Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

**MANUAL TECNICO**

**Proyecto 1**

Nombre: Cristian Alexander Azurdia Ajú

Carnet: 201020331

**Gramáticas**

* Gramática de GXML:

start with INICIO;

INICIO ::= CUERPO\_GXML:n

{:

if(importar == true && traducir == true)

{

RESULT = new Sentencia\_Inicio(n,entrada,importar);

salida = RESULT.ejecutar(entorno\_global,"");

}

else if(importar == true && traducir == false)

{

RESULT = new Sentencia\_Inicio(n,entrada,importar);

salida = RESULT.ejecutar(entorno\_global, new FS\_Arreglo(), entrada);

}

else if( importar == false && traducir == true)

{

RESULT = new Sentencia\_Inicio(n,entrada,importar);

salida = RESULT.ejecutar(null,"");

}

else if( importar == false && traducir == false)

{

RESULT = new Sentencia\_Inicio(n,entrada,importar);

salida = RESULT.ejecutar(null, new FS\_Arreglo(), entrada);

}

:};

CUERPO\_GXML ::= CUERPO\_GXML:n1 SENTENCIA\_IMPORTAR:n2

{:

RESULT = n1;

RESULT.add(n2);

:}

|CUERPO\_GXML:n1 TAG\_VENTANA:n2

{:

RESULT = n1;

RESULT.add(n2);

:}

|SENTENCIA\_IMPORTAR:n

{:

RESULT = new ArrayList<Instruccion>();

RESULT.add(n);

:}

|TAG\_VENTANA:n

{:

RESULT = new ArrayList<Instruccion>();

RESULT.add(n);

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SENTENCIA IMPORTAR\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SENTENCIA\_IMPORTAR ::= s\_less r\_importar:n1 s\_greather contenido\_tag:n2 s\_less s\_div r\_importar s\_greather

{:

RESULT = new Sentencia\_Importar(n2,n1right,n1left);

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CREAR VENTANAS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

TAG\_VENTANA ::= s\_less:n1 r\_ventana LISTA\_ELEMENTOS:n2 s\_greather LISTA\_CONTENIDO\_VENTANA:n3 s\_less s\_div r\_ventana s\_greather

{:

RESULT = new Sentencia\_Ventana(n2,n3,n1right,n1left);

:}

|s\_less:n1 r\_ventana LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_ventana s\_greather

{:

RESULT = new Sentencia\_Ventana(n2,n1right,n1left);

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*CONTENIDO VENTANAS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LISTA\_CONTENIDO\_VENTANA ::= LISTA\_CONTENIDO\_VENTANA:n1 CONTENIDO\_VENTANA:n2

{:

RESULT = n1;

RESULT.add(n2);

:}

|CONTENIDO\_VENTANA:n

{:

RESULT = new ArrayList<Instruccion>();

RESULT.add(n);

:};

CONTENIDO\_VENTANA ::= TAG\_CONTENEDOR:n

{:

RESULT = n;

:}

|TAG\_TEXTO:n

{:

RESULT = n;

:}

|TAG\_CONTROL:n

{:

RESULT = n;

:}

|TAG\_BOTON:n

{:

RESULT = n;

:}

|TAG\_ENVIAR:n

{:

RESULT = n;

:}

|TAG\_MULTIMEDIA:n

{:

RESULT = n;

:};

TAG\_CONTENEDOR ::= s\_less:n1 r\_contenedor LISTA\_ELEMENTOS:n2 s\_greather LISTA\_CONTENIDO\_VENTANA:n3 s\_less s\_div r\_contenedor s\_greather

{:

RESULT = new Sentencia\_Contenedor(n2,n3,n1right,n1left);

:}

|s\_less:n1 r\_contenedor LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_contenedor s\_greather

{:

RESULT = new Sentencia\_Contenedor(n2,n1right,n1left);

:};

TAG\_TEXTO ::= s\_less:n1 r\_texto LISTA\_ELEMENTOS:n2 s\_greather contenido\_tag:n3 s\_less s\_div r\_texto s\_greather

{:

RESULT = new Sentencia\_Texto(n2,n3,n1right,n1left);

:}

|s\_less:n1 r\_texto LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_texto s\_greather

{:

RESULT = new Sentencia\_Texto(n2,n1right,n1left);

:};

TAG\_CONTROL ::= s\_less:n1 r\_control LISTA\_ELEMENTOS:n2 s\_greather CONTENIDO\_CONTROL:n3 s\_less s\_div r\_control s\_greather

{:

RESULT = new Sentencia\_Controlador(n2,n3,n1right,n1left);

:}

|s\_less:n1 r\_control LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_control s\_greather

{:

RESULT = new Sentencia\_Controlador(n2,n1right,n1left);

:};

TAG\_BOTON ::= s\_less:n1 r\_boton LISTA\_ELEMENTOS:n2 s\_greather contenido\_tag:n3 s\_less s\_div r\_boton s\_greather

{:

RESULT = new Sentencia\_Boton(n2,n3,n1right,n1left,false);

:}

|s\_less:n1 r\_boton LISTA\_ELEMENTOS:n2 s\_greather s\_less r\_texto ELEMENTO\_NOMBRE s\_greather contenido\_tag:n3 s\_less s\_div r\_texto s\_greather s\_less s\_div r\_boton s\_greather

