Procesadores de Lenguajes

Memoria de proyecto — Hito 2:Analizador Sintáctico

Grupo 14

Rodrigo Souto Santos

Leonardo Prado de Souza

Juan Andrés Hibjan Cardona

Izan Rodrigo Sanz

Grado en Ingeniería informática
Facultad de Informática
Universidad Complutense de Madrid

2024–2025

Índice general

1. Especificación de la Sintaxis Abstracta **2**
1.1. Géneros de nodos . . . 2
1.2. Funciones constructoras de nodos . . . 2
1.2.1. Declaraciones . . . 2
1.2.2. Tipos . . . 2
1.2.3. Instrucciones . . . 3
1.2.4. Expresiones . . . 3

2. Especificación del constructor de ASTs **4**
2.1. Gramática s-atribuida . . . 4
2.1.1. Declaraciones . . . 4
2.1.2. Tipos . . . 4
2.1.3. Instrucciones . . . 5
2.1.4. Expresiones . . . 6
2.1.5. Operadores . . . 7
2.2. Funciones semánticas auxiliares . . . 7
2.2.1. Expresiones unarias . . . 7
2.2.2. Expresiones binarias . . . 7

3. Acondicionamiento del constructor de ASTs **8**
3.1. Gramática s-atribuida . . . 8
3.1.1. Declaraciones . . . 8
3.1.2. Tipos . . . 9
3.1.3. Instrucciones . . . 9
3.1.4. Expresiones . . . 10
3.1.5. Operadores . . . 12
3.2. Funciones semánticas auxiliares . . . 12
3.2.1. Expresiones unarias . . . 12
3.2.2. Expresiones binarias . . . 12

4. Especificación del procesamiento 'impresión bonita' **13**

1 Especificación de la Sintaxis Abstracta

1.1. Géneros de nodos

Bloq
SecDecs, *LDecs*, *Dec*
SecIs, *LIs*, *I*
ParamFs, *LParamFs*, *ParamF*
ParamRs, *LParamRs*
LCampos
TipoNom, *Tipo*
Exp

1.2. Funciones constructoras de nodos

$bloque : SecDecs \times SecIs \longrightarrow Bloq$

1.2.1. Declaraciones

$si\_decs : LDecs \longrightarrow SecDecs$
$no\_decs : \longrightarrow SecDecs$
$muchas\_decs : LDecs \times Dec \longrightarrow LDecs$
$una\_dec : Dec \longrightarrow LDecs$
$dec\_base : TipoNom \longrightarrow Dec$
$dec\_type : TipoNom \longrightarrow Dec$
$dec\_proc : \mathbf{string} \times ParamFs \times Bloq \longrightarrow Dec$
$si\_params\_f : LParamFs \longrightarrow ParamFs$
$no\_params\_f : \longrightarrow ParamFs$
$muchos\_params\_f : LParamFs \times ParamF \longrightarrow LParamFs$
$un\_param\_f : ParamF \longrightarrow LParamFs$
$si\_refparam\_f : Tipo \times \mathbf{string} \longrightarrow ParamF$
$no\_refparam\_f : Tipo \times \mathbf{string} \longrightarrow ParamF$

1.2.2. Tipos

$tipo\_nombre : Tipo \times \mathbf{string} \longrightarrow TipoNom$
$tipo\_array : Tipo \times \mathbf{string} \longrightarrow Tipo$
$tipo\_indir : Tipo \longrightarrow Tipo$
$tipo\_struct : LCampos \longrightarrow Tipo$
$tipo\_int : \longrightarrow Tipo$
$tipo\_real : \longrightarrow Tipo$
$tipo\_bool : \longrightarrow Tipo$
$tipo\_string : \longrightarrow Tipo$
$tipo\_type : \mathbf{string} \longrightarrow Tipo$
$muchos\_campos : LCampos \times TipoNom \longrightarrow LCampos$
$un\_campo : TipoNom \longrightarrow LCampos$

1.2.3. Instrucciones

$si\_ins : LIs \longrightarrow SecIs$
$no\_ins : \longrightarrow SecIs$
$muchas\_ins : LIs \times I \longrightarrow LIs$
$una\_ins : I \longrightarrow LIs$
$ins\_eval : Exp \longrightarrow I$
$ins\_if : Exp \times Bloq \longrightarrow I$
$ins\_if\_else : I \times Bloq \longrightarrow I$
$ins\_while : Exp \times Bloq \longrightarrow I$
$ins\_read : Exp \longrightarrow I$
$ins\_write : Exp \longrightarrow I$
$ins\_nl : \longrightarrow I$
$ins\_new : Exp \longrightarrow I$
$ins\_delete : Exp \longrightarrow I$
$ins\_call : \textbf{string} \times ParamRs \longrightarrow I$
$ins\_bloque : Bloq \longrightarrow I$
$si\_params\_r : LParamRs \longrightarrow ParamRs$
$no\_params\_r : \longrightarrow ParamRs$
$muchos\_params\_r : LParamRs \times Exp \longrightarrow LParamRs$
$un\_param\_r : Exp \longrightarrow LParamRs$

