# Gramática tipo 2 análisis sintáctico ascendente

Terminales = {cadena, int, double, boolean, char, String, true, false, new, if, else, switch, case, default, println, print, break, while, for, do, continue, return, void, toLower, toUpper, round, length, TypeOf, to_String, toCharArray, run, increment, decrement, mas, menos, multiplicacion, division, exponente, modulo, igualacion, diferenciacion, menorIgualQue, mayorIgualQue, menorQue, mayorQue, igual, interrogacion, dosPuntos, puntoYcoma, or, and, not, parentesisAbre, parentesisCierra, llaveAbre, llaveCierra, corcheteAbre, corecheteCierra, coma, identificador, decimal, entero, EOF}

Nota: aquí los terminales son todos mis tokens y van todo en mayúscula.

No Terminales = {INICIO, ...<>}

Inicio = INICIO

## Producciones

INICIO:

ENTRADAS EOF

;

ENTRADAS:

ENTRADAS ENTRADA

| ENTRADA

;

ENTRADA:

FUNCION

| METODO

| RUN

| DECLARACION_VAR puntoYcoma

| DECLARACION_VECT

| INSTRUCCION

;

```
FUNCION:
        identificador parentesisAbre parentesisCierra dosPuntos TIPO BLOQUE
        |   identificador parentesisAbre LISTA_PARAMETROS parentesisCierra dosPuntos TIPO BLOQUE
;

METODO:
        identificador parentesisAbre parentesisCierra dosPuntos void BLOQUE
        |   identificador parentesisAbre LISTA_PARAMETROS parentesisCierra dosPuntos void BLOQUE
        |   identificador parentesisAbre parentesisCierra BLOQUE
        |   identificador parentesisAbre LISTA_PARAMETROS parentesisCierra BLOQUE
;

LISTA_PARAMETROS:
               LISTA_PARAMETROS coma TIPO identificador
               |       TIPO identificador
;

RUN:
    run identificador parentesisAbre parentesisCierra puntoYcoma
    |   run identificador parentesisAbre LISTA_PARAMETROS parentesisCierra puntoYcoma
;

DECLARACION_VAR:
            TIPO LISTA_VARIABLES
            |   TIPO LISTA_VARIABLES igual EXPRESION
;
```

```
LISTA_VARIABLES:
                LISTA_VARIABLES coma identificador
                |   identificador
;

DECLARACION_VECT:
                TIPO identificador corcheteAbre corcheteCierra igual new TIPO corcheteAbre EXPRESION
corcheteCierra puntoYcoma
                |       TIPO identificador corcheteAbre corcheteCierra corcheteAbre corcheteCierra igual
new TIPO corcheteAbre EXPRESION corcheteCierra corcheteAbre EXPRESION corcheteCierra puntoYcoma
                |       TIPO identificador corcheteAbre corcheteCierra igual corcheteAbre LISTA_VALORES
corcheteCierra puntoYcoma
;

LISTA_VALORES:
                LISTA_VALORES coma EXPRESION
                | EXPRESION
;

INSTRUCCIONES:
        INSTRUCCIONES INSTRUCCION
        | INSTRUCCION
;

INSTRUCCION:
        DECLARACION_VAR puntoYcoma
        |       DECLARACION_VECT
        |       FOR
        |       WHILE
```

```
        |       DO_WHILE
|       SENTENCIA_TRANSFERENCIA
        |       IF
        |       SWITCH
        |       ASIGNACION puntoYcoma
        |       INCREMENTO puntoYcoma
        |       DECREMENTO puntoYcoma
        |       LLAMADA puntoYcoma
        |       PRINT
        |       PRINTLN
;

FOR: for parentesisAbre FOR_DECLARACION puntoYcoma EXPRESION puntoYcoma FOR_ITERADOR parentesisCierra
BLOQUE ;

FOR_DECLARACION:
                DECLARACION_VAR { $$ = $1; }
                |       ASIGNACION { $$ = $1; }
;

FOR_ITERADOR:
                ASIGNACION { $$ = $1; }
                |       INCREMENTO { $$ = $1; }
