Lenguaje FCA

letra = [a-zA-Z]

numero = [0-9]+

id ={letra}({letra}|"_"|[0-9])* simbolo

Simbolo =[!-}]

Terminales: String llaveiz, llaveder, puntocoma, pariz, parder, coma, igual, dpuntos, coriz, corder, dolar, definirglobales, generarreporteestadistico, compare, stringvar, doublevar, graficabarras, titulo, ejex, valores, titulox, tituloy, graficapie, graficalineas, archivo, puntajeespecifico, decimal, id, cadena, puntajegeneral, entero;

No Terminales: INICIO, INSTRUCCION, BLOQUEGLOBALES, BLOQUEARCHIVOS, BLOQUEGBARRAS, BLOQUEGPIE, BLOQUEGLINEAS, CUERPOGLOBALES, VARSTRING, VARDOUBLE, CUERPOGBARRAS, BTITULO, BEJEX, BVALORES, BTITULOX, BTITULOY, ARREGLO, ARREGLO2, ARREGLOV, BPUNTAJE, ARREGLOV2, CUERPOGPIE, CUERPOGLINEAS, BARCHIVO

inicio ::= bloqueglobales

| bloquearchivos

| bloquegbarras

| bloquegpie

| bloqueglineas

| instrucciones

bloqueglobales ::= definirglobales { cuerpoglobales }

cuerpoglobales ::= varstring

| vardouble

| cuerpoglobales

varstring ::= string id ::= cadena ;

vardouble ::= double id = numero ;

bloquearchivos ::= generarreporteestadistico { curepoarchivos }

cuerpoarchivos ::= compare (cadena , cadena) ;

bloquegbarras ::= graficabarras { cuerpogbarras }

cuerpogbarras ::= btitulo bejex bvalores btitulox btituloy

btitulo ::= titulo : id ;

| titulo : cadena ;

bejex ::= ejex [arreglo] ;

arreglo ::= cadena arreglo2

| id arreglo2

arreglo2 ::= , cadena

| , id

| arreglo2

bvalores ::= valores : [arreglov] ;

arreglov ::= cadena arreglov2

| id arreglov2

| bpuntajeespecifico

bpuntajeespecifico :: $ { puntajeespecifico , cadena , cadena , cadena }

arreglov2 ::= , cadena

| , id

| , bpuntajeespecifico

| arreglo2

btitulox ::= titulox : cadena ;

| titulox : id ;

btituloy ::= tituloy : cadena ;

| tituloy : id ;

bloquegpie ::= graficapie { cuerpogpie }

cuerpogpie ::= btitulo bejex bvalores

bloqueglineas ::= graficalineas { cuerpoglineas }

cuerpoglineas ::= btitulo barchivo

barchivo ::= archivo : id ;

| archivo : cadena ;

Lenguaje JavaScript

letra = [a-zA-Z]

numero = [0-9]+

id ={letra}({letra}|"_"|[0-9])* simbolo

Simbolo =[!-}]

Terminales: entero, decimal, id, cadena, igual, puntocoma, var, let, constvar, truevar, falsevar, ifpr, pariz, parder, llaveiz, llaveder, menorq, elsepr, imprimir, diferente, forpr, mas, menos, whilepr, dopr, switchpr, casepr, breakpr, defaultpr, dpuntos, mayorq, andpr, orpr, notpr, por, pot, div, porcentaje, menorigual, mayorigual, digual, coma, requirepr, classpr;

No Terminales: INICIO, INSTRUCCION, EXPRESION, VARIABLES, PRVARIABLES, VALOR, IF, CONDICION, CUERPOIF, RELACIONAL, ELSE, CUERPOELSE, PRINTCONSOLE, ELSEIF, CUERPOELSEIF, FOR, SETTINGFOR, CUERPOFOR, INICIOFOR, INCFOR, ASIGNACION, WHILE, DOWHILE, CUERPOWHILE, SWITCH, CUERPOSWITCH, FINALSWITCH, CUERPOCASE, LOGICO, INSTRUCCIONL, INC, DEC, CALLM, CUERPOMETODO, CUERPOMETODO2, IMPORT, CLASS, CUERPOCLASE, METODO, METODOBODY

INICIO ::= INSTRUCCION

| INSTRUCCION ;

INSTRUCCION ::= VARIABLES

| IF

| FOR

| ASIGNCACION

| WHILE

| DOWHILE

| SWITCH

| LLAMADAM

| INSTRUCCION

VARIABLES ::= PRVARIABLE id igual VALORVARIABLE

PRVARIABLE ::= var

| let

| const

VALORVARIABLE ::= entero

| decimal

| cadena

| true

| false

| IMPORT

IF ::= if (CONDICION) { CUERPOIF }

| if (CONDICION) { CUERPOIF } ELSE

| if (CONDICION) { CUERPOIF } ELSEIF

| if (CONDICION) { CUERPOIF } ELSEIF ELSE

CONDICION ::= RELACIONAL

RELCIONAL ::= EXPRESION == EXPRESION

| EXPRESION != EXPRESION

| EXPRESION < EXPRESION

| EXPRESION > EXPRESION

| EXPRESION <= EXPRESION

| EXPRESION >= EXPRESION

EXPRESION :: id

| entero

| decimal

| cadena

| true

| false

| EXPRESION + EXPRESION

| EXPRESION - EXPRESION

| EXPRESION * EXPRESION

| EXPRESION / EXPRESION

| EXPRESION ** EXPRESION

| EXPRESION % EXPRESION

| - EXPRESION

| (EXPRESION)

| RELACIONAL

| LOGICO

CUERPOIF ::= INSTRUCCION

| break ;

ELSE ::= else { CUERPOELSE }

CUERPOELSE ::= INSTRUCCION

| break ;

ELSEIF ::= CUERPOELSEIF

| ELSEIF

CUERPOELSEIF ::= else if (CONDICION) { CUERPOIF }

FOR ::= for (SETTINGSFOR) { CUERPOFOR }

SETTINGSFOR ::= INICIOFOR ; RELACIONAL ; INCFOR

INICIOFOR ::= VARIABLES

| ASIGNACION

ASIGNACION ::= id = EXPRESION

INCFOR ::=id mas mas

| id menos menos

CUERPOFOR ::= INSTRUCCION

| break ;

WHILE ::= while (RELACIONAL) { CUERPOWHILE }

CUERPOWHILE ::= INSTRUCCION

| break ;

DOWHILE ::= do { CUERPOWHILE } while (RELACIONAL)

SWITCH ::= switch (id) { CUERPOSWITCH FINALSWITCH }

CUERPOSWITCH ::= case EXPRESION : CUERPOCASE break ;

| CUERPOSWITCH

CUERPOCASE ::= INSTRUCCION

FINALSWITCH ::= default : CUERPOCASE

LOGICO ::= EXPRESION && EXPRESION

| EXPRESION || EXPRESION

| ! EXPRESION

LLAMADAM ::= id ( )

| id (CUERPOMETODO)

CUERPOMETODO ::= EXPRESION

| EXPRESION CUERPOMETODO2

CUERPOMETODO2 ::= , EXPRESION

| CUERPOMETODO2

IMPORT ::= require (cadena)

CLASS ::= class id () { CUERPOCLASE }

CUERPOCLASE ::= INSTRUCCION

METODO ::= id () { CUERPOMETODO }

| id (PARAMETRO) { CUERPOMETODO }

PARAMETRO ::= EXPRESION

| EXPRESION PARAMETRO2

PARAMETRO2 ::= , EXPRESION

| PARAMETR**O2**