1.2.4. Expresiones

$exp\_asig : Exp \times Exp \longrightarrow Exp$
$exp\_menor : Exp \times Exp \longrightarrow Exp$
$exp\_menor\_ig : Exp \times Exp \longrightarrow Exp$
$exp\_mayor : Exp \times Exp \longrightarrow Exp$
$exp\_mayor\_ig : Exp \times Exp \longrightarrow Exp$
$exp\_ig : Exp \times Exp \longrightarrow Exp$
$exp\_dist : Exp \times Exp \longrightarrow Exp$
$exp\_suma : Exp \times Exp \longrightarrow Exp$
$exp\_resta : Exp \times Exp \longrightarrow Exp$
$exp\_and : Exp \times Exp \longrightarrow Exp$
$exp\_or : Exp \times Exp \longrightarrow Exp$
$exp\_mul : Exp \times Exp \longrightarrow Exp$
$exp\_div : Exp \times Exp \longrightarrow Exp$
$exp\_mod : Exp \times Exp \longrightarrow Exp$
$exp\_menos : Exp \longrightarrow Exp$
$exp\_not : Exp \longrightarrow Exp$
$exp\_index : Exp \longrightarrow Exp$
$exp\_reg : Exp \longrightarrow Exp$
$exp\_indir : Exp \longrightarrow Exp$
$exp\_entero : \textbf{string} \longrightarrow Exp$
$exp\_real : \textbf{string} \longrightarrow Exp$
$exp\_true : \longrightarrow Exp$
$exp\_false : \longrightarrow Exp$
$exp\_cadena : \textbf{string} \longrightarrow Exp$
$exp\_iden : \textbf{string}\ Exp$
$exp\_null : \longrightarrow Exp$

2 Especificación del constructor de ASTs

2.1. Gramática s-atribuida

programa $\longrightarrow$ *bloque*
  programa.**a** = *bloque*.**a**
bloque $\longrightarrow$ {*seccion_declaraciones_opt seccion_instrucciones_opt*}
  bloque.**a** = *bloq*(*seccion_declaraciones_opt*.**a**, *seccion_instrucciones_opt*.**a**)

2.1.1. Declaraciones

seccion_declaraciones_opt $\longrightarrow$ *seccion_declaraciones* &&
  seccion_declaraciones_opt.**a** = *si_decs*(*seccion_declaraciones*)
seccion_declaraciones_opt $\longrightarrow$ ϵ
  seccion_declaraciones_opt.**a** = *no_decs*()
seccion_declaraciones $\longrightarrow$ *seccion_declaraciones* ; *declaracion*
  seccion_declaraciones$_0$.**a** = *muchas_decs*(*seccion_declaraciones*$_1$.**a**, *declaracion*.**a**)
seccion_declaraciones $\longrightarrow$ *declaracion*
  seccion_declaraciones.**a** = *una_dec*(*declaracion*.**a**)
declaracion $\longrightarrow$ *tipo_nombre*
  declaracion.**a** = *dec_base*(*tipo_nombre*.**a**)
declaracion $\longrightarrow$ **type** *tipo_nombre*
  declaracion.**a** = *dec_type*(*tipo_nombre*.**a**)
declaracion $\longrightarrow$ **proc identificador** *parametros_formales bloque*
  declaracion.**a** = *dec_proc*(**identificador.lex**, *parametros_formales*.**a**, *bloque*.**a**)
parametros_formales $\longrightarrow$ (*lista_parametros_opt*)
  parametros_formales.**a** = *lista_parametros_opt*.**a**
lista_parametros_opt $\longrightarrow$ *lista_parametros*
  lista_parametros_opt.**a** = *si_params_f*(*lista_parametros*.**a**)
lista_parametros_opt $\longrightarrow$ ϵ
  lista_parametros_opt.**a** = *no_params_f*()
lista_parametros $\longrightarrow$ *lista_parametros* , *parametro*
  lista_parametros$_0$.**a** = *muchos_params_f*(*lista_parametros*$_1$.**a**, *parametro*.**a**)
lista_parametros $\longrightarrow$ *parametro*
  lista_parametros.**a** = *un_param_f*(*parametro*.**a**)
parametro $\longrightarrow$ *tipo* & **identificador**
  parametro.**a** = *si_refparam_f*(*tipo*.**a**, **identificador.lex**)
parametro $\longrightarrow$ *tipo* **identificador**
  parametro.**a** = *no_refparam_f*(*tipo*.**a**, **identificador.lex**)

2.1.2. Tipos

tipo_nombre $\longrightarrow$ *tipo* **identificador**
  tipo_nombre.**a** = *tipo_nombre*(*tipo*.**a**, **identificador.lex**)
tipo $\longrightarrow$ *tipo*0
  tipo.**a** = *tipo*0.**a**
*tipo*0 $\longrightarrow$ *tipo*0 [**literalEntero**]
  *tipo*0$_0$.**a** = *tipo_array*(*tipo*0$_1$.**a**, **literalEntero.lex**)
*tipo*0 $\longrightarrow$ *tipo*1
  *tipo*0.**a** = *tipo*1.**a**
*tipo*1 $\longrightarrow$ ^*tipo*1
  *tipo*1$_0$.**a** = *tipo_indir*(*tipo*1$_1$.**a**)
*tipo*1 $\longrightarrow$ *tipo_base*
  *tipo*1.**a** = *tipo_base*.**a**

tipo_base $\longrightarrow$ **struct** {*lista_campos*}
    tipo_base.**a** = *tipo_struct*(*lista_campos*.**a**)
tipo_base $\longrightarrow$ **int**
    tipo_base.**a** = *tipo_int*()
tipo_base $\longrightarrow$ **real**
    tipo_base.**a** = *tipo_real*()
tipo_base $\longrightarrow$ **bool**
    tipo_base.**a** = *tipo_bool*()
tipo_base $\longrightarrow$ **string**
    tipo_base.**a** = *tipo_string*()
tipo_base $\longrightarrow$ **identificador**
    tipo_base.**a** = *tipo_type*()
lista_campos $\longrightarrow$ *lista_campos* , *tipo_nombre*
    lista_campos$_0$.**a** = *muchos_campos*(*lista_campos*$_1$.**a**, *tipo_nombre*.**a**)
lista_campos $\longrightarrow$ *tipo_nombre*
    lista_campos.**a** = *un_campo*(*tipo_nombre*.**a**)