                |       DECREMENTO { $$ = $1; }
;

WHILE: while parentesisAbre EXPRESION parentesisCierra BLOQUE;
```

```
DO_WHILE: do BLOQUE while parentesisAbre EXPRESION parentesisCierra puntoYcoma ;

IF: if  parentesisAbre EXPRESION parentesisCierra BLOQUE CONTROL_ELSE ;

CONTROL_ELSE:
        else BLOQUE { $$ = $2; }
        |       else IF { $$ = $2; }
        |       ε
;

BLOQUE:
 llaveAbre INSTRUCCIONES llaveCierra
    | llaveAbre              llaveCierra
;


SWITCH: switch parentesisAbre EXPRESION parentesisCierra llaveAbre CASELIST DEFAULT llaveCierra ;

CASELIST:
        CASELIST case EXPRESION dosPuntos INSTRUCCIONES
        |       CASELIST case EXPRESION dosPuntos INSTRUCCIONES break puntoYcoma
        |       case EXPRESION dosPuntos INSTRUCCIONES
        |       case EXPRESION dosPuntos INSTRUCCIONES  break puntoYcoma
        |       ε
;

DEFAULT:
        default dosPuntos INSTRUCCIONES break puntoYcoma
```

```
        |       default dosPuntos INSTRUCCIONES
        |       ε
;
ASIGNACION: identificador igual EXPRESION;
INCREMENTO: identificador incremento;
DECREMENTO: identificador decremento;
LLAMADA:identificador parentesisAbre LISTA_VALORES parentesisCierra;
PRINT: print parentesisAbre EXPRESION parentesisCierra puntoYcoma;
PRINTLN:println parentesisAbre EXPRESION parentesisCierra puntoYcoma;

EXPRESION:
        /*Operaciones aritmeticas*/
        EXPRESION mas EXPRESION
        |       EXPRESION menos EXPRESION
        |       EXPRESION multiplicacion EXPRESION
        |       EXPRESION division EXPRESION
        |       EXPRESION exponente EXPRESION
        |       EXPRESION modulo EXPRESION
        |       menos EXPRESION %prec umenos
        |       parentesisAbre EXPRESION parentesisCierra
        /*Operaciones condicionales*/
        |       EXPRESION igualacion EXPRESION
        |       EXPRESION diferenciacion EXPRESION
        |       EXPRESION menorQue EXPRESION
        |       EXPRESION mayorQue EXPRESION
        |       EXPRESION mayorIgualQue EXPRESION
        |       EXPRESION menorIgualQue EXPRESION
        |       not EXPRESION
```

```
        /*LLamada de función que devuelve un valor*/
        |       LLAMADA
        /*Casteos*/
        |       parentesisAbre TIPO parentesisCierra EXPRESION
        /*Funciones reservadas del lenguaje*/
        |       toLower parentesisAbre EXPRESION parentesisCierra
        |       toUpper parentesisAbre EXPRESION parentesisCierra
        |       round parentesisAbre EXPRESION parentesisCierra
        |       length parentesisAbre EXPRESION parentesisCierra
        |       typeOf parentesisAbre EXPRESION parentesisCierra
        |       tostring parentesisAbre EXPRESION parentesisCierra
        |       toCharArray parentesisAbre EXPRESION parentesisCierra
        /*Valores que pueden estar en las expresiones*/
        |       cadena
        |       entero
        |       decimal
        |       caracter
        |       true
        |       false
        |       identificador
        |       EXPRESION incremento
        |       EXPRESION decremento
;

SENTENCIA_TRANSFERENCIA:
                        break puntoYcoma
                        |       continue puntoYcoma
                        |       return EXPRESION puntoYcoma
```

```
                        |       return puntoYcoma
;


TIPO:
    int
    |   double
    |   boolean
    |   char
    |   string
;
```