Ingeniería en Sistemas Computacionales

**Lenguajes y Autómatas II**

**Análisis Semántico**

P R E S E N T A N:

Hernández Carmona Cristopher

Muñoz Sánchez René

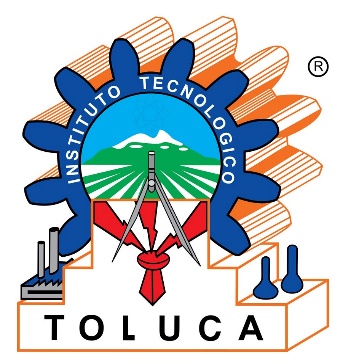
Peña Gómez Luis Ángel

**SEP**

**SNESTM**

**TNM**

**INSTITUTO TECNOLÓGICO DE TOLUCA**



Metepec, Estado de México; septiembre 2017

**Tabla de tokens fijos**

|  |  |  |
| --- | --- | --- |
| *Lexema* | *Nombre* | *Número* |
| inicio | Inicio | 1 |
| ( | Paréntesis que abre | 2 |
| ) | Paréntesis que cierra | 3 |
| { | Llave que abre | 4 |
| } | Llave que cierra | 5 |
| int | Tipo de dato | 6 |
| double |
| string |
|  |
| ; | Punto y coma | 7 |
| = | Igual | 8 |
| read | Leer | 9 |
| put | Imprime | 10 |
| : | Dos puntos | 11 |
| while | Repetición | 12 |
| < | Operador relacional | 13 |
| > |
| <= |
| >= |
| == |
| ¡= |
| and | Operador lógico | 14 |
| or |
| + | Operador aritmético | 15 |
| - |
| \* |
| / |
| , | Coma | 16 |
| for | Para | 17 |
| do | Hacer | 18 |
| # | Inicio/fin de programa | 19 |
| call | Llamada a función | 20 |

Tabla de tipo de dato

|  |  |
| --- | --- |
| *Nombre* | *Número* |
| Variable | 50 |
| Cadena | 51 |
| Número entero | 52 |
| Número doble | 53 |

Tabla de errores

|  |  |
| --- | --- |
| *Nombre* | *Número* |
| Variable incorrecta | 100 |
| Mensaje/Cadena o carácter no valido/a | 101 |
| Número entero erróneo | 102 |
| Número doble erróneo | 103 |
| Símbolo desconocido | 104 |

**Terminales y No Terminales**

**T**:{#, inicio, (,),{,},números, letras, variable, ; , int, double, string, =, read, put, : , while, for, do, opRelacional, opAritmético, opLógico}

**NT**:{inicio, declaraciones, acciones, lectura, escritura, repetición, condición, OM, opción, con´, for, decvar, declaraciones’, salida, for, función, fun’, par, par’}

**Gramática**

<inicio>-------------------> # inicio ( { <sentencias> } <función> #

<sentencias--------------> <declaraciones> | <acciones> | <llamadas> | λ

<acciones>--------------> variable = <acciones'> ; <sentencias> | <escritura> | <repetición>

<acciones'>-------------> <opción> <opmat> <sentencias> | read ( tipoDato ) <sentencias>

<declaraciones>-------> tipoDato variable <declaración’> ; <sentencias>

<declaraciones’>------> , variable <decvar> | λ

<llamadas>-------------> call variable ( <par> ) ; <sentencias>

<salida>------------------> cadena | variable

<escritura>--------------> put : ( <salida> ) ; <sentencias>

<repeticion>-------------> while ( <condición> ) { <sentencias> } <sentencias> | do { sentencias } ( <condicion> ) ; <sentencias>

| for ( <for> ) { <sentencias> } <sentencias>

<for>------------------------> tipoDato variable = <opcion> ; <condición> ; variable = <opcion> <OM>

