Procesadores de Lenguajes

Memoria de proyecto - Hito 2: Analizador Sintáctico

Grupo 10

Sergio Colet García

Laura Martínez Tomás

Rodrigo Souto Santos

Li Jie Chen Chen

Grado en Ingeniería informática

Facultad de Informática

Universidad Complutense de Madrid

2023-2024

Índice general

1 | Especificación de la sintaxis abstracta

1.1. Eliminación de terminales sin carga semántica

programa $\longrightarrow$ *bloque*
bloque $\longrightarrow$ *declaraciones instrucciones*
declaraciones $\longrightarrow$ *lista-declaraciones*
declaraciones $\longrightarrow \epsilon$
lista-declaraciones $\longrightarrow$ *lista-declaraciones declaracion*
lista-declaraciones $\longrightarrow$ *declaracion*
lista-variables $\longrightarrow$ *lista-variables declaracion*
lista-variables $\longrightarrow$ *declaracion*
declaracion $\longrightarrow$ *tipo* **identificador**
declaracion $\longrightarrow$ **type** *tipo* **identificador**
declaracion $\longrightarrow$ **identificador** *par-formales bloque*
tipo $\longrightarrow$ *tipo* **literalEntero**
tipo $\longrightarrow$ *tipo*2
*tipo*2 $\longrightarrow$ ^*tipo*2
*tipo*2 $\longrightarrow$ *tipo*3
*tipo*3 $\longrightarrow$ **bool**
*tipo*3 $\longrightarrow$ **int**
*tipo*3 $\longrightarrow$ **real**
*tipo*3 $\longrightarrow$ **string**
*tipo*3 $\longrightarrow$ **identificador**
*tipo*3 $\longrightarrow$ **struct** *lista-variables*
instrucciones $\longrightarrow$ *lista-instrucciones*
instrucciones $\longrightarrow \epsilon$
lista-instrucciones $\longrightarrow$ *lista-instrucciones instruccion*
lista-instrucciones $\longrightarrow$ *instruccion*
par-formales $\longrightarrow$ *lista-par-formal*
par-formales $\longrightarrow \epsilon$
lista-par-formal $\longrightarrow$ *par-formal lista-par-formal*
lista-par-formal $\longrightarrow$ *par-formal*
par-formal $\longrightarrow$ *tipo* &**identificador**
par-formal $\longrightarrow$ *tipo* **identificador**
par-reales $\longrightarrow$ *lista-par-real*
par-reales $\longrightarrow \epsilon$
lista-par-real $\longrightarrow E$ *lista-par-real*
lista-par-real $\longrightarrow E$
instruccion $\longrightarrow$ *eval*
instruccion $\longrightarrow$ **if** E *bloque*
instruccion $\longrightarrow$ **if** E *bloque bloque*
instruccion $\longrightarrow$ **while** E *bloque*
instruccion $\longrightarrow$ **new** E
instruccion $\longrightarrow$ **delete** E
instruccion $\longrightarrow$ **read** E
instruccion $\longrightarrow$ **write** E
instruccion $\longrightarrow$ **call identificador** *par-reales*
instruccion $\longrightarrow$ **nl**
instruccion $\longrightarrow$ *bloque*
eval $\longrightarrow E$
$E \longrightarrow E1 = E$
$E \longrightarrow E1$

$E1 \longrightarrow E1\ \ OP1\ \ E2$
$E1 \longrightarrow E2$
$E2 \longrightarrow E2 + E3$
$E2 \longrightarrow E3 - E3$
$E2 \longrightarrow E3$
$E3 \longrightarrow E4\ \ and\ \ E3$
$E3 \longrightarrow E4\ \ or\ \ E4$
$E3 \longrightarrow E4$
$E4 \longrightarrow E4\ \ OP4\ \ E5$
$E4 \longrightarrow E5$
$E5 \longrightarrow OP5\ \ E6$
$E5 \longrightarrow E6$
$E6 \longrightarrow E6\ OP6$
$E6 \longrightarrow E7$
$E7 \longrightarrow$ **true**
$E7 \longrightarrow$ **false**
$E7 \longrightarrow$ **literalEntero**
$E7 \longrightarrow$ **literalReal**
$E7 \longrightarrow$ **literalCadena**
$E7 \longrightarrow$ **identificador**
$E7 \longrightarrow$ **null**
$E7 \longrightarrow E$
$OP1 \longrightarrow <$
$OP1 \longrightarrow <=$
$OP1 \longrightarrow >$
$OP1 \longrightarrow >=$
$OP1 \longrightarrow ==$
$OP1 \longrightarrow !=$
$OP4 \longrightarrow\ *$
$OP4 \longrightarrow\ /$
$OP4 \longrightarrow\ \%$
$OP5 \longrightarrow\ +$
$OP5 \longrightarrow\ -$
$OP5 \longrightarrow$ **not**
$OP6 \longrightarrow\ E$
$OP6 \longrightarrow$ **identificador**
$OP6 \longrightarrow\ \hat{}$