{:

RESULT = new Sentencia\_Boton(n2,n3,n1right,n1left,false);

:}

|s\_less:n1 r\_boton LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_boton s\_greather

{:

RESULT = new Sentencia\_Boton(n2,n1right,n1left,false);

:};

TAG\_ENVIAR ::= s\_less:n1 r\_enviar LISTA\_ELEMENTOS:n2 s\_greather contenido\_tag:n3 s\_less s\_div r\_enviar s\_greather

{:

RESULT = new Sentencia\_Boton(n2,n3,n1right,n1left,true);

:}

|s\_less:n1 r\_enviar LISTA\_ELEMENTOS:n2 s\_greather s\_less r\_texto ELEMENTO\_NOMBRE s\_greather contenido\_tag:n3 s\_less s\_div r\_texto s\_greather s\_less s\_div r\_enviar s\_greather

{:

RESULT = new Sentencia\_Boton(n2,n3,n1right,n1left,true);

:}

|s\_less:n1 r\_enviar LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_enviar s\_greather

{:

RESULT = new Sentencia\_Boton(n2,n1right,n1left,true);

:};

TAG\_MULTIMEDIA ::= s\_less:n1 r\_multimedia LISTA\_ELEMENTOS:n2 s\_greather s\_less s\_div r\_multimedia s\_greather

{:

RESULT = null;

:};

CONTENIDO\_CONTROL ::= CONTENIDO\_CONTROL:n2 TAG\_DEFECTO:n1

{:

RESULT = n2;

RESULT.add(n1);

:}

|CONTENIDO\_CONTROL:n1 TAG\_LISTA\_DATOS:n2

{:

RESULT = n1;

RESULT.addAll(n2);

:}

|TAG\_LISTA\_DATOS:n

{:

RESULT = n;

:}

|TAG\_DEFECTO:n

{:

RESULT = new ArrayList<GXML\_Elemento>();

RESULT.add(n);

:};

TAG\_DEFECTO ::= s\_less:n1 r\_defecto s\_greather contenido\_tag:n2 s\_less s\_div r\_defecto s\_greather

{:

RESULT = new GXML\_Elemento("defecto", Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2, n1right, n1left);

:};

TAG\_LISTA\_DATOS ::= s\_less:n1 r\_listadatos s\_greather LISTA\_DATOS:n2 s\_less s\_div r\_listadatos s\_greather

{:

RESULT = n2;

:}

|s\_less:n1 r\_listadatos s\_greather s\_less s\_div r\_listadatos s\_greather

{:

RESULT = new ArrayList<GXML\_Elemento>();

:};

LISTA\_DATOS ::= LISTA\_DATOS:n1 TAG\_DATO:n2

{:

RESULT = n1;

RESULT.add(n2);

:}

|TAG\_DATO:n

{:

RESULT = new ArrayList<GXML\_Elemento>();

RESULT.add(n);

:};

TAG\_DATO ::= s\_less:n1 r\_dato s\_greather contenido\_tag:n2 s\_less s\_div r\_dato s\_greather

{:

RESULT = new GXML\_Elemento("dato", Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2, n1right, n1left);

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ELEMENTOS DE LAS ETIQUETAS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LISTA\_ELEMENTOS ::= LISTA\_ELEMENTOS:n1 ELEMENTO:n2

{:

RESULT = n1;

RESULT.add(n2);

:}

|ELEMENTO:n

{:

RESULT = new ArrayList<GXML\_Elemento>();

RESULT.add(n);

:};

ELEMENTO ::= ELEMENTO\_ID:n

{:

RESULT = n;

:}

|ELEMENTO\_NOMBRE:n

{:

RESULT = n;

:}

|ELEMENTO\_TIPO:n

{:

RESULT = n;

:}

|ELEMENTO\_COLOR:n

{:

RESULT = n;

:}

|ELEMENTO\_ACCIONINICIAL:n

{:

RESULT = n;

:}

|ELEMENTO\_ACCIONFINAL:n

{:

RESULT = n;

:}

|ELEMENTO\_POSX:n

{:

RESULT = n;

:}

|ELEMENTO\_POSY:n

{:

RESULT = n;

:}

|ELEMENTO\_ALTO:n

{:

RESULT = n;

:}

|ELEMENTO\_ANCHO:n

{:

RESULT = n;

:}

|ELEMENTO\_BORDE:n

{:

RESULT = n;

:}

|ELEMENTO\_FUENTE:n

{:

RESULT = n;

:}

|ELEMENTO\_TAM:n

{:

RESULT = n;

:}

|ELEMENTO\_NEGRITA:n

{:

RESULT = n;

:}

|ELEMENTO\_CURSIVA:n

{:

RESULT = n;

:}

|ELEMENTO\_TEXTO:n

{:

RESULT = n;

:}

|ELEMENTO\_NUMERICO:n

{:

RESULT = n;

:}

|ELEMENTO\_TEXTOAREA:n

{:

RESULT = n;

:}

|ELEMENTO\_DESPLEGABLE:n

{:

RESULT = n;

:}

|ELEMENTO\_MAXIMO:n

{:

RESULT = n;