2.1.3. Instrucciones

seccion_instrucciones_opt $\longrightarrow$ *seccion_instrucciones*
    seccion_instrucciones_opt.**a** = *si_ins*(*seccion_instrucciones*)
seccion_instrucciones_opt $\longrightarrow$ ϵ
    sseccion_instrucciones_opt.**a** = *no_ins*()
seccion_instrucciones $\longrightarrow$ *lista_instrucciones*
    seccion_instrucciones.**a** = *lista_instrucciones*.**a**
lista_instrucciones $\longrightarrow$ *lista_instrucciones* ; *instruccion*
    lista_instrucciones$_0$.**a** = *muchas_ins*(*lista_instrucciones*$_1$.**a**, *instruccion*.**a**)
lista_instrucciones $\longrightarrow$ *instruccion*
    lista_instrucciones.**a** = *una_ins*(*instruccion*.**a**)
instruccion $\longrightarrow$ @ *expresion*
    instruccion.**a** = *ins_eval*(*expresion*.**a**)
instruccion $\longrightarrow$ *if_ins*
    instruccion.**a** = *if_ins*.**a**
instruccion $\longrightarrow$ *if_ins* **else** *bloque*
    instruccion.**a** = *ins_if_else*(*if_ins*.**a**, *bloque*$_1$.**a**)
instruccion $\longrightarrow$ **while** *expresion bloque*
    instruccion.**a** = *ins_while*(*expresion*.**a**, *bloque*.**a**)
instruccion $\longrightarrow$ **read** *expresion*
    instruccion.**a** = *ins_read*(*expresion*.**a**)
instruccion $\longrightarrow$ **write** *expresion*
    instruccion.**a** = *ins_write*(*expresion*.**a**)
instruccion $\longrightarrow$ **nl**
    instruccion.**a** = *ins_nl*()
instruccion $\longrightarrow$ **new** *expresion*
    instruccion.**a** = *ins_new*(*expresion*.**a**)
instruccion $\longrightarrow$ **delete** *expresion*
    instruccion.**a** = *ins_delete*(*expresion*.**a**)
instruccion $\longrightarrow$ **call identificador** *parametros_reales*
    instruccion.**a** = *ins_call*(**identificador.lex**, *parametros_reales*.**a**)
instruccion $\longrightarrow$ *bloque*
    instruccion.**a** = *ins_bloque*(*bloque*.**a**)
if_ins $\longrightarrow$ **if** *expresion bloque*
    if_ins.**a** = *ins_if*(*expresion*.**a**, *bloque*.**a**)
parametros_reales $\longrightarrow$ (*lista_expresiones_opt*)
    parametros_reales.**a** = *lista_expresiones_opt*.**a**
lista_expresiones_opt $\longrightarrow$ *lista_expresiones*
    lista_expresiones_opt.**a** = *si_params_r*(*lista_expresiones*.**a**)
lista_expresiones_opt $\longrightarrow$ ϵ
    lista_expresiones_opt.**a** = *no_params_r*()
lista_expresiones $\longrightarrow$ *lista_expresiones* , *expresion*

$lista\_expresiones_0.\mathbf{a} = muchos\_params\_r(lista\_expresiones_1.\mathbf{a},\ expresion.\mathbf{a})$

$lista\_expresiones \longrightarrow expresion$

$lista\_expresiones.\mathbf{a} = un\_param\_r(expresion.\mathbf{a})$

2.1.4. Expresiones

$expresion \longrightarrow E0$

$expresion.\mathbf{a} = E0.\mathbf{a}$

$E0 \longrightarrow E1 = E0$

$E0_0.\mathbf{a} = mkopbin(" = ",\ E1.\mathbf{a},\ E0_1.\mathbf{a})$

$E0 \longrightarrow E1$

$E0.\mathbf{a} = E1.\mathbf{a}$

$E1 \longrightarrow E1\ op\_relacional\ E2$

$E1_0.\mathbf{a} = mkopbin(op\_relacional.\mathbf{op},\ E1_1.\mathbf{a},\ E2.\mathbf{a})$