1.2. Simplificación de la sintaxis

bloque $\longrightarrow$ *declaraciones instrucciones*
declaraciones $\longrightarrow$ *lista-declaraciones*
declaraciones $\longrightarrow \epsilon$
lista-declaraciones $\longrightarrow$ *lista-declaraciones declaracion*
lista-declaraciones $\longrightarrow$ *declaracion*
lista-variables $\longrightarrow$ *lista-variables declaracion*
lista-variables $\longrightarrow$ *declaracion*
declaracion $\longrightarrow$ *tipo* **identificador**
declaracion $\longrightarrow$ **type** *tipo* **identificador**
declaracion $\longrightarrow$ **identificador** *par-formales bloque*
tipo $\longrightarrow$ *tipo* **literalEntero**
tipo $\longrightarrow \hat{}$*tipo*
tipo $\longrightarrow$ **bool**
tipo $\longrightarrow$ **int**
tipo $\longrightarrow$ **real**
tipo $\longrightarrow$ **string**
tipo $\longrightarrow$ **identificador**
tipo $\longrightarrow$ **struct** *lista-variables*

instrucciones $\longrightarrow$ *lista-instrucciones*
instrucciones $\longrightarrow \epsilon$
lista-instrucciones $\longrightarrow$ *lista-instrucciones instruccion*
lista-instrucciones $\longrightarrow$ *instruccion*
par-formales $\longrightarrow$ *lista-par-formal*
par-formales $\longrightarrow \epsilon$
lista-par-formal $\longrightarrow$ *par-formal lista-par-formal*
lista-par-formal $\longrightarrow$ *par-formal*
par-formal $\longrightarrow$ *tipo* &**identificador**
par-formal $\longrightarrow$ *tipo* **identificador**
par-reales $\longrightarrow$ *lista-par-real*
par-reales $\longrightarrow \epsilon$
lista-par-real $\longrightarrow E$ *lista-par-real*
lista-par-real $\longrightarrow E$
instruccion $\longrightarrow E$
instruccion $\longrightarrow$ **if** E *bloque*
instruccion $\longrightarrow$ **if** E *bloque* **else** *bloque*
instruccion $\longrightarrow$ **while** E *bloque*
instruccion $\longrightarrow$ **new** E
instruccion $\longrightarrow$ **delete** E
instruccion $\longrightarrow$ **read** E
instruccion $\longrightarrow$ **write** E
instruccion $\longrightarrow$ **call identificador** *par-reales*
instruccion $\longrightarrow$ **nl**
instruccion $\longrightarrow$ *bloque*
$E \longrightarrow E = E$
$E \longrightarrow E < E$
$E \longrightarrow E <= E$
$E \longrightarrow E > E$
$E \longrightarrow E >= E$
$E \longrightarrow E == E$
$E \longrightarrow E \,!= E$
$E \longrightarrow E + E$
$E \longrightarrow E - E$
$E \longrightarrow E * E$
$E \longrightarrow E \,/\, E$
$E \longrightarrow E \,\%\, E$
$E \longrightarrow E$ *and* E
$E \longrightarrow E$ *or* E
$E \longrightarrow -E$
$E \longrightarrow$ **not** E
$E \longrightarrow E\ E$
$E \longrightarrow E$.**identificador**
$E \longrightarrow E\hat{}$
$E \longrightarrow$ **true**
$E \longrightarrow$ **false**
$E \longrightarrow$ **literalEntero**
$E \longrightarrow$ **literalReal**
$E \longrightarrow$ **literalCadena**
$E \longrightarrow$ **identificador**
$E \longrightarrow$ **null**

