Laboratorio de Organización de Lenguajes y Compiladores 1, Sección C.

PROYECTO 2 MFMSCRIPT

GRAMATICA

FECHA: 04/11/2022

Juan Josue Zuleta Beb
Carné: 202006353

GRAMATICA

```
%
start INICIO

%%

//INICIO DE LA GRAMATICA
INICIO
  : INSTRUCCIONES EOF
;

//INSTRUCCIONES RECURSIVAS
INSTRUCCIONES
  : INSTRUCCIONES INSTRUCCION
  | INSTRUCCION
;

//TIPOS DE INSTRUCCIONES
INSTRUCCIÓN
  : DEC_VARIABLE
  | DEC_FUNCION
  | DEC_ARREGLOS
  | ASIGNACION
  | ARRAY_PUSH
  | ARRAY_POP
  | PRINT
  | PRINTLN
  | SENTENCIA_IF
  | SWITCH
  | BREAK
  | RETURN
  | CONTINUE
  | WHILE
  | DO_WHILE
  | FOR
  | FOR_OF
  | FOR_IN
  | LLAMADA_FUNCION
  | RUN_FUNCION
  | MAS_MENOS
  | ARRAY_LENGTH
;

//TIPOS DE EXPRESIONES
EXPRESION
  : menos EXPRESION
  | EXPRESION mas EXPRESION
  | EXPRESION menos EXPRESION
  | EXPRESION por EXPRESION
  | EXPRESION div EXPRESION
  | EXPRESION mod EXPRESION
  | EXPRESION potencia EXPRESION
  | id mas_mas
  | id menos_menos
  | par_izq EXPRESION par_der
```

```
  | EXPRESION mayor EXPRESION
  | EXPRESION menor EXPRESION
  | EXPRESION mayor_igual EXPRESION
  | EXPRESION menor_igual EXPRESION
  | EXPRESION igual_que EXPRESION
  | EXPRESION dif_que EXPRESION
  | EXPRESION and EXPRESION
  | EXPRESION or EXPRESION
  | not EXPRESION
  | number
  | cadena
  | character
  | id
  | true
  | false
  | null
  | cor_izq EXPRESIONES cor_der
  | cor_izq cor_der
  | ACCESO_ARREGLO
  | ARRAY_POP
  | TERNARIO
  | FUNCIONES
;

//TIPOS DE DECLARACIONES
DECLARACIONES
  : DECLARACIONES coma DEC_ID
  | DECLARACIONES coma DEC_ID_TIPO
  | DECLARACIONES coma DEC_ID_COR
  | DECLARACIONES coma DEC_ID_EXP
  | DECLARACIONES coma DEC_ID_TIPO_EXP
  | DECLARACIONES coma DEC_ID_TIPO_CORCHETES_EXP
  | DEC_ID
  | DEC_ID_TIPO
  | DEC_ID_COR
  | DEC_ID_EXP
  | DEC_ID_TIPO_EXP
  | DEC_ID_TIPO_CORCHETES_EXP
  | CASTEOS
;
DEC_ID_COR
  : id dos_puntos TIPOS CORCHETES
;
DEC_ID_TIPO
  : id dos_puntos TIPOS
;
DEC_ID
  : id
;

//TIPOS DE VARIABLES AUXILIARES
TIPOS
  : void
  | id
;
```

```
//TIPO DE VARIABLES
TIPO
  : int
  | double
  | char
  | string
  | Boolean
;


//DEC_VARIABLE VARIABLES
DEC_VARIABLE
  : TIPO DECLARACIONES punto_coma
;

//DEC_VARIABLE EXPRESIONES
DEC_ID_EXP
  : id igual EXPRESION
;

//ASIGNACION
ASIGNACION
  : id IGUALES EXPRESION punto_coma
  | id ACCESOS IGUALES EXPRESION punto_coma
  | ACCESO_ARREGLO IGUALES EXPRESION punto_coma
;
IGUALES
  : igual
  | mas igual
  | menos igual
;

//DEC_VARIABLE DE FUNCIONES
DEC_FUNCION
  : id par_izq par_der dos_puntos TIPOS llave_izq INSTRUCCIONES llave_der
  | id par_izq par_der dos_puntos TIPOS CORCHETES llave_izq INSTRUCCIONES
llave_der
  | id par_izq par_der llave_izq INSTRUCCIONES llave_der
  | id par_izq PARAMETROS par_der dos_puntos TIPOS llave_izq INSTRUCCIONES
llave_der
  | id par_izq PARAMETROS par_der dos_puntos TIPOS CORCHETES llave_izq
INSTRUCCIONES llave_der
  | id par_izq PARAMETROS par_der llave_izq INSTRUCCIONES llave_der
;
CORCHETES
  : CORCHETES cor_izq cor_der
  | cor_izq cor_der
;
PARAMETRO
  : TIPO id
;
PARAMETROS
  : PARAMETROS coma PARAMETRO
  | PARAMETRO
;
```

```
//DECLARACION DE ARREGLOS
DEC_ARREGLOS
  : TIPO ACCESOS_ARRAY id igual new TIPO ACCESOS_ARRAY punto_coma
  | TIPO ACCESOS_ARRAY id igual tochararray par_izq EXPRESION par_der punto_coma
  | TIPO ACCESOS_ARRAY id igual llave_izq ARREGLOS llave_der punto_coma
  | TIPO ACCESOS_ARRAY id igual llave_izq EXPRESIONES llave_der punto_coma