$E1 \longrightarrow E2$

$E1.\mathbf{a} = E2.\mathbf{a}$

$E2 \longrightarrow E2 + E3$

$E2_0.\mathbf{a} = mkopbin(" + ",\ E2_1.\mathbf{a},\ E3.\mathbf{a})$

$E2 \longrightarrow E3 - E3$

$E2.\mathbf{a} = mkopbin(" - ",\ E3_0.\mathbf{a},\ E3_1.\mathbf{a})$

$E2 \longrightarrow E3$

$E2.\mathbf{a} = E3.\mathbf{a}$

$E3 \longrightarrow E4\ \mathbf{and}\ E3$

$E3_0.\mathbf{a} = mkopbin("and", E4.\mathbf{a},\ E3_1.\mathbf{a})$

$E3 \longrightarrow E4\ \mathbf{or}\ E4$

$E3.\mathbf{a} = mkopbin("or", E4_0.\mathbf{a},\ E4_1.\mathbf{a})$

$E3 \longrightarrow E4$

$E3.\mathbf{a} = E4.\mathbf{a}$

$E4 \longrightarrow E4\ op\_mult\ E5$

$E4_0.\mathbf{a} = mkopbin(op\_mult.\mathbf{op},\ E4_1.\mathbf{a},\ E5.\mathbf{a})$

$E4 \longrightarrow E5$

$E4.\mathbf{a} = E5.\mathbf{a}$

$E5 \longrightarrow - E5$

$E5_0.\mathbf{a} = mkopun(" - ", E5_1.\mathbf{a})$

$E5 \longrightarrow \mathbf{not}\ E5$

$E5_0.\mathbf{a} = mkopun("not", E5_1.\mathbf{a})$

$E5 \longrightarrow E6$

$E5.\mathbf{a} = E6.\mathbf{a}$

$E6 \longrightarrow E6\ op\_dirs$

$E6_0.\mathbf{a} = mkopun(op\_dirs.\mathbf{op},\ E6_1.\mathbf{a})$

$E6 \longrightarrow E7$

$E6.\mathbf{a} = E7.\mathbf{a}$

$E7 \longrightarrow expresion\_basica$

$E7.\mathbf{a} = expresion\_basica.\mathbf{a}$

$E7 \longrightarrow (E0)$

$E7.\mathbf{a} = E0.\mathbf{a}$

$expresion\_basica \longrightarrow \mathbf{literalEntero}$

$expresion\_basica.\mathbf{a} = exp\_entero(\mathbf{literalEntero.lex})$

$expresion\_basica \longrightarrow \mathbf{literalReal}$

$expresion\_basica.\mathbf{a} = exp\_real(\mathbf{literalReal.lex})$

$expresion\_basica \longrightarrow \mathbf{true}$

$expresion\_basica.\mathbf{a} = exp\_true()$

$expresion\_basica \longrightarrow \mathbf{false}$

$expresion\_basica.\mathbf{a} = exp\_false()$

$expresion\_basica \longrightarrow \mathbf{literalCadena}$

$expresion\_basica.\mathbf{a} = exp\_cadena(\mathbf{literalCadena.lex})$

$expresion\_basica \longrightarrow \mathbf{identificador}$

$expresion\_basica.\mathbf{a} = exp\_iden(\mathbf{identificador.lex})$

$expresion\_basica \longrightarrow \mathbf{null}$

$expresion\_basica.\mathbf{a} = exp\_null()$

2.1.5. Operadores

$op\_relacional \longrightarrow <$
$\quad op\_relacional.\mathbf{op} = \text{" < "}$
$op\_relacional \longrightarrow <=$
$\quad op\_relacional.\mathbf{op} = \text{" <= "}$
$op\_relacional \longrightarrow >$
$\quad op\_relacional.\mathbf{op} = \text{" > "}$
$op\_relacional \longrightarrow >=$
$\quad op\_relacional.\mathbf{op} = \text{" >= "}$
$op\_relacional \longrightarrow ==$
$\quad op\_relacional.\mathbf{op} = \text{" == "}$
$op\_relacional \longrightarrow !=$
$\quad op\_relacional.\mathbf{op} = \text{"!= "}$
$op\_mult \longrightarrow *$
$\quad op\_mult.\mathbf{op} = \text{" * "}$
$op\_mult \longrightarrow /$
$\quad op\_mult.\mathbf{op} = \text{"/"}$
$op\_mult \longrightarrow \%$
$\quad op\_mult.\mathbf{op} = \text{" \%"}$
$op\_dirs \longrightarrow [expresion]$
$\quad op\_dirs.\mathbf{op} = \text{"}index\text{"}$
$op\_dirs \longrightarrow .\ \mathbf{identificador}$
$\quad op\_dirs.\mathbf{op} = \text{"}reg\text{"}$
$op\_dirs \longrightarrow \hat{}$
$\quad op\_dirs.\mathbf{op} = \text{"\^{}"}$

2.2. Funciones semánticas auxiliares

2.2.1. Expresiones unarias

fun $mkopun(op,\ opnd)$:
$\quad op = \text{" - "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_menos}(opnd)$
$\quad op = \text{"}not\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_not}(opnd)$
$\quad op = \text{"}index\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_index}(opnd)$
$\quad op = \text{"}reg\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_reg}(opnd)$
$\quad op = \text{"\^{}"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_indir}(opnd)$

2.2.2. Expresiones binarias

fun $mkopbin(op,\ opnd1,\ opnd2)$:
$\quad op = \text{" = "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_asig}(opnd1,\ opnd2)$
$\quad op = \text{" < "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_menor}(opnd1,\ opnd2)$
$\quad op = \text{" <= "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_menor\_ig}(opnd1,\ opnd2)$
$\quad op = \text{" > "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mayor}(opnd1,\ opnd2)$
$\quad op = \text{" >= "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mayor\_ig}(opnd1,\ opnd2)$
$\quad op = \text{" == "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_ig}(opnd1,\ opnd2)$
$\quad op = \text{"!= "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_dist}(opnd1,\ opnd2)$
$\quad op = \text{" + "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_suma}(opnd1,\ opnd2)$
$\quad op = \text{" - "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_resta}(opnd1,\ opnd2)$
$\quad op = \text{"}and\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_and}(opnd1,\ opnd2)$
$\quad op = \text{"}or\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_or}(opnd1,\ opnd2)$
$\quad op = \text{" * "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mul}(opnd1,\ opnd2)$
$\quad op = \text{"/"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_div}(opnd1,\ opnd2)$
$\quad op = \text{" \%"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mod}(opnd1,\ opnd2)$