1.3. Asignación de géneros a cada no-terminal

Cuadro 1.3.1: Géneros de los no-terminales

No terminal	**Género**
bloque	Blo
declaraciones	Decs
lista-declaraciones	LDecs
lista-variables	LVar
declaracion	Dec
tipo	Tipo
instrucciones	Insts
lista-instrucciones	LInst
par-formales	PFmls
lista-par-formal	LPFml
par-formal	PFml
par-reales	PReales
lista-par-reas	LPReal
instruccion	Inst
E	Exp

1.4. Constructores de las reglas

Cuadro 1.4.1: Constructores de las diferentes reglas

Regla	**Constructor**
bloque $\longrightarrow$ *declaraciones instrucciones*	bloq: Decs $\times$ Insts $\rightarrow$ Blo
declaraciones $\longrightarrow$ *lista-declaraciones*	si_decs: LDecs $\rightarrow$ Decs
declaraciones $\longrightarrow \epsilon$	no_decs: $\rightarrow$ Decs
declaraciones $\longrightarrow$	LVar
lista-declaraciones $\longrightarrow$ *lista-declaraciones declaracion*	muchas_decs: LDecs $\times$ Dec $\rightarrow$ LDecs
lista-declaraciones $\longrightarrow$ *declaracion*	una_dec: Dec $\rightarrow$ LDecs
lista-variables $\longrightarrow$ *lista-variables declaracion*	muchas_var: LVar $\times$ Dec $\rightarrow$ LVar
lista-variables $\longrightarrow$ *declaracion*	una_var: Dec $\rightarrow$ LVar
declaracion $\longrightarrow$ *tipo* **identificador**	dec_simple: Tipo $\times$ **string** $\rightarrow$ Dec
declaracion $\longrightarrow$ **type** *tipo* **identificador**	dec_type: Tipo $\times$ **string** $\rightarrow$ Dec
declaracion $\longrightarrow$ **identificador** *par-formales bloque*	dec_proc: **string** $\times$ PFmls $\times$ Blo $\rightarrow$ Dec
tipo $\longrightarrow$ *tipo* **literalEntero**	tipo_array: Tipo $\times$ **string** $\rightarrow$ Tipo
tipo $\longrightarrow$ ^*tipo*	tipo_punt: Tipo $\rightarrow$ Tipo
tipo $\longrightarrow$ **bool**	tipo_bool: $\rightarrow$ Tipo
tipo $\longrightarrow$ **int**	tipo_int: $\rightarrow$ Tipo
tipo $\longrightarrow$ **real**	tipo_real: $\rightarrow$ Tipo
tipo $\longrightarrow$ **string**	tipo_string: $\rightarrow$ Tipo
tipo $\longrightarrow$ **identificador**	tipo_ident: **string** $\rightarrow$ Tipo

Continúa en la siguiente página

Cuadro 1.4.1: Constructores de las diferentes reglas (Continuación)