;
ARREGLOS
  : ARREGLOS coma llave_izq EXPRESIONES llave_der
  | llave_izq EXPRESIONES llave_der
;
EXPRESIONES
  : EXPRESIONES coma EXPRESION
  | EXPRESION
;

//FUNCION POP
ARRAY_POP
  : id punto pop par_izq par_der punto_coma
;

//FUNCION PUSH
ARRAY_PUSH
  : id punto push par_izq EXPRESION par_der punto_coma
  | id ACCESOS punto push par_izq EXPRESION par_der punto_coma
;

//FUNCION TAMAÑO
ARRAY_LENGTH
  : TIPO id igual length par_izq ACCESO_ARREGLO par_der  punto_coma
  | TIPO id igual length par_izq id par_der punto_coma
;
ACCESO_ARREGLO
  : id ACCESOS_ARRAY
;
ACCESOS
  : ACCESOS punto id
  | punto id
  | ACCESOS punto id ACCESOS_ARRAY
  | punto id ACCESOS_ARRAY
;
ACCESOS_ARRAY
  : ACCESOS_ARRAY cor_izq EXPRESION cor_der
  | cor_izq EXPRESION cor_der
  | ACCESOS_ARRAY cor_izq CAST cor_der
  | cor_izq CAST cor_der
  | ACCESOS_ARRAY cor_izq cor_der
  | cor_izq cor_der
;

//CICLO WHILE
WHILE
  : while par_izq EXPRESION par_der llave_izq INSTRUCCIONES llave_der
;
```

```
//CICLO DO WHILE
DO_WHILE
  : do llave_izq INSTRUCCIONES llave_der while par_izq EXPRESION par_der punto_coma
  | do llave_izq INSTRUCCIONES llave_der until par_izq EXPRESION par_der punto_coma
;

//CICLO FOR
FOR
  : for par_izq DEC_VARIABLE EXPRESION punto_coma ASIGNACION_FOR par_der
llave_izq INSTRUCCIONES llave_der
  | for par_izq ASIGNACION EXPRESION punto_coma ASIGNACION_FOR par_der llave_izq
INSTRUCCIONES llave_der
;
ASIGNACION_FOR
  : id IGUALES EXPRESION
  | id mas_mas
  | id menos_menos
  | id mas EXPRESION
  | id menos EXPRESION
;

//SENTENCIA SWITHC
SWITCH
  : switch par_izq EXPRESION par_der llave_izq CASES llave_der
;
CASES
  : CASES CASE
  | CASE
  | DEFAULT
  | CASES DEFAULT
;
CASE
  : case EXPRESION dos_puntos INSTRUCCIONES
;
DEFAULT
  : default dos_puntos INSTRUCCIONES
;
CONTINUE
  : continue punto_coma
;
BREAK
  : break punto_coma
;
RETURN
  : return EXPRESION punto_coma
  | return punto_coma
;

//SENTENCIA IF
SENTENCIA_IF
  : IF
  | IF ELSE
  | IF L_ELIF
  | IF L_ELIF ELSE
;
```

```
IF
  : if par_izq EXPRESION par_der llave_izq INSTRUCCIONES llave_der
;
ELSE
  : else llave_izq INSTRUCCIONES llave_der
;
ES_ELIF
  : elif par_izq EXPRESION par_der llave_izq INSTRUCCIONES llave_der
;
L_ELIF
  : L_ELIF ES_ELIF
  | ES_ELIF
;

//EJECUTAR FUNCION
RUN_FUNCION
  : run id par_izq par_der punto_coma
  | run id par_izq EXPRESIONES par_der punto_coma
;

//LLAMADA DE FUNCION
LLAMADA_FUNCION
  : id par_izq par_der punto_coma
  | id par_izq EXPRESIONES par_der punto_coma
;

//DECLARACION DE FUNCIONES
FUNCIONES
  : id par_izq par_der
  | id par_izq EXPRESIONES par_der
  | typeof par_izq EXPRESIONES par_der
  | tostring par_izq EXPRESIONES par_der
  | tolower par_izq EXPRESIONES par_der
  | toupper par_izq EXPRESIONES par_der
  | round par_izq EXPRESIONES par_der
;

//CASTEOS
CASTEOS
  : id igual par_izq TIPO par_der EXPRESION
;
CAST
  : par_izq TIPO par_der EXPRESION
;

//INCREMENTO Y DECREMENTO
MAS_MENOS
  : id mas_mas punto_coma
  | id menos_menos punto_coma
  | mas_mas id punto_coma
  | menos_menos id punto_coma
;
```

```
//OPERADOR TERNARIO
TERNARIO
  : EXPRESION interrogacion EXPRESION dos_puntos EXPRESION
;

//PRINT AND PRINTLN
PRINT
  : print par_izq EXPRESIONES par_der punto_coma
;
PRINTLN
  : println par_izq EXPRESIONES par_der punto_coma
;
```

Repositorio del proyecto: https://github.com/BeKingGO/OLC1-202006353