3 Acondicionamiento del constructor de ASTs

3.1. Gramática s-atribuida

programa $\longrightarrow$ *bloque*
  programa.**a** = *bloque*.**a**
bloque $\longrightarrow$ {*seccion_declaraciones_opt* *seccion_instrucciones_opt*}
  bloque.**a** = *bloq*(*seccion_declaraciones_opt*.**a**, *seccion_instrucciones_opt*.**a**)

3.1.1. Declaraciones

seccion_declaraciones_opt $\longrightarrow$ *seccion_declaraciones* &&
  seccion_declaraciones_opt.**a** = *si_decs*(*seccion_declaraciones*)
seccion_declaraciones_opt $\longrightarrow$ ϵ
  seccion_declaraciones_opt.**a** = *no_decs*()
seccion_declaraciones $\longrightarrow$ *declaracion resto_sd*
  resto_sd.**ah** = *una_dec*(*declaracion*.**a**)
  seccion_declaraciones_opt.**a** = *resto_sd*.**a**
resto_sd $\longrightarrow$; *declaracion resto_sd*
  $resto\_sd_1$.**ah** = *muchas_decs*($resto\_sd_0$.**ah**, *declaracion*.**a**)
  $resto\_sd_0$.**a** = $resto\_sd_1$.**a**
resto_sd $\longrightarrow$ ϵ
  resto_sd.**a** = *resto_sd*.**ah**
declaracion $\longrightarrow$ *tipo_nombre*
  declaracion.**a** = *dec_base*(*tipo_nombre*.**a**)
declaracion $\longrightarrow$ **type** *tipo_nombre*
  declaracion.**a** = *dec_type*(*tipo_nombre*.**a**)
declaracion $\longrightarrow$ **proc identificador** *parametros_formales bloque*
  declaracion.**a** = *dec_proc*(**identificador.lex**, *parametros_formales*.**a**, *bloque*.**a**)
parametros_formales $\longrightarrow$ (*lista_parametros_opt*)
  parametros_formales.**a** = *lista_parametros_opt*.**a**
lista_parametros_opt $\longrightarrow$ *lista_parametros*
  lista_parametros_opt.**a** = *si_params_f*(*lista_parametros*.**a**)
lista_parametros_opt $\longrightarrow$ ϵ
  lista_parametros_opt.**a** = *no_params_f*()
lista_parametros $\longrightarrow$ *parametro resto_lp*
  resto_lp.**ah** = *un_param_f*(*parametro*.**a**)
  lista_parametros.**a** = *resto_lp*.**a**
resto_lp $\longrightarrow$, *parametro resto_lp*
  $resto\_lp_1$.**ah** = *muchos_params_f*($resto\_lp_0$.**ah**, *parametro*.**a**)
  $resto\_lp_0$.**a** = $resto\_lp_1$.**a**
resto_lp $\longrightarrow$ ϵ
  $resto\_lp_0$.**a** = $resto\_lp_1$.**ah**
parametro $\longrightarrow$ *tipo resto_parametro*
  resto_parametro.**ah** = *tipo*.**a**
  parametro.**a** = *resto_parametro*.**a**
resto_parametro $\longrightarrow$ & **identificador**
  resto_parametro.**a** = *si_refparam_f*(*resto_parametro*.**ah**, **identificador.lex**)
resto_parametro $\longrightarrow$ **identificador**
  resto_parametro.**a** = *no_refparam_f*(*resto_parametro*.**ah**, **identificador.lex**)

3.1.2. Tipos

tipo_nombre $\longrightarrow$ *tipo* **identificador**
  tipo_nombre.**a** = *tipo_nombre*(*tipo*.**a**, **identificador.lex**)
tipo $\longrightarrow$ *tipo*0
  tipo.**a** = *tipo*0.**a**
*tipo*0 $\longrightarrow$ *tipo*1 *resto_tipo*0
  *resto_tipo*0.**ah** = *tipo*1.**a**
  *tipo*0.**a** = *resto_tipo*0.**a**
*resto_tipo*0 $\longrightarrow$ [**literalEntero**] *resto_tipo*0
  $resto\_tipo0_1$.**ah** = *tipo_array*($tipo0_0$.**ah**, **literalEntero.lex**)
  $tipo0_0$.**a** = $tipo0_1$.**a**
*resto_tipo*0 $\longrightarrow$ ϵ
  *tipo*0.**a** = *tipo*0.**ah**
*tipo*1 $\longrightarrow$ ^*tipo*1
  $tipo1_0$.**a** = *tipo_indir*($tipo1_1$.**a**)
*tipo*1 $\longrightarrow$ *tipo_base*
  *tipo*1.**a** = *tipo_base*.**a**
tipo_base $\longrightarrow$ **struct** {*lista_campos*}
  tipo_base.**a** = *tipo_struct*(*lista_campos*.**a**)
tipo_base $\longrightarrow$ **int**
  tipo_base.**a** = *tipo_int*()
tipo_base $\longrightarrow$ **real**
  tipo_base.**a** = *tipo_real*()
tipo_base $\longrightarrow$ **bool**
  tipo_base.**a** = *tipo_bool*()
tipo_base $\longrightarrow$ **string**
  tipo_base.**a** = *tipo_string*()
tipo_base $\longrightarrow$ **identificador**
  tipo_base.**a** = *tipo_type*()
lista_campos $\longrightarrow$ *tipo_nombre resto_lc*
  resto_lc.**ah** = *un_campo*(*tipo_nombre*.**a**)
  lista_campos.**a** = *resto_lc*.**a**
resto_lc $\longrightarrow$, *tipo_nombre resto_lc*
  $resto\_lc_1$.**ah** = *muchos_campos*($resto\_lc_0$.**ah**, *tipo_nombre*.**a**)
  $resto\_lc_0$.**a** = $resto\_lc_0$.**a**
resto_lc $\longrightarrow$ ϵ
  resto_lc.**a** = *resto_lc*.**ah**

