

GRAMATICAS

/* terminales */

terminal String PTCOMA,PAR_IZQ,PAR_DER,COR_IZQ,COR_DER; // mismo nombre en el archivo lexico

terminal String MAS,MENOS,POR,DIV;

terminal String ENTERO;

terminal String DECIMAL;

terminal String UMENOS; // para la precedencia para los negativos y no en

terminal String REVALUAR;

terminal String CADENA;

terminal String WRITE;

terminal String PUNTO;

terminal String IMPRIMIR;

terminal String MAIN;

terminal String VOID;

terminal String CHR_IZQ;

terminal String CHR_DER;

terminal String ID;

terminal String IGUAL;

terminal String RIF;

terminal String EJEX, EJEY;

terminal String GLOBAL, GLOBALM;

terminal String GBARRAS;

terminal String IGUALDAD, MENOR_QUE,MAYOR_QUE, MENOR_IGUAL, MAYOR_IGUAL,
DIFERENTE, OR, AND;

terminal String ENCICLADO;

terminal String DESIF;

```
terminal String BOOLTRUE,BOOLFAL, NEGACION;  
terminal String COMMA;  
terminal String RSWITCH ,RCASE ,RBREAK, DOSPT, DEFECTO;  
terminal String TITLE;  
terminal String RDO;  
terminal String FOR;  
terminal String CHAR;
```

```
//variables
```

```
terminal String PRUEBA;  
terminal String VARCHAR;  
terminal String VARBOOL;  
terminal String VARSTRING;  
terminal String VARDOUBLE;
```

```
//terminal String RDOUBLE,RCHAR,RSTRING,RBOOL;
```

```
non terminal ini;                // terminales para las transiciones  
non terminal instrucciones;  
non terminal instruccion;  
non terminal expresion;  
non terminal sentencia;  
non terminal sumar;  
non terminal declaracion;  
non terminal tipoDato;  
non terminal condicion;  
non terminal comparacion;  
non terminal sentencia_while;
```

non terminal vacio;
non terminal operaciones;
non terminal inicio;
non terminal comentario;
non terminal Sentencia_else;
non terminal sentencia_if;
non terminal else_simple;
non terminal sentencia_switch;
non terminal CASES;
non terminal CASO;
non terminal JSON;
non terminal sentencias_json, sentencia_json;
non terminal logica;
non terminal INSTANCIA;
non terminal DO_WHILE;
non terminal CICLO_FOR;
non terminal comparativa;
non terminal CONBREAK;
non terminal Variables_globales;
non terminal GLOBALES;
non terminal instrucciones_json;
non terminal BARRAS;

precedence left MAS,MENOS; // precedencia de operadores
precedence left POR,DIV;
precedence left IGUAL;
precedence right UMENOS ;
precedence right COMMA;

precedence left IGUALDAD,MENOR_QUE, MAYOR_QUE, MENOR_IGUAL,
MAYOR_IGUAL,DIFERENTE, OR , AND,NEGACION;

start with ini; // iniciar la gramataica

ini::= inicio;

inicio ::= comentario

| JSON

| VOID MAIN PAR_IZQ {:

 tabuladores ="";

 reduccionDeTabulaciones= 0;

 reduccionDeTabulaciones +=1;

 Traduccion.add("def main: ");

 tabuladores = "\t".repeat(reduccionDeTabulaciones);

 :}

 PAR_DER CHR_IZQ instrucciones CHR_DER

 | error CHR_DER

;

instrucciones ::= instrucciones instruccion

| instruccion

;

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
;

sumar::= REVALUAR COR_IZQ expresion:a COR_DER
{
    //System.out.println("El valor de la expresión es: "+a);
}
;

sentencia::= IMPRIMIR PUNTO WRITE PAR_IZQ expresion:a PAR_DER
{
    Traducccion.add(tabuladores+"print("+a+"");
    //System.out.println("print("+a+"");
}
;

//DECLARACION VARIABLES

declaracion::= tipoDato expresion:b
{
    Traducccion.add(tabuladores+b);
}

```

```

//System.out.println("Declaración encontrada: "+a+" con valor: "+b) ;
:}
;
INSTANCIA::= ID:a IGUAL expresion:b
{
    Traduccion.add(tabuladores+a+"="+" "+b);
:}
;

```

//SENTENCIA WHILE

```

sentencia_while::= ENCICLADO PAR_IZQ expresion:a PAR_DER CHR_IZQ
{
    reduccionDeTabulaciones +=1;
    Traduccion.add(tabuladores+"while "+a);
    tabuladores = "\t".repeat(reduccionDeTabulaciones);
    //System.out.println("if "+a+" "+" "+b+" "+" "+c) ;
:}
instrucciones CHR_DER
{
    tabuladores = tabuladores.replaceFirst("\t", "");
    reduccionDeTabulaciones -=1;
:}
;

```

//DO WHILE

```

DO_WHILE::= RDO CHR_IZQ {
    reduccionDeTabulaciones +=1;

```

```

    Traduccion.add(tabuladores+"while true");

    tabuladores = "\t".repeat(reduccionDeTabulaciones);

:}

instrucciones CHR_DER ENCICLADO PAR_IZQ expresion:a PAR_DER PTCOMA

{:
    tabuladores = tabuladores.replaceFirst("\t", "");

    reduccionDeTabulaciones -=1;