:}

|ELEMENTO\_MINIMO:n

{:

RESULT = n;

:}

|ELEMENTO\_ACCION:n

{:

RESULT = n;

:}

|ELEMENTO\_REFERENCIA:n

{:

RESULT = n;

:}

|ELEMENTO\_PATH:n

{:

RESULT = n;

:}

|ELEMENTO\_AUTOREPRODUCCION:n

{:

RESULT = n;

:};

ELEMENTO\_ID ::= r\_id:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2.substring(1,n2.length()-1), n1right, n1left);

:};

ELEMENTO\_NOMBRE ::= r\_nombre:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2.substring(1,n2.length()-1), n1right, n1left);

:};

ELEMENTO\_TIPO ::= r\_tipo:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2.substring(1,n2.length()-1), n1right, n1left);

:};

ELEMENTO\_POSX ::= r\_posx:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Integer.parseInt(n2), n1right, n1left);

:}

|r\_posx:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.decimal, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_POSY ::= r\_posy:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Integer.parseInt(n2), n1right, n1left);

:}

|r\_posy:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.decimal, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_ANCHO ::= r\_alto:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Integer.parseInt(n2), n1right, n1left);

:}

|r\_alto:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.decimal, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_ALTO ::= r\_ancho:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Integer.parseInt(n2), n1right, n1left);

:}

|r\_ancho:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.decimal, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_BORDE ::= r\_borde:n1 s\_equal booleano:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano, n2.equalsIgnoreCase("verdadero") ? true : false, n1right, n1left);

:};

ELEMENTO\_COLOR ::= r\_color:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2.substring(1,n2.length()-1), n1right, n1left);

:};

ELEMENTO\_FUENTE ::= r\_fuente:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2.substring(1,n2.length()-1), n1right, n1left);

:};

ELEMENTO\_TAM ::= r\_tamano:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Integer.parseInt(n2), n1right, n1left);

:}

|r\_tamano:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_NEGRITA ::= r\_negrita:n1 s\_equal booleano:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano, n2.equalsIgnoreCase("verdadero") ? true : false, n1right, n1left);

:};

ELEMENTO\_CURSIVA ::= r\_cursiva:n1 s\_equal booleano:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano, n2.equalsIgnoreCase("verdadero") ? true : false, n1right, n1left);

:};

ELEMENTO\_MAXIMO ::= r\_maximo:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Integer.parseInt(n2), n1right, n1left);

:}

|r\_maximo:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.entero, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_MINIMO ::= r\_minimo:n1 s\_equal entero:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano, Integer.parseInt(n2), n1right, n1left);

:}

|r\_minimo:n1 s\_equal decimal:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano, Double.parseDouble(n2), n1right, n1left);

:};

ELEMENTO\_REFERENCIA ::= r\_referencia:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2.substring(1,n2.length()-1), n1right, n1left);

:};

ELEMENTO\_ACCIONINICIAL ::= r\_accioninicial:n1 s\_equal s\_key\_open SENTENCIA\_LLAMADA:n2 s\_key\_close

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.identificador, n2, n1right, n1left);

:};

ELEMENTO\_ACCIONFINAL ::= r\_accionfinal:n1 s\_equal s\_key\_open SENTENCIA\_LLAMADA:n2 s\_key\_close

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.identificador, n2, n1right, n1left);

:};

ELEMENTO\_ACCION ::= r\_accion:n1 s\_equal s\_key\_open SENTENCIA\_LLAMADA:n2 s\_key\_close

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.identificador, n2, n1right, n1left);

:};

ELEMENTO\_PATH ::= r\_path:n1 s\_equal cadena:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena, n2, n1right, n1left);

:};

ELEMENTO\_AUTOREPRODUCCION ::= r\_autoreproduccion:n1 s\_equal booleano:n2

{:

RESULT = new GXML\_Elemento(n1, Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano, n2, n1right, n1left);

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EXPRESIONES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SENTENCIA\_LLAMADA ::= identificador:n1 s\_par\_open LISTA\_EXPRESIONES:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_LLAMADA");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:}

|identificador:n1 s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_LLAMADA");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EXPRESIONES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LISTA\_EXPRESIONES ::= LISTA\_EXPRESIONES:n1 s\_coma EXPRESION:n2

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(n2right));

nodo\_expresion.setColumna(String.valueOf(n2left));

nodo\_expresion.getHijos().add(n2);

RESULT = n1;

RESULT.getHijos().add(nodo\_expresion);

:}

|EXPRESION:n

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(nright));

nodo\_expresion.setColumna(String.valueOf(nleft));

nodo\_expresion.getHijos().add(n);

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_EXPRESIONES");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(nodo\_expresion);

:};

EXPRESION ::= EXPRESION\_ARITMETICA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_ARITMETICA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|EXPRESION\_RELACIONAL:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_RELACIONAL");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|EXPRESION\_UNARIA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_UNARIA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|EXPRESION\_LOGICA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_LOGICA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|s\_par\_open EXPRESION:n s\_par\_close

{:

RESULT = n;

:}

|DATO\_PRIMITIVO:n

{:

RESULT = n;

:};

EXPRESION\_ARITMETICA ::= EXPRESION:n1 s\_plus:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_minus:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_mul:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_div:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_mod:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_to:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_increment:o