3.1.3. Instrucciones

seccion_instrucciones_opt $\longrightarrow$ *seccion_instrucciones*
  seccion_instrucciones_opt.**a** = *si_ins*(*seccion_instrucciones*)
seccion_instrucciones_opt $\longrightarrow$ ϵ
  sseccion_instrucciones_opt.**a** = *no_ins*()
seccion_instrucciones $\longrightarrow$ *lista_instrucciones*
  seccion_instrucciones.**a** = *lista_instrucciones*.**a**
lista_instrucciones $\longrightarrow$ *instruccion resto_li*
  resto_li.**ah** = *una_ins*(*instruccion*.**a**)
  lista_instrucciones.**a** = *resto_li*.**a**
resto_li $\longrightarrow$; *instruccion resto_li*
  $resto\_li_1$.**ah** = *muchas_ins*($resto\_li_0$.**ah**, *instruccion*.**a**)
  $resto\_li_0$.**a** = $resto\_li_1$.**a**
resto_li $\longrightarrow$ ϵ
  resto_li.**a** = *resto_li*.**ah**
instruccion $\longrightarrow$ @ *expresion*
  instruccion.**a** = *ins_eval*(*expresion*.**a**)
instruccion $\longrightarrow$ *if_ins resto_ii*
  resto_ii.**ah** = *ins_if*(*expresion*.**a**, *bloque*.**a**)
  instruccion.**a** = *resto_ii*.**a**

$resto\_ii \longrightarrow$ **else** $bloque$
$\quad resto\_ii.\mathbf{a} = ins\_if\_else(resto\_ii.\mathbf{ah},\ bloque_1.\mathbf{a})$
$resto\_ii \longrightarrow \epsilon$
$\quad resto\_ii.\mathbf{a} = resto\_ii.\mathbf{ah}$
$instruccion \longrightarrow$ **while** $expresion\ bloque$
$\quad instruccion.\mathbf{a} = ins\_while(expresion.\mathbf{a},\ bloque.\mathbf{a})$
$instruccion \longrightarrow$ **read** $expresion$
$\quad instruccion.\mathbf{a} = ins\_read(expresion.\mathbf{a})$
$instruccion \longrightarrow$ **write** $expresion$
$\quad instruccion.\mathbf{a} = ins\_write(expresion.\mathbf{a})$
$instruccion \longrightarrow$ **nl**
$\quad instruccion.\mathbf{a} = ins\_nl()$
$instruccion \longrightarrow$ **new** $expresion$
$\quad instruccion.\mathbf{a} = ins\_new(expresion.\mathbf{a})$
$instruccion \longrightarrow$ **delete** $expresion$
$\quad instruccion.\mathbf{a} = ins\_delete(expresion.\mathbf{a})$
$instruccion \longrightarrow$ **call identificador** $parametros\_reales$
$\quad instruccion.\mathbf{a} = ins\_call(\mathbf{identificador.lex},\ parametros\_reales.\mathbf{a})$
$instruccion \longrightarrow bloque$
$\quad instruccion.\mathbf{a} = ins\_bloque(bloque.\mathbf{a})$
$if\_ins \longrightarrow$ **if** $expresion\ bloque$
$\quad if\_ins.\mathbf{a} = ins\_if(expresion.\mathbf{a},\ bloque.\mathbf{a})$
$parametros\_reales \longrightarrow (lista\_expresiones\_opt)$
$\quad parametros\_reales.\mathbf{a} = lista\_expresiones\_opt.\mathbf{a}$
$lista\_expresiones\_opt \longrightarrow lista\_expresiones$
$\quad lista\_expresiones\_opt.\mathbf{a} = si\_params\_r(lista\_expresiones.\mathbf{a})$
$lista\_expresiones\_opt \longrightarrow \epsilon$
$\quad lista\_expresiones\_opt.\mathbf{a} = no\_params\_r()$
$lista\_expresiones \longrightarrow expresion\ resto\_le$
$\quad resto\_le.\mathbf{ah} = un\_param\_r(expresion.\mathbf{a})$
$\quad lista\_expresiones.\mathbf{a} = resto\_le.\mathbf{a}$
$resto\_le \longrightarrow$, $expresion\ resto\_le$
$\quad resto\_le_1.\mathbf{ah} = muchos\_params\_r(resto\_le_0.\mathbf{ah},\ expresion.\mathbf{a})$
$\quad resto\_le_0.\mathbf{a} = resto\_le_1.\mathbf{a}$
$resto\_le \longrightarrow \epsilon$
$\quad resto\_le.\mathbf{a} = resto\_le.\mathbf{ah}$