Regla	**Constructor**
tipo $\longrightarrow$ **struct** *lista-variables*	tipo_struct: LVar $\rightarrow$ Tipo
instrucciones $\longrightarrow$ *lista-instrucciones*	si_inst: LInst $\rightarrow$ Insts
instrucciones $\longrightarrow \epsilon$	no_inst: $\rightarrow$ Insts
lista-instrucciones $\longrightarrow$ *lista-instrucciones instruccion*	muchas_inst: LInst $\times$ Inst $\rightarrow$ LInst
lista-instrucciones $\longrightarrow$ *instruccion*	una_inst: Inst $\rightarrow$ LInst
par-formales $\longrightarrow$ *lista-par-formal*	si_pformal: LPFml $\rightarrow$ PFmls
par-formales $\longrightarrow \epsilon$	no_pformal: $\rightarrow$ PFmls
lista-par-formal $\longrightarrow$ *par-formal lista-par-formal*	muchos_pformal: PFml $\times$ LPFml $\rightarrow$ LPFml
lista-par-formal $\longrightarrow$ *par-formal*	un_pformal: PFml $\rightarrow$ LPFml
par-formal $\longrightarrow$ *tipo* &**identificador**	pformal_ref: Tipo $\times$ **string** $\rightarrow$ PFml
par-formal $\longrightarrow$ *tipo* **identificador**	pformal_noref: Tipo $\times$ **string** $\rightarrow$ PFml
par-reales $\longrightarrow$ *lista-par-real*	si_preales: LPReal $\rightarrow$ PReales
par-reales $\longrightarrow \epsilon$	no_preales: $\rightarrow$ PReales
lista-par-real $\longrightarrow$ *E lista-par-real*	muchas_exp: Exp $\times$ LPReal $\rightarrow$ LPReal
lista-par-real $\longrightarrow$ *E*	una_exp: Exp $\rightarrow$ LPReal
instruccion $\longrightarrow$ *E*	inst_eval: Exp $\rightarrow$ Inst
instruccion $\longrightarrow$ **if** *E bloque*	inst_if: Exp $\times$ Blo $\rightarrow$ Inst
instruccion $\longrightarrow$ **if** *E bloque* **else** *bloque*	inst_else: Exp $\times$ Blo $\times$ Blo $\rightarrow$ Inst
instruccion $\longrightarrow$ **while** *E bloque*	inst_while: Exp $\times$ Blo $\rightarrow$ Inst
instruccion $\longrightarrow$ **new** *E*	inst_new: Exp $\rightarrow$ Inst
instruccion $\longrightarrow$ **delete** *E*	inst_delete: Exp $\rightarrow$ Inst
instruccion $\longrightarrow$ **read** *E*	inst_read: Exp $\rightarrow$ Inst
instruccion $\longrightarrow$ **write** *E*	inst_write: Exp $\rightarrow$ Inst
instruccion $\longrightarrow$ **call identificador** *par-reales*	inst_call: **string** $\times$ PReales $\rightarrow$ Inst
instruccion $\longrightarrow$ **nl**	inst_nl: $\rightarrow$ Inst
instruccion $\longrightarrow$ *bloque*	inst_blo: Blo $\rightarrow$ Inst
$E \longrightarrow E = E$	exp_asig: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E < E$	exp_menor: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E <= E$	exp_menIgual: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E > E$	exp_mayor: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E >= E$	exp_mayIgual: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E == E$	exp_igual: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E != E$	exp_dist: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E + E$	exp_sum: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E - E$	exp_resta: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E * E$	exp_mult: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E / E$	exp_div: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E \% E$	exp_mod: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E$ *and* E	exp_and: Exp $\times$ Exp $\rightarrow$ Exp
$E \longrightarrow E$ *or* E	exp_or: Exp $\times$ Exp $\rightarrow$ Exp

Continúa en la siguiente página

Cuadro 1.4.1: Constructores de las diferentes reglas (Continuación)

Regla	**Constructor**
$E \longrightarrow -E$	exp_menos: Exp → Exp
$E \longrightarrow$ **not** E	exp_not: Exp → Exp
$E \longrightarrow E\ E$	exp_index: Exp × Exp → Exp
$E \longrightarrow E$ **.identificador**	exp_reg: Exp × **string** → Exp
$E \longrightarrow E\hat{}$	exp_ind: Exp → Exp
$E \longrightarrow$ **true**	exp_true: → Exp
$E \longrightarrow$ **false**	exp_false: → Exp
$E \longrightarrow$ **literalEntero**	exp_litEnt: **string** → Exp
$E \longrightarrow$ **literalReal**	exp_litReal: **string** → Exp
$E \longrightarrow$ **literalCadena**	exp_litCad: **string** → Exp
$E \longrightarrow$ **identificador**	exp_iden: **string** → Exp
$E \longrightarrow$ **null**	exp_null: → Exp

2 | Especificación del constructor ATS's mediante una gramatica s-atribuida

.

