Gramatica Regular (Javascript)

# Terminales =

“{” “}” “+” “-” “\*” “/” “(” “)” “<” “>” “=” “>=” “<=” “==” “!=” “!” “;” “,” “//” “/\*” “\*/” “:” “.” “&&” “||” “&” ”|” “[” “]” “%” “^” “?” “int” “double” “boolean” “char” “string” “new” “dynamiclist” “append” “getvalue” “setvalue” “if” “else” “switch” “default” “case” “break” “while” “for” “do” “continue” “return” “void” “writeline” “tolower” “toupper” “length” “truncate” “round” “typeof” “tostring” “tochararray” “start” “with” “true” “false”

“Entero” = [0-9]+,

“Numero” = [0-9]+("."[0-9]+)?\b,

“ID” = ([a-zA-Z\_])[a-zA-Z0-9\_]\*,

“Cadenas” = \"[^\"]\*\"

“Carácter” = \'[^\"]?\'

“Carácterentero” = [0-9]

“CarácterAlfabeto” = ([a-zA-Z\_])

Todo = Todos los terminales anteriores

# No Teminales = Inicio, LlavesAp, LlavesMet, LlavesFN, Opcion, Instrucción, Consola, Sentencias, SentenciaIf-Else, IfElseIf, SentenciaFor, SentenciaWhile, Comando, SentenciaSwitch, Casos, Elegir, Llamada, Metodos, Variables, PalabrasReservadas, FinalLinea, Carácter, Direccion2, Comentario, Alfabeto, Numero, Operaciones, Expresiones, Logicos, Relacionales

No terminal Inicial = S

# Producciones:

S ::= SENTENCIAS EOF ;

SENTENCIAS ::= SENTENCIAS SENTENCIA

| SENTENCIA ;

SENTENCIA ::= VARIABLES

|COMENTARIOS

|VECTORES

|LISTAS

|IF

|SWITCH

|BREAK “;”

|WHILE

|FOR

|INCRE\_DECRE “;”

|DO\_WHILE “;”

|CONTINUE “;”

|RETURN “;”

|FUNCIONES

|METODOS

|WRITE “;”

|START “;”

|TERNARIO “;”

|DECLARACION “;”

|DECLA\_VECTOR “;”

|APPENDLISTA “;”

|GETVAL “;”

|SETVAL “;”

|LLAMADA “;” ;

VARIABLES ::= VARINT “;”

|VARDOUBLE “;”

|VARBOOLEAN “;”

|VARCHAR “;”

|VARSTRING “;” ;

VECTORES ::= VECTORINT “;”

|VECTORSTRING “;”

|VECTORCHAR “;”

|VECTORBOOLEAN “;”

|VECTORDOUBLE “;” ;

LISTAS ::= LISTAINT “;”

|LISTASTRING “;”

|LISTACHAR “;”

|LISTABOOLEAN “;”

|LISTADOUBLE “;” ;

FUNCIONES ::= “int” “ID” PARAMETROS BLOQUE

|”string” “ID” PARAMETROS BLOQUE

|”char” “ID” PARAMETROS BLOQUE

|”boolean” “ID” PARAMETROS BLOQUE

|”double” “ID” PARAMETROS BLOQUE ;

METODOS ::= “void” “ID” PARAMETROS BLOQUE ;

PARAMETROS ::= “(” TIPOPARAMETRO “)”

|”(” “)” ;

TIPOPARAMETRO ::= TIPOPARAMETRO “,” TIPO “ID”

|TIPO “ID” ;

TIPO ::= “int”

|”string”

|”char”

|”boolean”

|”double” ;

LLAMADA ::= “ID” “(“ “)”

|”ID” “(“ PARALLAMADA “)” ;

PARALLAMADA ::= PARALLAMADA “,” EXP

|EXP ;

COMENTARIOS ::= “/\*” “Todo” “\*/”;

VECTORINT ::= “int” “ID” “[” “]” “=” VECINT;

VECINT ::= “new” “int” “[” “Entero” “]”

|”{” VALORESINT “}”;

VALORESINT ::= VALORESINT “,” EXP

| EXP;

VECTORSTRING ::= “string” “ID” “[” “]” ”=” VECSTRING;

VECSTRING ::= “new” “string” “[” “Entero” “]”

|”{” VALORESSTRING “}” ;

VALORESSTRING ::= VALORESSTRING “,” EXP

| EXP ;

VECTORCHAR ::= “char” “ID” “[” “]” “=” VECCHAR;

VECCHAR ::= “new” “char” “[” “Entero” “]”

|”{” VALORESCHAR “}”;

VALORESCHAR ::= VALORESCHAR “,” EXP

| EXP ;

VECTORBOOLEAN ::= “boolean” “ID” “[” “]” “=” VECBOOLEAN;