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:}

|EXPRESION:n1 s\_decrement:o

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

EXPRESION\_UNARIA ::= s\_minus:o EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:}

|s\_plus:o EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

EXPRESION\_LOGICA ::= EXPRESION:n1 s\_or:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_and:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|s\_not:o EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

EXPRESION\_RELACIONAL ::= EXPRESION:n1 s\_greather:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_less:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_greather\_equal:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_less\_equal:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_compare:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_diferent:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:};

DATO\_PRIMITIVO ::= r\_nulo:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.nulo.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|booleano:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|entero:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.entero.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|decimal:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.decimal.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|caracter:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.caracter.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|cadena:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|identificador:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.identificador.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:};

* Gramática de FuncionScript:

start with INICIO;

INICIO ::= CUERPO\_FS:n

{:

raiz = new Nodo\_AST\_FS();

raiz.setEtiqueta("AST");

raiz.setValor("");

raiz.setFila(String.valueOf(nright));

raiz.setColumna(String.valueOf(nleft));

raiz.getHijos().add(n);

ejecutar\_AST();

:};

CUERPO\_FS ::= CUERPO\_FS:n1 SENTENCIA\_IMPORTAR:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

|CUERPO\_FS:n1 FUNCION:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

|CUERPO\_FS:n1 SENTENCIA:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

|SENTENCIA\_IMPORTAR:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("CUERPO\_FS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|FUNCION:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("CUERPO\_FS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|SENTENCIA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("CUERPO\_FS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

SENTENCIA\_IMPORTAR ::= r\_importar:n1 s\_par\_open cadena:n2 s\_par\_close s\_dotcoma

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_IMPORTAR");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION ::= r\_funcion:n identificador:n1 s\_par\_open LISTA\_PARAMETROS:n2 s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n3 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n2);

:}

|r\_funcion:n identificador:n1 s\_par\_open s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n2);

:};

LISTA\_SENTENCIAS ::= LISTA\_SENTENCIAS:n1 SENTENCIA:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

| SENTENCIA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_SENTENCIAS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

SENTENCIA ::= SENTENCIA\_DECLARACION:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_ASIGNACION:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_IMPRIMIR:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_SI:n

{:

RESULT = n;

:}

|SENTENCIA\_SELECCIONA:n

{:

RESULT = n;

:}

|SENTENCIA\_DETENER:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_LLAMADA:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_RETORNAR:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_LLAMADA\_FUNCIONES\_ARREGLO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_VENTANA:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_LEER\_GXML:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_CONTENEDOR:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_TEXTO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_CAJA\_TEXTO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_AREA\_TEXTO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_CONTROL\_NUMERICO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_DESPLEGABLE:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_BOTON:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_IMAGEN:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_REPRODUCTOR:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_VIDEO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_EVENTO\_ALCLIC:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_EVENTO\_ALCARGAR:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_EVENTO\_ALCERRAR:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_DESDE\_ARCHIVO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_INCREMENTO:n s\_dotcoma

{:

RESULT = n;

:}

|SENTENCIA\_DECREMENTO:n s\_dotcoma

{:

RESULT = n;

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INSTRUCCCIONES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//DECLACION

SENTENCIA\_DECLARACION ::= r\_var:n1 LISTA\_IDENTIFICADORES:n2 s\_equal VALOR:n3

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_DECLARACION");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

:}

|r\_var:n1 LISTA\_IDENTIFICADORES:n2 s\_dotcoma

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_DECLARACION");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:};

//ASIGNACION

SENTENCIA\_ASIGNACION ::= LISTA\_IDENTIFICADORES:n1 s\_equal VALOR:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_ASIGNACION");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|SENTENCIA\_ACCESO:n1 s\_equal VALOR:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_ASIGNACION");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:};

VALOR ::= EXPRESION:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|ARREGLO:n

{:

RESULT = n;

:}

|OBJETO:n

{:

RESULT = n;

:};

SENTENCIA\_IMPRIMIR ::= r\_imprimir:n s\_par\_open EXPRESION:n2 s\_par\_close

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(n2right));

nodo\_expresion.setColumna(String.valueOf(n2left));

nodo\_expresion.getHijos().add(n2);

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_IMPRIMIR");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(nodo\_expresion);

:};

SENTENCIA\_SELECCION ::= EXPRESION:n1 s\_ternario EXPRESION:n2 s\_dobledot EXPRESION:n3

{:

Nodo\_AST\_FS nodo\_expresion\_relacional = new Nodo\_AST\_FS();

nodo\_expresion\_relacional.setEtiqueta("EXPRESION\_RELACIONAL");

nodo\_expresion\_relacional.setValor("");

nodo\_expresion\_relacional.setFila(String.valueOf(n1right));

nodo\_expresion\_relacional.setColumna(String.valueOf(n1left));

nodo\_expresion\_relacional.getHijos().add(n1);

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SELECCION");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

:};

SENTENCIA\_SI ::= r\_si:i s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SI");

RESULT.setValor("");

RESULT.setFila(String.valueOf(iright));

RESULT.setColumna(String.valueOf(ileft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|r\_si:i s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close r\_sino s\_key\_open LISTA\_SENTENCIAS:n3 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SI");