:};

//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

{:
    reduccionDeTabulaciones +=1;

    Traduccion.add(tabuladores+"for "+a+" in "+" range "+"("+b+", "+c+""));

    tabuladores = "\t".repeat(reduccionDeTabulaciones);

:}

instrucciones CHR_DER {:
    tabuladores = tabuladores.replaceFirst("\t", "");

    reduccionDeTabulaciones -=1;

:}

;

//SENTENCIA IF

condicion::= RIF PAR_IZQ expresion:a PAR_DER CHR_IZQ

{:
    reduccionDeTabulaciones +=1;

```

```

Traduccion.add(tabuladores+else_+"if "+a);
tabuladores = "\t".repeat(reduccionDeTabulaciones);
else_ = "";
//System.out.println(tabuladores+"if "+a+ " " +b+ " " +c) ;
:}
instrucciones CHR_DER
{:
    else_ = "";
    tabuladores = tabuladores.replaceFirst("\t", "");
    reduccionDeTabulaciones -=1;
:}
;

```

Sentencia_else ::= condicion DESIF {:

```

    else_ = "else ";
:}

```

sentencia_if

```

{:

```

```

:};

```

sentencia_if ::= condicion

```

    |else_simple
    | Sentencia_else
;

```

else_simple ::= CHR_IZQ

```

{:
    reduccionDeTabulaciones +=1;

```



```

Traduccion.add(tabuladores+else_);

tabuladores = "\t".repeat(reduccionDeTabulaciones);

else_ = "";

//System.out.println(tabuladores+"if "+a+ " " +b+ " " +c) ;

:}

instrucciones CHR_DER

{:

    else_ = "";

    tabuladores = tabuladores.replaceFirst("\t", "");

    reduccionDeTabulaciones -=1;

:}

;

//

//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 { : RESULT = a; : }

| PRUEBA:a { : RESULT = a; : }

| VARCHAR:a { : RESULT = a; : }

| VARBOOL:a { : RESULT = a; : }

| VARSTRING:a { : RESULT = a; : }

| BOOLTRUE:a { : RESULT = a; : }

| BOOLFAL:a { : RESULT = a; : }

| VARDOUBLE:a { : RESULT = a; : }

;

expresion ::= operaciones:a { : RESULT = a; : }

| comparacion:a { : RESULT = a; : }

| ENTERO:a { : RESULT = a; : }

| DECIMAL:a { : RESULT = a; : }

| VARBOOL:a { : RESULT = a; : }

| logica:a { : RESULT = a; : }

| ID:a { : RESULT = a; : }

BOOLTRUE:a	{: RESULT = a;:}
BOOLFAL:a	{: RESULT = a;:}
CADENA:a	{: RESULT = a;:}
CHAR:a	{: RESULT = a;:}
PAR_IZQ expresion:e PAR_DER	{:RESULT = e;:}

;

operaciones ::=

expresion:a MAS expresion:b	{:RESULT = a + "+" + b;:}
expresion:a MENOS expresion:b	{:RESULT = a + "-" + b;:}
expresion:a POR expresion:b	{:RESULT = a + "*" + b;:}
expresion:a DIV expresion:b	{:RESULT = a + "/" + b;:}
expresion:a IGUAL expresion:b	{:RESULT = a + "=" + b;:}
expresion:a COMMA expresion:b	{:RESULT = a + "," + b;:}

;

comparacion ::=

expresion:a IGUALDAD:l expresion:b	{:RESULT = a + l + b;:}
expresion:a MAYOR_QUE:l expresion:b	{:RESULT = a + l + b;:}
expresion:a MENOR_QUE:l expresion:b	{:RESULT = a + l + b;:}
expresion:a MENOR_IGUAL:l expresion:b	{:RESULT = a + l + b;:}
expresion:a MAYOR_IGUAL:l expresion:b	{:RESULT = a + l + b;:}
expresion:a DIFERENTE:l expresion:b	{:RESULT = a + l + b;:}

;

logica ::=

expresion:a OR:l expresion:b	{:RESULT = a + " or " + b;:}
expresion:a AND:l expresion:b	{:RESULT = a + " and " + b;:}

```
    | NEGACION:l expresion:b {:RESULT = " not " + 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("HE Recibido un JSON");  
:};
```

```
sentencias_json ::= sentencia_json  
    | sentencia_json COMMA sentencias_json  
;  

```

```
sentencia_json::= TITLE DOSPT CADENA:a {: title = a; :}  
    | EJEX DOSPT CADENA:a {: ejex = a; :}  
    | EJEY DOSPT CADENA:a {: ejey = a; :}  
    | CADENA DOSPT CADENA:a {: Grafica.add(a); :}  
    | CADENA DOSPT DECIMAL:a {: Valores.add(a); :}  
;  

```

GLOBALES ::= GLOBALM

| GLOBAL ;

Variables_globales ::= VOID GLOBALES PAR_IZQ PAR_DER CHR_IZQ instrucciones_json CHR_DER;

instrucciones_json ::= VARSTRING ID:a IGUAL CADENA:b PTCOMA {: mapa.put(a, b); :}

| VARDOUBLE ID IGUAL:a DECIMAL:b PTCOMA {: mapa.put(a, b); :}

;

BARRAS ::= VOID GBARRAS PAR_IZQ PAR_DER CHR_IZQ CHR_DER;