2.1. Especificación del Constructor de ASTs

programa $\longrightarrow$ *bloque*
| *programa*.**a** = *broque*.**a**
bloque $\longrightarrow$ { *declaraciones instrucciones* }
| *bloque*.**a** = **bloq**(*declaraciones*.**a**, *instrucciones*.**a**)
declaraciones $\longrightarrow$ *lista-declaraciones* &&
| *declaraciones*.**a** = **si_decs**(*lista-declaraciones*.**a**)
declaraciones $\longrightarrow \epsilon$
| *declaraciones*.**a** = **no_decs**()
lista-declaraciones $\longrightarrow$ *lista-declaraciones* ; *declaracion*
| *lista-declaraciones*$_0$.**a** = **muchas_decs**(*lista-declaraciones*$_1$.**a**, *declaraciones*.**a**)
lista-declaraciones $\longrightarrow$ *declaracion*
| *lista-declaraciones*.**a** = **una_dec**(*declaraciones*.**a**)
lista-variables $\longrightarrow$ *lista-variables* , *declaracion*
| *lista-variables*$_0$.**a** = **muchas_var**(*lista-variables*$_1$.**a**, *declaraciones*.**a**)
lista-variables $\longrightarrow$ *declaracion*
| *lista-variables*.**a** = **una_var**(*declaraciones*.**a**)
declaracion $\longrightarrow$ *tipo* **identificador**
| *declaracion*.**a** = **dec_simple**(*tipo*.**a**, **identificador.lex**)
declaracion $\longrightarrow$ **type** *tipo* **identificador**
| *declaracion*.**a** = **dec_type**(*tipo*.**a**, **identificador.lex**)
declaracion $\longrightarrow$ **identificador** *par-formales bloque*
| *declaracion*.**a** = **dec_proc**(**identificador.lex**, *par-formales*.**a**, *bloque*.**a**)
tipo $\longrightarrow$ *tipo*[**literalEntero**]
| *tipo*.**a** = **tipo_array**(*tipo*.**a**, **literalEntero.lex**)
tipo $\longrightarrow$ *tipo2*
| *tipo*.**a** = *tipo2*.**a**
tipo2 $\longrightarrow$ ^*tipo2*
| *tipo2*.**a** = **tipo_punt**(*tipo2*.**a**)
tipo2 $\longrightarrow$ *tipo3*
| *tipo2*.**a** = *tipo3*.**a**
tipo3 $\longrightarrow$ **bool**
| *tipo3*.**a** = **tipo_bool**
tipo3 $\longrightarrow$ **int**
| *tipo3*.**a** = **tipo_int**
tipo3 $\longrightarrow$ **real**
| *tipo3*.**a** = **tipo_real**
tipo3 $\longrightarrow$ **string**
| *tipo3*.**a** = **tipo_string**
tipo3 $\longrightarrow$ **identificador**
| *tipo3*.**a** = **tipo_ident**(**identificador.lex**)
tipo3 $\longrightarrow$ **struct** { *lista-variables* }
| *tipo3*.**a** = **tipo_struct**(*lista-variables*.**a**)
instrucciones $\longrightarrow$ *lista-instrucciones*
| *instrucciones*.**a** = **si_inst**(*lista-instrucciones*.**a**)
instrucciones $\longrightarrow \epsilon$
| *instrucciones*.**a** = **no_inst**()
lista-instrucciones $\longrightarrow$ *lista-instrucciones* ; *instruccion*