3.1.4. Expresiones

$expresion \longrightarrow E0$
$\quad expresion.\mathbf{a} = E0.\mathbf{a}$
$E0 \longrightarrow E1\ resto\_E0$
$\quad resto\_E0.\mathbf{ah} = E1.\mathbf{a}$
$\quad E0.\mathbf{a} = resto\_E0.\mathbf{a}$
$resto\_E0 \longrightarrow =\ E0$
$\quad resto\_E0.\mathbf{a} = mkopbin(" = ",\ resto\_E0.\mathbf{ah},\ E0.\mathbf{a})$
$resto\_E0 \longrightarrow \epsilon$
$\quad resto\_E0.\mathbf{a} = resto\_E0.\mathbf{ah}$
$E1 \longrightarrow E2\ resto\_E1$
$\quad resto\_E1.\mathbf{ah} = E2.\mathbf{a}$
$\quad E1.\mathbf{a} = resto\_E1.\mathbf{a}$
$resto\_E1 \longrightarrow op\_relacional\ E2\ resto\_E1$
$\quad resto\_E1_1.\mathbf{ah} = mkopbin(op\_relacional.\mathbf{op},\ resto\_E1_0.\mathbf{ah},\ E2.\mathbf{a})$
$\quad resto\_E1_0.\mathbf{a} = resto\_E1_1.\mathbf{a}$
$resto\_E1 \longrightarrow \epsilon$
$\quad resto\_E1.\mathbf{a} = resto\_E1.\mathbf{ah}$
$E2 \longrightarrow E3\ resto\_E2\_F\ resto\_E2\_R$
$\quad resto\_E2\_F.\mathbf{ah} = E3.\mathbf{a}$
$\quad resto\_E2\_R.\mathbf{ah} = resto\_E2\_F.\mathbf{a}$
$\quad E2.\mathbf{a} = resto\_E2\_R.\mathbf{a}$

$resto\_E2\_R \longrightarrow\ +\ E3\ resto\_E2\_R$
$\quad resto\_E2\_R_1.\mathbf{ah} = mkopbin(" + ",\ resto\_E2\_R_0.\mathbf{ah},\ E3.\mathbf{a})$
$\quad resto\_E2\_R_0.\mathbf{a} = resto\_E2\_R_1.\mathbf{a}$
$resto\_E2\_R \longrightarrow\ \epsilon$
$\quad resto\_E2\_R.\mathbf{a} = resto\_E2\_R.\mathbf{ah}$
$resto\_E2\_F \longrightarrow\ -\ E3$
$\quad resto\_E2\_F.\mathbf{a} = mkopbin(" - ",\ resto\_E2\_F.\mathbf{ah},\ E3.\mathbf{a})$
$resto\_E2\_F \longrightarrow\ \epsilon$
$\quad resto\_E2\_F.\mathbf{a} = resto\_E2\_F.\mathbf{ah}$
$E3 \longrightarrow\ E4\ resto\_E3$
$\quad resto\_E3.\mathbf{ah} = E4.\mathbf{a}$
$\quad E3.\mathbf{a} = resto\_E3.\mathbf{a}$
$resto\_E3 \longrightarrow\ \mathbf{and}\ E3$
$\quad resto\_E3.\mathbf{a} = mkopbin("and", resto\_E3.\mathbf{ah},\ E3.\mathbf{a})$
$resto\_E3 \longrightarrow\ \mathbf{or}\ E4$
$\quad resto\_E3.\mathbf{a} = mkopbin("or", resto\_E3.\mathbf{ah},\ E4.\mathbf{a})$
$resto\_E3 \longrightarrow\ \epsilon$
$\quad resto\_E3.\mathbf{a} = resto\_E3.\mathbf{ah}$
$E4 \longrightarrow\ E5\ resto\_E4$
$\quad resto\_E4.\mathbf{ah} = E5.\mathbf{a}$
$\quad E4.\mathbf{a} = resto\_E4.\mathbf{a}$
$resto\_E4 \longrightarrow\ op\_mult\ E5\ resto\_E4$
$\quad resto\_E4_1.\mathbf{ah} = mkopbin(op\_mult.\mathbf{op},\ resto\_E4_0.\mathbf{ah},\ E5.\mathbf{a})$
$\quad resto\_E4_0.\mathbf{a} = resto\_E4_1.\mathbf{a}$
$resto\_E4 \longrightarrow \epsilon$
$\quad resto\_E4.\mathbf{a} = resto\_E4.\mathbf{ah}$
$E5 \longrightarrow\ -\ E5$
$\quad E5_0.\mathbf{a} = mkopun(" - ", E5_1.\mathbf{a})$
$E5 \longrightarrow\ \mathbf{not}\ E5$
$\quad E5_0.\mathbf{a} = mkopun("not", E5_1.\mathbf{a})$
$E5 \longrightarrow\ E6$
$\quad E5.\mathbf{a} = E6.\mathbf{a}$
$E6 \longrightarrow\ E7\ resto\_E6$
$\quad resto\_E6.\mathbf{ah} = E7.\mathbf{a}$
$\quad E6.\mathbf{a} = resto\_E6.\mathbf{a}$
$resto\_E6 \longrightarrow\ op\_dirs\ resto\_E6$
$\quad resto\_E6_1.\mathbf{ah} = mkopun(op\_dirs.\mathbf{op},\ resto\_E6_0.\mathbf{ah})$
$\quad resto\_E6_0.\mathbf{a} = resto\_E6_1.\mathbf{a}$
$resto\_E6 \longrightarrow\ \epsilon$
$\quad resto\_E6.\mathbf{a} = resto\_E6.\mathbf{ah}$
$E7 \longrightarrow\ expresion\_basica$
$\quad E7.\mathbf{a} = expresion\_basica.\mathbf{a}$
$E7 \longrightarrow\ (E0)$
$\quad E7.\mathbf{a} = E0.\mathbf{a}$
$expresion\_basica \longrightarrow\ \mathbf{literalEntero}$
$\quad expresion\_basica.\mathbf{a} = exp\_entero(\mathbf{literalEntero.lex})$
$expresion\_basica \longrightarrow\ \mathbf{literalReal}$
$\quad expresion\_basica.\mathbf{a} = exp\_real(\mathbf{literalReal.lex})$
$expresion\_basica \longrightarrow\ \mathbf{true}$
$\quad expresion\_basica.\mathbf{a} = exp\_true()$
$expresion\_basica \longrightarrow\ \mathbf{false}$
$\quad expresion\_basica.\mathbf{a} = exp\_false()$
$expresion\_basica \longrightarrow\ \mathbf{literalCadena}$
$\quad expresion\_basica.\mathbf{a} = exp\_cadena(\mathbf{literalCadena.lex})$
$expresion\_basica \longrightarrow\ \mathbf{identificador}$
$\quad expresion\_basica.\mathbf{a} = exp\_iden(\mathbf{identificador.lex})$
$expresion\_basica \longrightarrow\ \mathbf{null}$
$\quad expresion\_basica.\mathbf{a} = exp\_null()$