<condicion>------------------> <opción> opRel <opción> <con'>

<con'>-----------------------> opLog <condicion> | λ

<opcion>---------------------> variable | entero | doble | cadena

<OM>-------------------------> opMat <OM'> | λ

<OM'>------------------------> <opción> <OM>

<funcion>--------------------> variable ( <par> ) { <sentencias> } <fun’> | λ

<fun’>-----------------------> <funcion> | λ

<par>------------------------> tipoDato variable <par’> | λ

<par’>-----------------------> , <par> | λ

<decvar>-----------------> , variable <declaraciones’> | λ

**Producciones**

1. <inicio>-------------------------> # inicio ( { <sentencias> } <función> #
2. <sentencias>-----------------> <declaraciones>
3. <sentencias>-----------------> <acciones>
4. <sentencias->----------------> λ
5. <acciones>-------------------> variable = <acciones'> ; <sentencias>
6. <acciones>-------------------> <escritura>
7. <acciones>-------------------> <repetición>
8. < λ>-----------------------------> λ
9. <acciones'>------------------> <opción> <OM> <sentencias>
10. <acciones'>-------------------> read ( tipoDato ) <sentencias>
11. <OM>---------------------------> opMat <OM'>
12. <OM->--------------------------> λ
13. <escritura>--------------------> put : ( <salida> ) ; <sentencias>
14. <declaracion>----------------> tipoDato variable <declaración’> ; <sentencias>
15. <repeticion>------------------> while ( <condición> ) { <sentencias> } <sentencias>
16. <condicion>------------------> <opción> opRel <opción> <con'>
17. <con'->-------------------------> λ
18. <con'>--------------------------> opLog <condicion>
19. <opcion>-----------------------> variable
20. <opcion>-----------------------> entero
21. <opcion>----------------------> doble
22. <OM'>--------------------------> <opción> <opmat>
23. <funcion>----------------------> variable ( <par> ) { <sentencias> } <fun’>
24. <fun’>---------------------------> <funcion>
25. <fun’>---------------------------> λ
26. <par>---------------------------> tipoDato variable <par’>
27. <par’>---------------------------> , <par>
28. <par’>--------------------------> λ
29. <repeticion2>----------------> for ( <for> ) { <sentencias> } <sentencias>
30. <for> ---------------------------> tipoDato variable = <opcion> ; <condición> ; variable = <opcion> <opmat>
31. <repeticion>------------------> do { sentencias } ( <condicion> ) ; <sentencias>
32. <funcion>---------------------> λ
33. <par>---------------------------> λ
34. <opcion>-----------------------> cadena
35. <declaraciones’>-------------> λ
36. <declaraciones’>-------------> , variable <decvar>
37. <decvar>----------------------->, variable <declaraciones’>
38. <decvar>------------------> λ
39. <sentencias>-----------------> <llamada>
40. <llamada>---------------------> call variable ( <par> ) ; <sentencias>
41. <salida>-----------------------> cadena
42. <salida>-----------------------> variable

**Tabla de transiciones**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Producción** | **1** | **3** | **5** | **6** | **7** | **9** | **10** | **12** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **50** | **51** | **52** | **53** |
| s0 | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| <inicio> | 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| <declaraciones> |  |  |  | 14 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| <acciones> |  |  |  |  |  |  | 6 | 7 |  |  |  | 7 | 7 |  |  | 5 |  |  |  |
| <acciones’> |  |  |  |  |  | 10 |  |  |  |  |  |  |  |  |  | 9 | 9 | 9 | 9 |
| <sentencias> |  |  | 4 | 2 | 4 |  | 3 | 3 |  |  |  | 3 | 3 |  |  | 3 |  |  |  |
| <escritura> |  |  |  |  |  |  | 13 |  |  |  |  |  |  |  |  |  |  |  |  |
| <repetición> |  |  |  |  |  |  |  | 15 |  |  |  | 29 | 31 |  |  |  |  |  |  |
| <condición> |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 16 |  | 16 | 16 |
| <con’> |  | 17 |  |  | 17 |  |  |  | 18 |  |  |  |  |  |  |  |  |  |  |
| <opción> |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 19 | 34 | 20 | 21 |
| <op mat> |  | 12 |  | 12 | 12 |  | 12 | 12 |  | 11 |  |  |  |  |  | 12 |  |  |  |
| <op mat’> |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 22 |  | 22 | 22 |
| <función> |  |  |  |  |  |  |  |  |  |  |  |  |  | 32 |  | 23 |  |  |  |
| <fun'> |  |  |  |  |  |  |  |  |  |  |  |  |  | 25 |  | 24 |  |  |  |
| <par> |  | 33 |  | 26 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| <par'> |  | 28 |  |  |  |  |  |  |  |  | 27 |  |  |  |  |  |  |  |  |
| <for> |  |  |  | 30 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| <declaraciones’> |  |  |  |  | 35 |  |  |  |  |  | 36 |  |  |  |  |  |  |  |  |
| <decvar> |  |  |  |  | 38 |  |  |  |  |  | 37 |  |  |  |  |  |  |  |  |
| <llamada> |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 40 |  |  |  |  |
| <salida> |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 42 | 41 |  |  |

**Primeros y siguientes**

|  |  |  |
| --- | --- | --- |
| *No Terminal* | *Primeros* | *Siguientes* |
| <S0> | # | $ |
| <inicio> | # | $ |
| <sentencias> | variable put while tipoDato for do call λ | } ; |
| <declaraciones> | tipoDato | } ; variable put while tipoDato for do |
| <acciones> | variable put while for do | } ; |
| <escritura> | put | } ; |
| <repetición> | while do for | } ; |
| <acciones’> | variable int double string read | ; |
| <opción> | variable int double string | ) ; opMat and opRel put while tipoDato |
| <OM> | opMat λ | ) ; variable put while tipoDato |
| <condición> | variable int double string | ) ; |
| <con’> | and or λ | ) ; |
| <OM’> | variable int double string | ; variable put while tipoDato |
| <funcion> | variable λ | } ; # |
| <fun’> | variable λ | # |
| <par> | tipoDato λ | ) |
| <par’> | , λ | ) |
| <for> | tipoDato | ) ; |
| <declaraciones’> | , λ | ; |
| <decvar> | , λ | ; |
| <llamadas> | call | } ; |
| <salida> | variable cadena | ) |