| $lista\text{-}instrucciones_0.\mathbf{a} = \mathbf{muchas\_inst}(lista\text{-}instrucciones_1.\mathbf{a}, instruccion.\mathbf{a})$
$lista\text{-}instrucciones \longrightarrow instruccion$
| $lista\text{-}instrucciones.\mathbf{a} = \mathbf{una\_inst}(instruccion.\mathbf{a})$
$par\text{-}formales \longrightarrow (lista\text{-}par\text{-}formal)$
| $par\text{-}formales.\mathbf{a} = \mathbf{si\_pformal}(lista\text{-}par\text{-}formal.\mathbf{a})$
$par\text{-}formales \longrightarrow ()$
| $par\text{-}formales.\mathbf{a} = \mathbf{no\_pformal}()$
$lista\text{-}par\text{-}formal \longrightarrow par\text{-}formal, lista\text{-}par\text{-}formal$
| $lista\text{-}par\text{-}formal_0.\mathbf{a} = \mathbf{muchos\_pformal}(par\text{-}formal.\mathbf{a}, lista\text{-}par\text{-}formal_1.\mathbf{a})$
$lista\text{-}par\text{-}formal \longrightarrow par\text{-}formal$
| $lista\text{-}par\text{-}formal.\mathbf{a} = \mathbf{un\_pformal}(par\text{-}formal.\mathbf{a})$
$par\text{-}formal \longrightarrow tipo\ \&\mathbf{identificador}$
| $par\text{-}formal.\mathbf{a} = \mathbf{pformal\_ref}(tipo.\mathbf{a}, \mathbf{indentificador.lex})$
$par\text{-}formal \longrightarrow tipo\ \mathbf{identificador}$
| $par\text{-}formal.\mathbf{a} = \mathbf{pformal\_noref}(tipo.\mathbf{a}, \mathbf{indentificador.lex})$
$par\text{-}reales \longrightarrow (lista\text{-}par\text{-}real)$
| $par\text{-}reales.\mathbf{a} = \mathbf{si\_preales}(lista\text{-}par\text{-}real.\mathbf{a})$
$par\text{-}reales \longrightarrow ()$
| $par\text{-}reales.\mathbf{a} = \mathbf{no\_preales}()$
$lista\text{-}par\text{-}real \longrightarrow E0, lista\text{-}par\text{-}real$
| $lista\text{-}par\text{-}real_0.\mathbf{a} = \mathbf{muchas\_exp}(E0.\mathbf{a}, lista\text{-}par\text{-}real_1.\mathbf{a})$
$lista\text{-}par\text{-}real \longrightarrow E0$
| $lista\text{-}par\text{-}real.\mathbf{a} = \mathbf{una\_exp}(E0.\mathbf{a})$
$instruccion \longrightarrow eval$
| $instruccion.\mathbf{a} = eval.\mathbf{a}$
$instruccion \longrightarrow E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_eval}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{if}\ E0\ \ bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_if}(E0.\mathbf{a}, bloque.\mathbf{a})$
$instruccion \longrightarrow \mathbf{if}\ E0\ \ bloque\ \mathbf{else}\ bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_else}(E0.\mathbf{a}, bloque.\mathbf{a}, bloque.\mathbf{a})$
$instruccion \longrightarrow \mathbf{while}\ E0\ bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_while}(E0.\mathbf{a}, bloque.\mathbf{a})$
$instruccion \longrightarrow \mathbf{new}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_new}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{delete}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_delete}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{read}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_read}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{write}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_write}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{call\ identificador}\ par\text{-}reales$
| $instruccion.\mathbf{a} = \mathbf{inst\_call}(\mathbf{identificador.a}, par\text{-}reales.\mathbf{a})$
$instruccion \longrightarrow \mathbf{nl}$
$instruccion \longrightarrow bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_blo}(bloque.\mathbf{a})$
$eval \longrightarrow @\ E0$
| $eval.\mathbf{a} = E0.\mathbf{a}$
$E0 \longrightarrow E1 = E0$
| $E0_0.\mathbf{a} = \mathbf{exp\_asig}(E1.\mathbf{a}, E0_1.\mathbf{a})$
$E0 \longrightarrow E1$
| $E.\mathbf{a} = E1.\mathbf{a}$
$E1 \longrightarrow E1\ \ OP1\ \ E2$
| $E1_0.\mathbf{a} = \mathbf{mkop}(OP1.\mathbf{op}, E1_1.\mathbf{a}, E2.\mathbf{a})$
$E1 \longrightarrow E2$
| $E1.\mathbf{a} = E2.\mathbf{a}$
$E2 \longrightarrow E2 + E3$
| $E2_0.\mathbf{a} = \mathbf{exp\_suma}(E2_1.\mathbf{a}, E3.\mathbf{a})$
$E2 \longrightarrow E3 - E3$
| $E2.\mathbf{a} = \mathbf{exp\_resta}(E3_0.\mathbf{a}, E3_1.\mathbf{a})$

