Gramaticas

start with ini; // iniciar la gramataica

ini::= inicio;

inicio ::= comentario

| JSON

| VOID MAIN PAR_IZQ PAR_DER CHR_IZQ instrucciones CHR_DER

| error CHR_DER

;

instrucciones ::= instrucciones instruccion

| instruccion

;

Intrucicones dentro del main

instruccion ::= sentencia PTCOMA

| sumar PTCOMA

| declaracion PTCOMA

| condicion

| sentencia_while

| vacio

| Sentencia_else

| sentencia_switch

| INSTANCIA PTCOMA

| DO_WHILE

| CICLO_FOR

| CONBREAK

| Variables_globales

| BARRAS

| PIE

;

sumar::= REVALUAR COR_IZQ expresion:a COR_DER ;

sentencia::= IMPRIMIR PUNTO WRITE PAR_IZQ expresion:a PAR_DER ;

//DECLARACION VARIABLES

declaracion::= tipoDato expresion:b;

INSTANCIA::= ID:a IGUAL expresion:b;

//SENTENCIA WHILE

sentencia_while::= ENCICLADO PAR_IZQ expresion:a PAR_DER CHR_IZQ instrucciones CHR_DER ;

//DO WHILE

DO_WHILE::= RDO CHR_IZQ instrucciones CHR_DER ENCICLADO PAR_IZQ expresion:a PAR_DER PTCOMA

//FOR

CICLO_FOR::= FOR PAR_IZQ PRUEBA ID:a comparativa ENTERO:b PTCOMA ID comparativa ENTERO:c PTCOMA ID MAS MAS PAR_DER CHR_IZQ instrucciones CHR_DER ;

//SENTENCIA IF

condicion::= RIF PAR_IZQ expresion:a PAR_DER CHR_IZQ instrucciones CHR_DER ;

Else

Sentencia_else::= condicion DESIF sentencia_if

Formas que puede venir el else

```
sentencia_if ::= condicion
   |else_simple
   | Sentencia_else
;
else_simple::= CHR_IZQ instrucciones CHR_DER
```

//SETENCIA SWITCH

```
sentencia_switch ::= RSWITCH PAR_IZQ expresion:a PAR_DER CHR_IZQ
{:
   reduccionDeTabulaciones +=1;
   Traduccion.add(tabuladores+"match "+a);
   tabuladores = "\t".repeat(reduccionDeTabulaciones);
:}
CASES CHR_DER{:
   tabuladores = tabuladores.replaceFirst("\t", "");
   reduccionDeTabulaciones -=1;
:};


CASO::= RCASE expresion:a DOSPT {:
   Traduccion.add(tabuladores+"case "+a);
:}
;


CASES ::=
   CASO instrucciones  CASES
   | CASO instrucciones
   | DEFECTO DOSPT {:Traduccion.add(tabuladores+"default"); :}instrucciones
;
CONBREAK::= RBREAK PTCOMA;
```

```
tipoDato::=
 REVALUAR:a
| PRUEBA:a
| VARCHAR:a
| VARBOOL:a
| VARSTRING:a
| BOOLTRUE:a
| BOOLFAL:a
| VARDOUBLE:a
;

expresion ::= operaciones:a
        | comparacion:a
        | ENTERO:a
        | DECIMAL:a
        | VARBOOL:a
        | logica:a
        | ID:a
        | BOOLTRUE:a
        | BOOLFAL:a
        | CADENA:a
        | CHAR:a
        | PAR_IZQ expresion:e PAR_DER

;
```

```
operaciones ::=
            expresion:a MAS expresion:b
          | MENOS expresion:a
          | expresion:a COMMA expresion:b
          | expresion:a MENOS expresion:b
          | expresion:a POR expresion:
          | expresion:a DIV expresion:b
          | expresion:a IGUAL expresion:b

         ;

comparacion ::=
            expresion:a IGUALDAD:signo expresion:b
          | expresion:a MAYOR_QUE:signo expresion:b
          | expresion:a MENOR_IGUAL:signo expresion:b
          | expresion:a MENOR_QUE:signo expresion:b
          | expresion:a MAYOR_IGUAL:signo expresion:b
          | expresion:a DIFERENTE:signo expresion:b
;

logica ::=
         expresion:a OR:l expresion:b
       | expresion:a AND:l expresion:b
       | NEGACION:l expresion:b
;
```

```
comparativa::=
            IGUAL:a          {:RESULT = a;:}
          | MAYOR_QUE:a      {:RESULT = a;:}
          | MENOR_QUE:a      {:RESULT = a;:}
          | MENOR_IGUAL:a    {:RESULT = a;:}
          | MAYOR_IGUAL:a    {:RESULT = a;:}
          | DIFERENTE:a      {:RESULT = a;:}
;

JSON::= CHR_IZQ sentencias_json CHR_DER {:
   System.out.println(Graficas.nombrejson);
   Jsons.put( Graficas.nombrejson, mapa);
:}
;



sentencias_json ::= sentencia_json
        | sentencia_json COMMA sentencias_json
;

//JSON
sentencia_json::= CADENA:a DOSPT CADENA:b {: mapa.put(a, b); :}
          | CADENA:a DOSPT DECIMAL:b {: mapa.put(a, b); System.out.println(b);  :}
          | CADENA:a DOSPT ENTERO:b {: mapa.put(a, b); :}
;
```

```
Variables_globales::= VOID GLOBAL PAR_IZQ PAR_DER CHR_IZQ globales CHR_DER



globales ::=  global PTCOMA
        | global PTCOMA globales;

global ::= VARDOUBLE ID:a IGUAL DECIMAL:b
|  VARSTRING ID:a IGUAL CADENA:b
|  VARDOUBLE ID:a IGUAL ENTERO:b
|  VARDOUBLE ID:a IGUAL instancia:b
|  VARSTRING ID:a IGUAL instancia:b
;
```

//GRAFICA DE BARRAS

```
BARRAS::= VOID GBARRAS PAR_IZQ PAR_DER CHR_IZQ BarrasGlobales CHR_DER
{:
   //Graficas.GraficarPie(title,ejex,ejey,ejes,valores);
   Graficas.Graficar(title,ejex,ejey,ejes,valores);

:}
;

BarrasGlobales::= BarrasGlobal PTCOMA
            | BarrasGlobal PTCOMA BarrasGlobales
;
```

```
BarrasGlobal::= VARSTRING ARREGLO ID IGUAL CHR_IZQ Intancias:a CHR_DER
        | VARDOUBLE ARREGLO ID IGUAL CHR_IZQ Intancias:a CHR_DER
        | VARSTRING TITULOG IGUAL CADENA:a
        | VARSTRING TITLEY IGUAL CADENA:a
        | VARSTRING TITLEX IGUAL CADENA:a
        | VARSTRING TITULOG IGUAL ID:a
        | VARDOUBLE ARREGLO ID IGUAL instancia:a
        | VARSTRING TITULOG IGUAL instancia:a
        | VARSTRING TITLEY IGUAL ID:a
        | VARSTRING TITLEX IGUAL ID:
;

Intancias ::= ID:
        | ID:a COMMA Intancias:
        | DECIMAL:b
        | DECIMAL:a COMMA Intancias:b
        | CADENA:b
        | CADENA:a COMMA Intancias:b
        | instancia:
        | instancia:a COMMA Intancias:b
;

instancia ::= DOLLAR CHR_IZQ NEWVAL COMMA CADENA:a COMMA CADENA:b CHR_DER
;

//GRAFICA DE PIE
PIE::= VOID GPIE PAR_IZQ PAR_DER CHR_IZQ BarrasGlobales CHR_DER
;
```