RESULT.setValor("");

RESULT.setFila(String.valueOf(iright));

RESULT.setColumna(String.valueOf(ileft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

:}

|r\_si:i s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close LISTA\_SENTENCIAS\_SINO\_SI:n3

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SI");

RESULT.setValor("");

RESULT.setFila(String.valueOf(iright));

RESULT.setColumna(String.valueOf(ileft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

:}

|r\_si:i s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close LISTA\_SENTENCIAS\_SINO\_SI:n3 r\_sino s\_key\_open LISTA\_SENTENCIAS:n4 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SI");

RESULT.setValor("");

RESULT.setFila(String.valueOf(iright));

RESULT.setColumna(String.valueOf(ileft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

:};

SENTENCIA\_SELECCIONA ::= r\_selecciona:n s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS\_CASOS:n2 r\_defecto s\_dobledot s\_key\_open LISTA\_SENTENCIAS:n3 s\_key\_close s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SELECCIONA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

:}

|r\_selecciona:n s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS\_CASOS:n2 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SELECCIONA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:};

SENTENCIA\_CASO ::= r\_caso:n EXPRESION:n1 s\_dobledot s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CASOS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:};

SENTENCIA\_SINO\_SI ::= r\_sino r\_si:i s\_par\_open EXPRESION:n1 s\_par\_close s\_key\_open LISTA\_SENTENCIAS:n2 s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_SINO\_SI");

RESULT.setValor("");

RESULT.setFila(String.valueOf(iright));

RESULT.setColumna(String.valueOf(ileft));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:};

SENTENCIA\_DETENER ::= r\_detener:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_DETENER");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:};

SENTENCIA\_LLAMADA ::= identificador:n1 s\_par\_open LISTA\_EXPRESIONES:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_LLAMADA");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:}

|identificador:n1 s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_LLAMADA");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

SENTENCIA\_RETORNAR ::= r\_retornar:n EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_RETORNAR");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n1);

:};

SENTENCIA\_ACCESO ::= identificador:n1 s\_dot identificador:n2

{:

Nodo\_AST\_FS nodo\_id = new Nodo\_AST\_FS();

nodo\_id.setEtiqueta("IDENTIFICADOR");

nodo\_id.setValor(n2);

nodo\_id.setFila(String.valueOf(n2right));

nodo\_id.setColumna(String.valueOf(n2left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_ACCESO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_id);

:}

|identificador:n1 s\_cor\_open EXPRESION:n2 s\_cor\_close

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(n2right));

nodo\_expresion.setColumna(String.valueOf(n2left));

nodo\_expresion.getHijos().add(n2);

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_ACCESO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_expresion);

:};

SENTENCIA\_LLAMADA\_FUNCIONES\_ARREGLO ::= identificador:n1 LISTA\_LLAMADAS\_FUNCIONES\_ARREGLO:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_LLAMADA\_FUNCIONES\_ARREGLO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:};

LISTA\_LLAMADAS\_FUNCIONES\_ARREGLO ::= LISTA\_LLAMADAS\_FUNCIONES\_ARREGLO:n1 s\_dot FUNCION\_ARREGLO:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

| s\_dot FUNCION\_ARREGLO:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_LLAMADAS\_FUNCIONES\_ARREGLO");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

FUNCION\_ARREGLO ::= FUNCION\_DESCENDENTE:n

{:

RESULT = n;

:}

|FUNCION\_ASCENDENTE:n

{:

RESULT = n;

:}

|FUNCION\_INVERTIR:n

{:

RESULT = n;

:}

|FUNCION\_MAXIMO:n

{:

RESULT = n;

:}

|FUNCION\_MINIMO:n

{:

RESULT = n;

:}

|FUNCION\_FILTRAR:n

{:

RESULT = n;

:}

|FUNCION\_BUSCAR:n

{:

RESULT = n;

:}

|FUNCION\_MAP:n

{:

RESULT = n;

:}

|FUNCION\_REDUCE:n

{:

RESULT = n;

:}

|FUNCION\_TODOS:n

{:

RESULT = n;

:}

|FUNCION\_ALGUNO:n

{:

RESULT = n;

:}

|FUNCION\_OBTENER\_POR\_ETIQUETA:n

{:

RESULT = n;

:}

|FUNCION\_OBTENER\_POR\_ID:n

{:

RESULT = n;

:}

|FUNCION\_OBTENER\_POR\_NOMBRE:n

{:

RESULT = n;

:};

FUNCION\_DESCENDENTE ::= r\_descendente:n s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_DESCENDENTE");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:};

FUNCION\_ASCENDENTE ::= r\_ascendente:n s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_ASCENDENTE");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:};

FUNCION\_INVERTIR ::= r\_invertir:n s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_INVERTIR");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:};

FUNCION\_MAXIMO ::= r\_maximo:n s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_MAXIMO");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:};

FUNCION\_MINIMO ::= r\_minimo:n s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_MINIMO");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:};