**Ejemplos de programa**

#

inicio() {

int x, y, z;

double z;

x = 2 + 4;

y = 3.14;

call funcionA();

call funcionB(int x, double y);

}

funcionA() {

int y;

y = read(int);

put:("CHUX");

}

funcionB(int x, double y) {

int w;

double z;

string a, b;

w = "CHUX";

put:(x);

while(x > y or y < z) {

z = z + 1;

}

do {

y = y + 1;

} while(x > y and y < z);

for(int i = 1; z < x; i = i + 1) {

z = z + 1;

}

}

#

#

inicio() {

int x, y, z;

double z;

x = 2 + 4 \* 2;

y = 3.14;

z = "cad";

call funcionB(int x, double y);

}

funcionB(int x, double y) {

string a;

w = "TEST";

put:(x);

}

#

**Tabla de errores**

|  |  |
| --- | --- |
| *Número* | *Error* |
| 201 | Variable no declarada |
| 202 | Variable no utilizada |
| 203 | Variable duplicada |
| 204 | Tipo de dato incompatible |

**Tabla de símbolos**

|  |  |  |  |
| --- | --- | --- | --- |
| *Lexema* | *Tipo de dato* | *Valor* | *Error* |
| a | int | 3 | 201 |

**Tabla de tipos de dato**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Tipo* | *int* | *double* | *boolean* | *string* | *char* |
| *Int* | int | double |  | string | string |
| *double* | double |  |  | string | string |
| *boolean* |  |  | boolean |  |  |
| *string* | string | string |  | string | string |
| *char* | string | string |  | string | string |

**Rutinas semánticas**

1. <inicio>-------------------------> # inicio ( { <sentencias>.código } <función>.código #

2. <sentencias>-----------------> <declaraciones>.código

3. <sentencias>-----------------> <acciones>.código

4. <sentencias->----------------> avanzar

5. <acciones>-------------------> variable = <acciones'>.código ; <sentencias>.código

// Buscar variable en T.S. // Crear pilas e insertar variable en pila1 //

//Errores: variable no declarada

6. <acciones>-------------------> <escritura>.código

7. <acciones>-------------------> <repetición>.código

8. < λ>-----------------------------> avanzar

9. <acciones'>------------------> <opción>.código <OM>.código <sentencias>.código

10. <acciones'>-------------------> read ( tipoDato ) <sentencias>.código

11. <OM>---------------------------> opMat <OM'>.código

12. <OM->--------------------------> avanzar

Vaciar pila 2 a pila 1 y evaluar expresión.

13. <escritura>--------------------> put : ( <salida>.código ) ; <sentencias>.código

14. <declaracion>----------------> tipoDato variable <declaración’>.código ; <sentencias>

//Agregar tipo de dato a T.S. // Checar tipo de dato válido // Insertar tipo de dato a pila1 // Agregar id variable a T.S // Checar id variable en T.S. // Poner propiedad *isDuplicada* = false

//Errores: variable duplicada

15. <repeticion>------------------> while ( <condición>.código ) { <sentencias>.código } <sentencias>

16. <condicion>------------------> <opción>.código opRel <opción>.código <con'>.código

17. <con'->-------------------------> avanzar

18. <con'>--------------------------> opLog <condicion>.código

19. <opcion>-----------------------> variable

20. <opcion>-----------------------> entero

21. <opcion>----------------------> doble

22. <OM'>--------------------------> <opción>.código <opmat>.código

23. <funcion>----------------------> variable ( <par>.código ) { <sentencias>.código } <fun’>.código

24. <fun’>---------------------------> <funcion>.código

25. <fun’>---------------------------> avanzar

26. <par>---------------------------> tipoDato variable <par’>.código

27. <par’>---------------------------> , <par>.código

28. <par’>--------------------------> λ

29. <repeticion2>----------------> for ( <for>.código ) { <sentencias>.código } <sentencias>

30. <for> ---------------------------> tipoDato variable = <opcion>.código ; <condición>.código ; variable = <opcion> <opmat>

31. <repeticion>------------------> do { <sentencias>.código } ( <condicion>.código ) ; <sentencias>

32. <funcion>---------------------> avanzar

33. <par>---------------------------> avanzar

34. <opcion>-----------------------> cadena

35. <declaraciones’>-------------> avanzar

36. <declaraciones’>-------------> , variable <decvar>.código

37. <decvar>----------------------->, variable <declaraciones’>.código

38. <decvar>------------------> avanzar

39. <sentencias>-----------------> <llamada>.código

40. <llamada>---------------------> call variable ( <par>.código ) ; <sentencias>.código

41. <salida>-----------------------> cadena

42. <salida>-----------------------> variable