3.1.5. Operadores

$op\_relacional \longrightarrow <$
$\quad op\_relacional.\mathbf{op} = \text{" < "}$
$op\_relacional \longrightarrow <=$
$\quad op\_relacional.\mathbf{op} = \text{" <= "}$
$op\_relacional \longrightarrow >$
$\quad op\_relacional.\mathbf{op} = \text{" > "}$
$op\_relacional \longrightarrow >=$
$\quad op\_relacional.\mathbf{op} = \text{" >= "}$
$op\_relacional \longrightarrow ==$
$\quad op\_relacional.\mathbf{op} = \text{" == "}$
$op\_relacional \longrightarrow !=$
$\quad op\_relacional.\mathbf{op} = \text{"!= "}$
$op\_mult \longrightarrow *$
$\quad op\_mult.\mathbf{op} = \text{" * "}$
$op\_mult \longrightarrow /$
$\quad op\_mult.\mathbf{op} = \text{"/"}$
$op\_mult \longrightarrow \%$
$\quad op\_mult.\mathbf{op} = \text{" \%"}$
$op\_dirs \longrightarrow [expresion]$
$\quad op\_dirs.\mathbf{op} = \text{"}index\text{"}$
$op\_dirs \longrightarrow .\ \mathbf{identificador}$
$\quad op\_dirs.\mathbf{op} = \text{"}reg\text{"}$
$op\_dirs \longrightarrow \hat{}$
$\quad op\_dirs.\mathbf{op} = \text{"\^{}"}$

3.2. Funciones semánticas auxiliares

3.2.1. Expresiones unarias

fun $mkopun(op,\ opnd)$:
$\quad op = \text{" - "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_menos}(opnd)$
$\quad op = \text{"}not\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_not}(opnd)$
$\quad op = \text{"}index\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_index}(opnd)$
$\quad op = \text{"}reg\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_reg}(opnd)$
$\quad op = \text{"\^{}"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_indir}(opnd)$

3.2.2. Expresiones binarias

fun $mkopbin(op,\ opnd1,\ opnd2)$:
$\quad op = \text{" = "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_asig}(opnd1,\ opnd2)$
$\quad op = \text{" < "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_menor}(opnd1,\ opnd2)$
$\quad op = \text{" <= "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_menor\_ig}(opnd1,\ opnd2)$
$\quad op = \text{" > "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mayor}(opnd1,\ opnd2)$
$\quad op = \text{" >= "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mayor\_ig}(opnd1,\ opnd2)$
$\quad op = \text{" == "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_ig}(opnd1,\ opnd2)$
$\quad op = \text{"!= "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_dist}(opnd1,\ opnd2)$
$\quad op = \text{" + "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_suma}(opnd1,\ opnd2)$
$\quad op = \text{" - "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_resta}(opnd1,\ opnd2)$
$\quad op = \text{"}and\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_and}(opnd1,\ opnd2)$
$\quad op = \text{"}or\text{"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_or}(opnd1,\ opnd2)$
$\quad op = \text{" * "} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mul}(opnd1,\ opnd2)$
$\quad op = \text{"/"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_div}(opnd1,\ opnd2)$
$\quad op = \text{" \%"} \longrightarrow \mathbf{return}\ \ \mathbf{exp\_mod}(opnd1,\ opnd2)$

4 Especificación del procesamiento 'impresión bonita'