VECBOOLEAN ::= “new” “boolean” “[” “Entero” “]”

|”{” VALORESBOOLEAN “}” ;

VALORESBOOLEAN ::= VALORESBOOLEAN “,” EXP

| EXP;

VECTORDOUBLE ::= “double” “ID” “[” “]” “=” VECDOUBLE;

VECDOUBLE ::= “new” “double” “[” “Entero” “]”

|”{” VALORESDOUBLE “}” ;

VALORESDOUBLE ::= VALORESDOUBLE “,” EXP

| EXP;

VARINT ::= “int” “ID” VARIABLEINT ;

VARIABLEINT ::= “,” “ID” VARIABLEINT

| “ID”

| “=” EXP

| “=” “(” “int” “)” EXP

| “=” “length” “(” EXP “)”

| “=” “truncate” “(” EXP “)” ;

DECI ::= “Numero”

|”ID” ;

VARDOUBLE ::= “double” “ID” VARIABLEDOUBLE ;

VARIABLEDOUBLE ::= “,” “ID” VARIABLEDOUBLE

| “ID”

| “=” EXP

| “=” “(” “double” “)” EXP

|”=” “round” “(” EXP “)”;

VARBOOLEAN ::= “boolean” “ID” VARIABLEBOOLEAN ;

VARIABLEBOOLEAN ::= “,” “ID” VARIABLEBOOLEAN

| “P\_ID”

| “=” EXP ;

TRUEFALSE ::= “true”

|”false” ;

VARSTRING ::= “string” “ID” VARIABLESTRING ;

VARIABLESTRING ::= “,” “ID” VARIABLESTRING

| “ID”

| “=” EXP

| “=” MAYMEN;

MAYMEN ::= “tolower” “(” EXP “)”

|”toupper” “(” EXP “)”

|”typeof” “(” EXP “)”

|”tostring” “(” EXP “)”;

TEXT ::= “Numero”

|TRUEFALSE ;

VARCHAR ::= “char” “ID” VARIABLECHAR ;

VARIABLECHAR ::= “,” “ID” VARIABLECHAR

| “ID”

| “=” EXP

| “=” “(” “char” “)” EXP;

PalabrasReservadas ::= VAR VarCaracter:var IGUAL Caracter:car FinalLinea:fin

| LET VarCaracter:var IGUAL Caracter:car FinalLinea:fin

| CONST VarCaracter:var IGUAL Caracter:car FinalLinea:fin

| VarCaracter:var IGUAL Caracter:car FinalLinea:fin

;

FinalLinea ::= PUNTOYCOMA Instruccion:ins

| Instruccion:ins

;

Caracter ::= Operaciones:exp

| REQUIRE:req PAR1:pa1 CADENAS:cad PUNTO:p1 PUNTO:p2 Direccion2:dic2 PAR2:pa2

;

Direccion2 ::= VariablesVar:alf

| VariablesVar:alf Direccion2:dic2

| DIVIDIDO:div Direccion2:dic2

;

Comentario ::= Alfabeto:alf Comentario:com

| Numero:num Comentario:com

| Alfabeto:alf

| Numero:num

| SIGRESERVA:num Comentario:com

| SIGRESERVA:num

;

Alfabeto ::= LETRAS:let

| ENTEROS2:ent

| COMBINACION:com

;

Numero ::= ENTEROS:ent

| DECIMALES:dec

;

Operaciones ::= Expresiones:exp

| Logicos:log

| Relacionales:res

| Numero:exp1

| Alfabeto:exp1

| CADENAS:cad VariablesVar:var

| PAR1:p1 Operaciones:op PAR2:p2

;

Expresiones ::= Operaciones:op1 MAS:m Operaciones:op2

| Operaciones:op1 MENOS:m Operaciones:op2

| Operaciones:op1 POR:m Operaciones:op2

| Operaciones:op1 DIVIDIDO:m Operaciones:op2

| Operaciones:op1 POR:m POR:n Operaciones:op2

| Operaciones:op1 PORCENTAJE:m Operaciones:op2

| Operaciones:op1 MAS:m MAS:n

| Operaciones:op1 MENOS:m MENOS:n

| MENOS:m Operaciones:op

;

Logicos ::= Operaciones:op1 AND:m Operaciones:op2

| Operaciones:op1 OR:m Operaciones:op2

| NOT:m Operaciones:op1

;

Relacionales ::= Operaciones:op1 IGUAL:m IGUAL:n Operaciones:op2

| Operaciones:op1 NOT:m IGUAL:n Operaciones:op2

| Operaciones:op1 MENOR:m Operaciones:op2

| Operaciones:op1 MAYOR:m Operaciones:op2

| Operaciones:op1 MENOR:m IGUAL:n Operaciones:op2

| Operaciones:op1 MAYOR:m IGUAL:n Operaciones:op2

;