Universidad de San Carlos de Guatemala

Organización de lenguajes y compiladores 1

Primer Semestre 2021

Diego Alejandro Sierra García 201903969

**Gramática**

**Tokens Lexicos**

Los terminales se utilizan para denotar todos los elementos y cadenas validas en el analizador léxico. Estos son:

|  |  |  |  |
| --- | --- | --- | --- |
| Token | Valor | Token | Valor |
| Incremento | ++ | Decremento | -- |
| Igualdad | == | Diferencia | != |
| MenorIgual | <= | MayorIgual | >= |
| Menor | < | Mayor | > |
| Or | || | And | && |
| Not | ! | PuntoComa | ; |
| DosPuntos | : | Coma | , |
| ParentesisIzquierdo | ( | ParentesisDerecho | ) |
| CorecheteIzquierdo | [ | CorcheteDerecho | ] |
| LlaveIzquierda | { | LlaveDerecha | } |
| Interrogacion | ? | Igual | = |
| Mas | + | Menos | - |
| Por | \* | Dividido | / |
| Potencia | ^ | Modulo | % |
| IdentificadorInt | Int | IdentificadorDouble | Double |
| IdentificadorBool | Booleano | IdentificadorChar | Char |
| IdentificadorString | String | If | If |
| Switch | Switch | Else | Else |
| Case | Case | Default | Default |
| Break | Break | While | While |
| For | For | Do | Do |
| Continue | Continue | Return | Return |
| Void | Void | True | True |
| False | False | New | New |
| List | List | Add | Add |
| Print | Print | Tolower | Tolower |
| Toupper | Toupper | Lenght | Lenght |
| Truncate | Truncate | Round | Round |
| Typeof | Typeof | Tostr | Tostring |
| Tocharray | Tochararray | Exec | Exec |
| Decimal | [0-9]+("."[0-9]+) | Entero | [0-9]+ |
| Identificador | ([a-zA-Z])[a-zA-Z0-9\_]\* | Char | \'[^\']\' |
| Cadena | ["\""]([^"\""])\*["\""] |  |  |

**No Terminales**

Los no terminales se utilizan en la gramática para denotar las maneras en las que se puede producir el texto, los no terminales tienen la función de variables y no toman un valor especifico, si no que son recursivos. Estos son:

|  |  |  |  |
| --- | --- | --- | --- |
| Inicio | InstruccionGlobal | Tipo | Exp |
| Casteo | InstruccionReturn | ElementoGlobal | Declaracion |
| Asignacion | ListaAgregar | Metodo | Funcion |
| Llamada | Instrucciones | ElementoInstruccion | DefIf |
| DefSwitch | DefWhile | DefFor | DefDoWhile |
| Imprimir | Vectores | ListaValores | Listas |
| InsToCharArray | Cases | ListaParametros | ListaLlamada |
| InstToLower | InsToUpper | InsLenght | InsTruncate |
| InsRound | InsType | InsToString |  |

**Producciones**

La primera producción establece el inicio del código, y el final del archivo.

Inicio -> InstruccionGlobal EndOfFile

La producción InstruccionGlobal y ElementoGlobal indica la forma en la que puede venir el texto antes plano fuera de alguna instrucción especifica.

InstruccionGlobal -> InstruccionGlobal ElementoGlobal

| ElementoGlobal

ElementoGlobal -> Declaracion

|Asignacion PuntoComa

|ListaAgregar

|Metodo

|Funcion

|Exec Llamada PuntoComa

La producción Instrucciones y ElementoInstruccion indica la forma en la que puede venir el texto dentro de alguna instrucción especifica

Instrucciones -> Instrucciones ElementoInstrucciones

| ElementoInstrucciones

ElementoInstrucciones -> Declaracion

|Asignacion PuntoComa

|DefIf

|DefSwitch

|DefWhile

|DefFor

|DefDoWhile

|Break PuntoComa

|Continue PuntoComa

|Return PuntoComa

|Return Exp PuntoComa

|imprimir

|ListaAgregar

|Llamada PuntoComa

La Produccion Exp denota la forma en la que se puede construir una expresión para asignaciones u operaciones.

Exp -> Exp Interr Exp Dospuntos Exp

|Exp Igualdad Exp

|Exp Diferente Exp

|Exp Menor Exp

|Exp Menorigual Exp

|Exp Mayor Exp

|Exp Mayorigual Exp

|Exp Or Exp

|Exp And Exp

|Exp Mas Exp

|Exp Menos Exp

|Exp Por Exp

|Exp Dividido Exp

|Exp Potencia Exp

|Exp Modulo Exp

|Parizq Exp Parder

|Menos Exp

|Not Exp

|Exp Suma2

|Exp Resta2

|Casteo

|Entero

|Decimal

|Char

|Cadena

|True

|False

|Identificador

|Instretur

|Identificador Corizq Exp Corder

|Identificador Corizq Corizq Exp Corder Corder

La producción Identificador muestra los tipos de datos que se pueden declarar o identificador.

Tipo -> Idenentero

|Idendouble

|Idenbool

|Idenchar

|Idenstring

La producción Declaracion y Asignacion demuestran la manera en la que pueden venir estas instrucciones

Declaracion -> Tipo Identificador Igual Exp Ptcoma

|Tipo Identificador Ptcoma

Asignacion -> Identificador Igual Exp

|Identificador Suma2

|Identificador Resta2

|Identificador Corizq Exp Corder Igual Exp

|Identificador Corizq Corizq Exp Corder Corder Igual Exp

Las siguientes producciones denotan la manera en la que pueden venir las secuencias de control y sentencias cíclicas

Defif -> Defif Else If Parizq Exp Parder Llavizq Instrucciones Llavder

|Defif Else Llavizq Instrucciones Llavder

|If Parizq Exp Parder Llavizq Instrucciones Llavder

Defswitch -> Switch Parizq Exp Parder Llavizq Cases Llavder

Cases -> Cases Default Dospuntos Instrucciones

|Cases Case Exp Dospuntos Instrucciones

|Default Dospuntos Instrucciones

|Case Exp Dospuntos Instrucciones

Defwhile -> While Parizq Exp Parder Llavizq Instrucciones Llavder

Deffor -> For Parizq Declaracion Exp Ptcoma Asignacion Parder Llavizq Instrucciones Llavder

|For Parizq Asignacion Ptcoma Exp Ptcoma Asignacion Parder Llavizq Instrucciones Llavder

Defdowhile -> Do Llavizq Instrucciones Llavder While Parizq Exp Parder Ptcoma

Las siguientes producciones denota como se puede construir un método o una función con sus parametros.

Metodo -> Void Identificador Parizq Listapar Parder Llavizq Instrucciones Llavder

|Void Identificador Parizq Parder Llavizq Instrucciones Llavder

Funcion -> Tipo Identificador Parizq Listapar Parder Llavizq Instrucciones Llavder

|Tipo Identificador Parizq Parder Llavizq Instrucciones Llavder

Listapar -> Listapar Coma Tipo Identificador

|Tipo Identificador

Finalmente, están las funciones definidas del lenguaje.

Imprimir -> Print Parizq Exp Parder Ptcoma

Instlower -> Tolower Parizq Exp Parder

Instupper -> Toupper Parizq Exp Parder

Inslength -> Length Parizq Exp Parder

Instruncate -> Truncate Parizq Exp Parder

Insround -> Round Parizq Exp Parder

Instype -> Typeof Parizq Exp Parder

Instostr -> Tostr Parizq Exp Parder