ACCION e IR\_A.

Tabla 3: *Primeros y Siguientes*

PRIMEROS Y SIGUIENTES		
No Terminal	Primeros	Siguientes
P'	{inicio}	{}
P	{inicio}	{}
B	{print,if,int,id}	{fin}
S	{print,int,id}	{;}
TV	{int}	{id}
UV	{id}	{=}
VAL	{(,id,num}	{;}
E	{(,id,num}	{;,+,)}
T	{(,id,num}	{;,+*,)}
F	{(,id,num}	{;,+*,)}
CTRL	{if}	{fin,print,if,int,id}
COND	{id}	{)}
OR	{==,!=}	{id}

Tabla 4: *Tabla de Análisis Sintáctico LR*

[illegible]

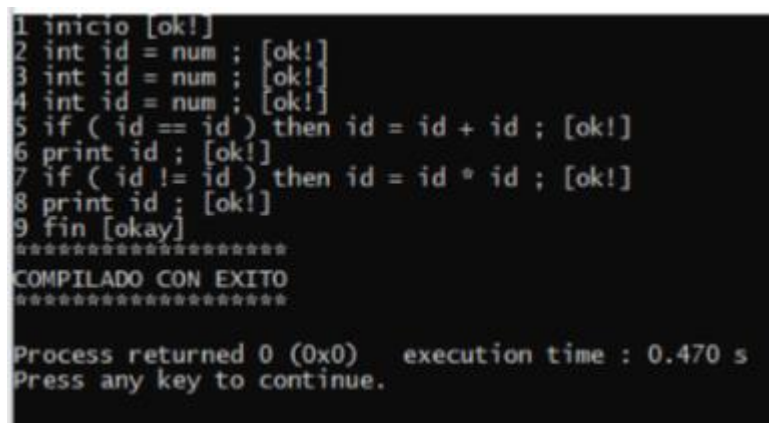
### Código Fuente Para Ejecutar

```

1  inicio
2  int a = 1;
3  int b = 2;
4  int c = 0;
5  if (a == b) then c = a + b;
6  print c;
7  if (a != b) then c = a * b;
8  print c;
9  fin

```

Como resultado se puede observar el correcto análisis del código desarrollado por lectura para cada línea del código y por estados de la pila:

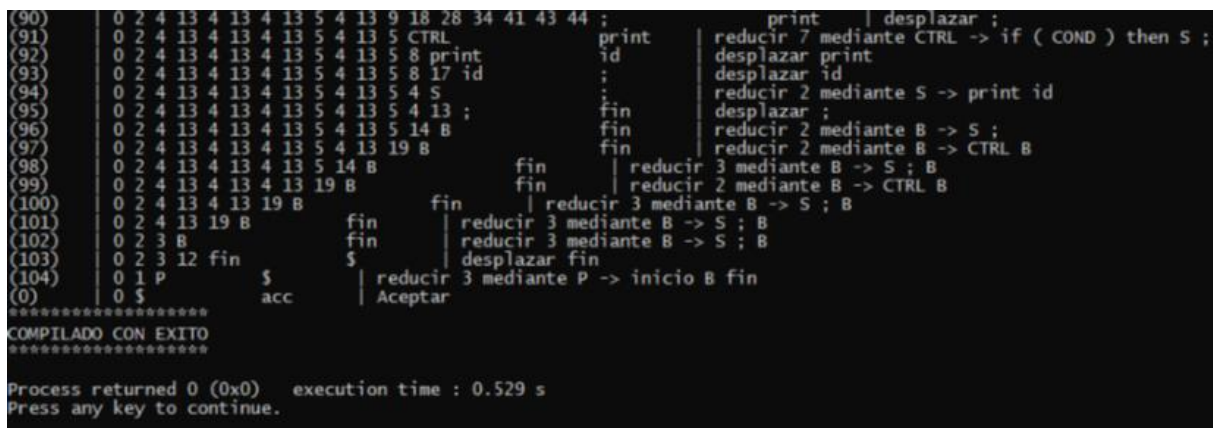


```

1 inicio [ok!]
2 int id = num ; [ok!]
3 int id = num ; [ok!]
4 int id = num ; [ok!]
5 if ( id == id ) then id = id + id ; [ok!]
6 print id ; [ok!]
7 if ( id != id ) then id = id * id ; [ok!]
8 print id ; [ok!]
9 fin [okay]
*****
COMPILADO CON EXITO
*****
Process returned 0 (0x0)   execution time : 0.470 s
Press any key to continue.

```

**Figura 3:** Ejecución del Código Fuente para Análisis Sintáctico por cada línea del código.



```

(90) 0 2 4 13 4 13 4 13 5 4 13 9 18 28 34 41 43 44 ; print | desplazar ;
(91) 0 2 4 13 4 13 4 13 5 4 13 5 CTRL print reducir 7 mediante CTRL -> if ( COND ) then S ;
(92) 0 2 4 13 4 13 4 13 5 4 13 5 8 print id desplazar print
(93) 0 2 4 13 4 13 4 13 5 4 13 5 8 17 id ; desplazar id
(94) 0 2 4 13 4 13 4 13 5 4 13 5 4 S ; reducir 2 mediante S -> print id
(95) 0 2 4 13 4 13 4 13 5 4 13 5 4 13 ; fin desplazar ;
(96) 0 2 4 13 4 13 4 13 5 4 13 5 14 B ; fin reducir 2 mediante B -> S ;
(97) 0 2 4 13 4 13 4 13 5 4 13 19 B ; fin reducir 2 mediante B -> CTRL B
(98) 0 2 4 13 4 13 4 13 5 14 B fin reducir 3 mediante B -> S ; B
(99) 0 2 4 13 4 13 4 13 19 B fin fin reducir 2 mediante B -> CTRL B
(100) 0 2 4 13 4 13 19 B fin | reducir 3 mediante B -> S ; B
(101) 0 2 4 13 19 B fin reducir 3 mediante B -> S ; B
(102) 0 2 3 B fin reducir 3 mediante B -> S ; B
(103) 0 2 3 12 fin $ desplazar fin
(104) 0 1 P $ reducir 3 mediante P -> inicio B fin
(0) 0 $ acc Aceptar
*****
COMPILADO CON EXITO
*****
Process returned 0 (0x0)   execution time : 0.529 s
Press any key to continue.

```

**Figura 4:** Ejecución del Código Fuente para Análisis Sintáctico por Estados de la Pila.