$E2 \longrightarrow E3$
$\mid \quad E2.\mathbf{a} = E3.\mathbf{a}$
$E3 \longrightarrow E4 \ \mathit{and} \ E3$
$\mid \quad E3_0.\mathbf{a} = \mathbf{exp\_and}(E4.\mathbf{a}, E3_1.\mathbf{a})$
$E3 \longrightarrow E4 \ \mathit{or} \ E4$
$\mid \quad E3.\mathbf{a} = \mathbf{exp\_or}(E4_0.\mathbf{a}, E4_1.\mathbf{a})$
$E3 \longrightarrow E4$
$\mid \quad E3.\mathbf{a} = E4.\mathbf{a}$
$E4 \longrightarrow E4 \ OP4 \ E5$
$\mid \quad E4_0.\mathbf{a} = \mathbf{mkop}(OP4.\mathbf{op}, E4_1.\mathbf{a}, E5.\mathbf{a})$
$E4 \longrightarrow E5$
$\mid \quad E4.\mathbf{a} = E5.\mathbf{a}$
$E5 \longrightarrow OP5 \ E6$
$\mid \quad E5.\mathbf{a} = \mathbf{mkop}(OP5.\mathbf{op}, E6.\mathbf{a})$
$E5 \longrightarrow E6$
$\mid \quad E5.\mathbf{a} = E6.\mathbf{a}$
$E6 \longrightarrow E6 \ OP6$
$\mid \quad E6.\mathbf{a} = \mathbf{mkop}(E6.\mathbf{a}, OP6.\mathbf{op})$
$E6 \longrightarrow E7$
$\mid \quad E6.\mathbf{a} = E7.\mathbf{a}$
$E7 \longrightarrow \mathbf{true}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_true}(\mathbf{true.lex})$
$E7 \longrightarrow \mathbf{false}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_false}(\mathbf{false.lex})$
$E7 \longrightarrow \mathbf{literalEntero}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_litEnt}(\mathbf{literalEntero.lex})$
$E7 \longrightarrow \mathbf{literalReal}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_litReal}(\mathbf{literalReal.lex})$
$E7 \longrightarrow \mathbf{literalCadena}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_litCad}(\mathbf{literalCadena.lex})$
$E7 \longrightarrow \mathbf{identificador}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_iden}(\mathbf{identificador.lex})$
$E7 \longrightarrow \mathbf{null}$
$\mid \quad E7.\mathbf{a} = \mathbf{exp\_null}(\mathbf{null.lex})$
$E7 \longrightarrow (\ E0\ )$
$\mid \quad E7.\mathbf{a} = E.\mathbf{a}$
$OP1 \longrightarrow <$
$\mid \quad OP1.\mathbf{op} = \text{“ } < \text{ ”}$
$OP1 \longrightarrow <=$
$\mid \quad OP1.\mathbf{op} = \text{“ } <= \text{ ”}$
$OP1 \longrightarrow >$
$\mid \quad OP1.\mathbf{op} = \text{“ } > \text{ ”}$
$OP1 \longrightarrow >=$
$\mid \quad OP1.\mathbf{op} = \text{“ } >= \text{ ”}$
$OP1 \longrightarrow ==$
$\mid \quad OP1.\mathbf{op} = \text{“ } == \text{ ”}$
$OP1 \longrightarrow !=$
$\mid \quad OP1.\mathbf{op} = \text{“ } \mathbf{!=} \text{ ”}$
$OP4 \longrightarrow *$
$\mid \quad OP4.\mathbf{op} = \text{“ } \mathbf{*} \text{ ”}$
$OP4 \longrightarrow /$
$\mid \quad OP4.\mathbf{op} = \text{“ } / \text{ ”}$
$OP4 \longrightarrow \%$
$\mid \quad OP4.\mathbf{op} = \text{“ } \% \text{ ”}$
$OP5 \longrightarrow -$
$\mid \quad OP5.\mathbf{op} = \text{“ } \mathbf{-} \text{ ”}$
$OP5 \longrightarrow \mathbf{not}$
$\mid \quad OP5.\mathbf{op} = \text{“ } \mathbf{not} \text{ ”}$
$OP6 \longrightarrow [\ E0\ ]$
$\mid \quad OP6.\mathbf{op} = E0.\mathbf{a}$