FUNCION\_FILTRAR ::= r\_filtrar:n1 s\_par\_open identificador:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_FILTRAR");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_BUSCAR ::= r\_buscar:n1 s\_par\_open identificador:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_BUSCAR");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_MAP ::= r\_map:n1 s\_par\_open identificador:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_MAP");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_REDUCE ::= r\_reduce:n1 s\_par\_open identificador:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_REDUCE");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_TODOS ::= r\_todos:n1 s\_par\_open identificador:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_TODOS");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_ALGUNO ::= r\_alguno:n1 s\_par\_open identificador:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_ALGUNO");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_OBTENER\_POR\_ETIQUETA ::= r\_obteneretiqueta:n1 s\_par\_open cadena:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_OBTENER\_POR\_ETIQUETA");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_OBTENER\_POR\_ID ::= r\_obtenerid:n1 s\_par\_open cadena:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_OBTENER\_POR\_ID");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

FUNCION\_OBTENER\_POR\_NOMBRE ::= r\_obtenernombre:n1 s\_par\_open cadena:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("FUNCION\_OBTENER\_POR\_NOMBRE");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

SENTENCIA\_LEER\_GXML ::= r\_leergxml:n1 s\_par\_open cadena:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_LEER\_GXML");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:};

SENTENCIA\_CREAR\_VENTANA ::= r\_crearventana:n1 s\_par\_open cadena:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_VENTANA");

RESULT.setValor(n2);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

:};

SENTENCIA\_CREAR\_CONTENEDOR ::= identificador:n1 s\_dot r\_crearcontenedor s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma cadena:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_par\_close

{:

Nodo\_AST\_FS nodo\_color = new Nodo\_AST\_FS();

nodo\_color.setEtiqueta("COLOR");

nodo\_color.setValor(n4);

nodo\_color.setFila(String.valueOf(n4right));

nodo\_color.setColumna(String.valueOf(n4left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_CONTENEDOR");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_color);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

:};

SENTENCIA\_CREAR\_TEXTO ::= identificador:n1 s\_dot r\_creartexto s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma cadena:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_coma EXPRESION:n9 s\_par\_close

{:

Nodo\_AST\_FS nodo\_color = new Nodo\_AST\_FS();

nodo\_color.setEtiqueta("COLOR");

nodo\_color.setValor(n4);

nodo\_color.setFila(String.valueOf(n4right));

nodo\_color.setColumna(String.valueOf(n4left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_TEXTO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_color);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

RESULT.getHijos().add(n9);

:};

SENTENCIA\_CREAR\_CAJA\_TEXTO ::= identificador:n1 s\_dot r\_crearcajatexto s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma cadena:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_coma EXPRESION:n9 s\_coma EXPRESION:n10 s\_coma EXPRESION:n11 s\_coma EXPRESION:n12 s\_par\_close

{:

Nodo\_AST\_FS nodo\_color = new Nodo\_AST\_FS();

nodo\_color.setEtiqueta("COLOR");

nodo\_color.setValor(n6);

nodo\_color.setFila(String.valueOf(n6right));

nodo\_color.setColumna(String.valueOf(n6left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_CAJA\_TEXTO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_color);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

RESULT.getHijos().add(n9);

RESULT.getHijos().add(n10);

RESULT.getHijos().add(n11);

RESULT.getHijos().add(n12);

:};

SENTENCIA\_CREAR\_AREA\_TEXTO ::= identificador:n1 s\_dot r\_crearareatexto s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma cadena:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_coma EXPRESION:n9 s\_coma EXPRESION:n10 s\_coma EXPRESION:n11 s\_coma EXPRESION:n12 s\_par\_close

{:

Nodo\_AST\_FS nodo\_color = new Nodo\_AST\_FS();

nodo\_color.setEtiqueta("COLOR");

nodo\_color.setValor(n6);

nodo\_color.setFila(String.valueOf(n6right));

nodo\_color.setColumna(String.valueOf(n6left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_AREA\_TEXTO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_color);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

RESULT.getHijos().add(n9);

RESULT.getHijos().add(n10);

RESULT.getHijos().add(n11);

RESULT.getHijos().add(n12);

:};

SENTENCIA\_CREAR\_CONTROL\_NUMERICO ::= identificador:n1 s\_dot r\_crearcontrolnumerico s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_coma EXPRESION:n9 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_CONTROL\_NUMERICO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

RESULT.getHijos().add(n9);

:};

SENTENCIA\_CREAR\_DESPLEGABLE ::= identificador:n1 s\_dot r\_creardesplegable s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_DESPLEGABLE");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

:};

SENTENCIA\_CREAR\_BOTON ::= identificador:n1 s\_dot r\_crearboton s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma cadena:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_coma EXPRESION:n9 s\_coma EXPRESION:n10 s\_par\_close

{:

Nodo\_AST\_FS nodo\_color = new Nodo\_AST\_FS();

nodo\_color.setEtiqueta("COLOR");

nodo\_color.setValor(n4);

nodo\_color.setFila(String.valueOf(n4right));

nodo\_color.setColumna(String.valueOf(n4left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_BOTON");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_color);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

RESULT.getHijos().add(n9);

RESULT.getHijos().add(n10);

:}

|identificador:n1 s\_dot r\_crearboton s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma cadena:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_coma EXPRESION:n8 s\_coma EXPRESION:n9 s\_par\_close

{:

Nodo\_AST\_FS nodo\_color = new Nodo\_AST\_FS();

nodo\_color.setEtiqueta("COLOR");

nodo\_color.setValor(n4);

nodo\_color.setFila(String.valueOf(n4right));

nodo\_color.setColumna(String.valueOf(n4left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_BOTON");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_color);

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

RESULT.getHijos().add(n8);

RESULT.getHijos().add(n9);

:};

SENTENCIA\_CREAR\_IMAGEN ::= identificador:n1 s\_dot r\_crearimagen s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_par\_close

{:

RESULT.setEtiqueta("SENTENCIA\_CREAR\_IMAGEN");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

:};

SENTENCIA\_CREAR\_REPRODUCTOR ::= identificador:n1 s\_dot r\_crearreproductor s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_par\_close

{:

RESULT.setEtiqueta("SENTENCIA\_CREAR\_REPRODUCTOR");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

:};

SENTENCIA\_CREAR\_VIDEO ::= identificador:n1 s\_dot r\_crearvideo s\_par\_open EXPRESION:n2 s\_coma EXPRESION:n3 s\_coma EXPRESION:n4 s\_coma EXPRESION:n5 s\_coma EXPRESION:n6 s\_coma EXPRESION:n7 s\_par\_close

{:

RESULT.setEtiqueta("SENTENCIA\_CREAR\_VIDEO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

RESULT.getHijos().add(n3);

RESULT.getHijos().add(n4);

RESULT.getHijos().add(n5);

RESULT.getHijos().add(n6);

RESULT.getHijos().add(n7);

:};

SENTENCIA\_EVENTO\_ALCLIC ::= identificador:n1 s\_dot r\_clic s\_par\_open SENTENCIA\_LLAMADA:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_EVENTO\_ALCLIC");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:};

SENTENCIA\_EVENTO\_ALCARGAR ::= identificador:n s\_dot r\_cargar s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_EVENTO\_ALCARGAR");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:}

| identificador:n1 s\_dot r\_cargar s\_par\_open SENTENCIA\_LLAMADA:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_EVENTO\_ALCARGAR");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:};

SENTENCIA\_EVENTO\_ALCERRAR ::= identificador:n s\_dot r\_cerrar s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_EVENTO\_ALCERRAR");

RESULT.setValor(n);

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

:}

|identificador:n1 s\_dot r\_cerrar s\_par\_open SENTENCIA\_LLAMADA:n2 s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_EVENTO\_ALCERRAR");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n2);

:};

SENTENCIA\_CREAR\_DESDE\_ARCHIVO ::= identificador:n1 s\_dot r\_creararray s\_par\_open s\_par\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_DESDE\_ARCHIVO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:}

|r\_creararray:n1 s\_par\_open cadena:n2 s\_par\_close

{:

Nodo\_AST\_FS nodo\_archivo = new Nodo\_AST\_FS();

nodo\_archivo.setEtiqueta("PATH");

nodo\_archivo.setValor(n2);

nodo\_archivo.setFila(String.valueOf(n2right));

nodo\_archivo.setColumna(String.valueOf(n2left));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_CREAR\_DESDE\_ARCHIVO");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_archivo);

:};

//PRODUCCIONES AUXILIARES

LISTA\_PARAMETROS ::= LISTA\_PARAMETROS:n1 s\_coma r\_var identificador:n2

{:

Nodo\_AST\_FS nodo\_identificador = new Nodo\_AST\_FS();

nodo\_identificador.setEtiqueta("IDENTIFICADOR");

nodo\_identificador.setValor(n2);

nodo\_identificador.setFila(String.valueOf(n2right));

nodo\_identificador.setColumna(String.valueOf(n2left));

n1.getHijos().add(nodo\_identificador);

RESULT = n1;

:}

|r\_var identificador:n

{:

Nodo\_AST\_FS nodo\_identificador = new Nodo\_AST\_FS();

nodo\_identificador.setEtiqueta("IDENTIFICADOR");

nodo\_identificador.setValor(n);

nodo\_identificador.setFila(String.valueOf(nright));

nodo\_identificador.setColumna(String.valueOf(nleft));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_PARAMETROS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(nodo\_identificador);

:};

LISTA\_IDENTIFICADORES ::= LISTA\_IDENTIFICADORES:n1 s\_coma identificador:n2

{:

Nodo\_AST\_FS nodo\_identificador = new Nodo\_AST\_FS();

nodo\_identificador.setEtiqueta("IDENTIFICADOR");

nodo\_identificador.setValor(n2);

nodo\_identificador.setFila(String.valueOf(n2right));

nodo\_identificador.setColumna(String.valueOf(n2left));

n1.getHijos().add(nodo\_identificador);

RESULT = n1;

:}

|identificador:n

{:

Nodo\_AST\_FS nodo\_identificador = new Nodo\_AST\_FS();

nodo\_identificador.setEtiqueta("IDENTIFICADOR");

nodo\_identificador.setValor(n);

nodo\_identificador.setFila(String.valueOf(nright));

nodo\_identificador.setColumna(String.valueOf(nleft));

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_IDENTIFICADORES");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(nodo\_identificador);

:};

LISTA\_SENTENCIAS\_SINO\_SI ::= LISTA\_SENTENCIAS\_SINO\_SI:n1 SENTENCIA\_SINO\_SI:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

|SENTENCIA\_SINO\_SI:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_SENTENCIA\_SINO\_SI");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

LISTA\_SENTENCIAS\_CASOS ::= LISTA\_SENTENCIAS\_CASOS:n1 SENTENCIA\_CASO:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

|SENTENCIA\_CASO:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_SENTENCIA\_CASOS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

SENTENCIA\_INCREMENTO ::= identificador:n1 s\_increment

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_INCREMENTO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:}

|SENTENCIA\_ACCESO:n1 s\_increment

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_INCREMENTO");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

SENTENCIA\_DECREMENTO ::= identificador:n1 s\_decrement

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_DECREMENTO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

:}

|SENTENCIA\_ACCESO:n1 s\_decrement

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("SENTENCIA\_DECREMENTO");

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EXPRESIONES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LISTA\_EXPRESIONES ::= LISTA\_EXPRESIONES:n1 s\_coma EXPRESION:n2

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(n2right));

nodo\_expresion.setColumna(String.valueOf(n2left));

nodo\_expresion.getHijos().add(n2);

RESULT = n1;

RESULT.getHijos().add(nodo\_expresion);

:}

|EXPRESION:n

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(nright));

nodo\_expresion.setColumna(String.valueOf(nleft));

nodo\_expresion.getHijos().add(n);

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_EXPRESIONES");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(nodo\_expresion);

:};

EXPRESION ::= EXPRESION\_ARITMETICA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_ARITMETICA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|EXPRESION\_RELACIONAL:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_RELACIONAL");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|EXPRESION\_UNARIA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_UNARIA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|EXPRESION\_LOGICA:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("EXPRESION\_LOGICA");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:}

|s\_par\_open EXPRESION:n s\_par\_close

{:

RESULT = n;

:}

|DATO\_PRIMITIVO:n

{:

RESULT = n;

:}

|SENTENCIA\_SELECCION:n

{:

RESULT = n;

:}

|SENTENCIA\_LLAMADA:n

{:

RESULT = n;

:}

|SENTENCIA\_ACCESO:n

{:

RESULT = n;

:}

|SENTENCIA\_LLAMADA\_FUNCIONES\_ARREGLO:n

{:

RESULT = n;

:}

|SENTENCIA\_LEER\_GXML:n

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_VENTANA:n

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_CONTENEDOR:n

{:

RESULT = n;

:}

|SENTENCIA\_CREAR\_BOTON:n

{:

RESULT = n;

:};

EXPRESION\_ARITMETICA ::= EXPRESION:n1 s\_plus:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_minus:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_mul:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_div:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_mod:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_to:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_increment:o

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:}

|EXPRESION:n1 s\_decrement:o

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

EXPRESION\_UNARIA ::= s\_minus:o EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:}

|s\_plus:o EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

EXPRESION\_LOGICA ::= EXPRESION:n1 s\_or:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_and:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|s\_not:o EXPRESION:n1

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

:};

EXPRESION\_RELACIONAL ::= EXPRESION:n1 s\_greather:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_less:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_greather\_equal:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_less\_equal:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_compare:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:}

|EXPRESION:n1 s\_diferent:o EXPRESION:n2

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(o);

RESULT.setValor("");

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(n1);

RESULT.getHijos().add(n2);

:};

DATO\_PRIMITIVO ::= r\_nulo:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.nulo.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|booleano:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.booleano.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|entero:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.entero.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|decimal:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.decimal.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|caracter:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.caracter.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|cadena:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.cadena.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:}

|identificador:v

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta(Tabla\_Enums.tipo\_primitivo\_Simbolo.identificador.toString());

RESULT.setValor(v);

RESULT.setFila(String.valueOf(vright));

RESULT.setColumna(String.valueOf(vleft));

:};

ARREGLO ::= s\_cor\_open LISTA\_EXPRESIONES:n s\_cor\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("ARREGLO");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

OBJETO ::= s\_key\_open LISTA\_ATRIBUTOS:n s\_key\_close

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("OBJETO");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

LISTA\_ATRIBUTOS ::= LISTA\_ATRIBUTOS:n1 s\_coma ATRIBUTO:n2

{:

RESULT = n1;

RESULT.getHijos().add(n2);

:}

|ATRIBUTO:n

{:

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("LISTA\_ATRIBUTOS");

RESULT.setValor("");

RESULT.setFila(String.valueOf(nright));

RESULT.setColumna(String.valueOf(nleft));

RESULT.getHijos().add(n);

:};

ATRIBUTO ::= identificador:n1 s\_dobledot EXPRESION:n2

{:

Nodo\_AST\_FS nodo\_expresion = new Nodo\_AST\_FS();

nodo\_expresion.setEtiqueta("EXPRESION");

nodo\_expresion.setValor("");

nodo\_expresion.setFila(String.valueOf(n2right));

nodo\_expresion.setColumna(String.valueOf(n2left));

nodo\_expresion.getHijos().add(n2);

RESULT = new Nodo\_AST\_FS();

RESULT.setEtiqueta("ATRIBUTO");

RESULT.setValor(n1);

RESULT.setFila(String.valueOf(n1right));

RESULT.setColumna(String.valueOf(n1left));

RESULT.getHijos().add(nodo\_expresion);

:};