$OP6 \longrightarrow$ **identificador**
| $OP6$.**op** = **identificador.lex**
$OP6 \longrightarrow$ ^
| $OP6$.**op** = "^"

fun **mkop**($op, opnd1, opnd2$) :
| op = " < " $\longrightarrow$ $return$ **exp_menor**($opnd1, opnd2$)
| op = " <= " $\longrightarrow$ $return$ **exp_menIgual**($opnd1, opnd2$)
| op = " > " $\longrightarrow$ $return$ **exp_mayor**($opnd1, opnd2$)
| op = " >= " $\longrightarrow$ $return$ **exp_mayIgual**($opnd1, opnd2$)
| op = " == " $\longrightarrow$ $return$ **exp_igual**($opnd1, opnd2$)
| op = " = " $\longrightarrow$ $return$ **exp_dist**($opnd1, opnd2$)
| op = " $*$ " $\longrightarrow$ $return$ **exp_mult**($opnd1, opnd2$)
| op = " / " $\longrightarrow$ $return$ **exp_div**($opnd1, opnd2$)
| op = " % " $\longrightarrow$ $return$ **exp_mod**($opnd1, opnd2$)

fun **mkop**($op, opnd1$) :
| op = " − " $\longrightarrow$ $return$ **exp_menos**($opnd1$)
| op = "**not**" $\longrightarrow$ $return$ **exp_not**($opnd1$)
| op = $E0.a \longrightarrow$ $return$ **exp_index**($opnd1$)
| op = **identificador.lex** $\longrightarrow$ $return$ **exp_reg**($opnd1$)
| op = "^" $\longrightarrow$ $return$ **exp_ind**($opnd1$)

3 | Acondicionamiento de la especificación del constructor ATS's

programa $\longrightarrow$ *bloque*
| *programa*.**a** = *broque*.**a**
bloque $\longrightarrow$ { *declaraciones instrucciones* }
| *bloque*.**a** = **bloq**(*declaraciones*.**a**, *instrucciones*.**a**)
declaraciones $\longrightarrow$ *lista-declaraciones* &&
| *declaraciones*.**a** = **si_decs**(*lista-declaraciones*.**a**)
declaraciones $\longrightarrow \epsilon$
| *declaraciones*.**a** = **no_decs**()
lista-declaraciones $\longrightarrow$ *declaracion rlista-decs*
| *rlista-decs*.**ah** = **una_dec**(*declaracion*.**a**)
| *lista-declaraciones*.**a** = *rlista-decs*.**a**
rlista-decs $\longrightarrow$; *declaracion rlista-decs*
| $rlista\text{-}decs_1$.**ah** = **muchas_decs**($rlista\text{-}decs_0$.**ah**, *declaraciones*.**a**)
| $rlista\text{-}decs_0$.**a** = $rlista\text{-}decs_1$.**a**
rlista-decs $\longrightarrow \epsilon$
| *rlista-decs*.**a** = *rlista-decs*.**ah**
lista-variables $\longrightarrow$ *declaracion rlista-var*
| *rlista-var*.**ah** = **una_var**(*declaracion*.**a**)
| *lista-variables*.**a** = *rlista-var*.**a**
rlista-var $\longrightarrow$, *declaracion rlista-var*
| $rlista\text{-}var_1$.**ah** = **muchas_var**($rlista\text{-}var_0$.**ah**, *declaraciones*.**a**)
| $rlista\text{-}var_0$.**a** = $rlista\text{-}var_1$.**a**
rlista-var $\longrightarrow \epsilon$
| *rlista-var*.**a** = *rlista-var*.**ah**
declaracion $\longrightarrow$ *tipo* **identificador**
| *declaracion*.**a** = **dec_simple**(*tipo*.**a**, **identificador.lex**)
declaracion $\longrightarrow$ **type** *tipo* **identificador**
| *declaracion*.**a** = **dec_type**(*tipo*.**a**, **identificador.lex**)
declaracion $\longrightarrow$ **identificador** *par-formales bloque*
| *declaracion*.**a** = **dec_proc**(**identificador.lex**, *par-formales*.**a**, *bloque*.**a**)
tipo $\longrightarrow$ *tipo*2 *rtipo*
| *rtipo*.**ah** = *tipo*2.**a**
| *tipo*.**a** = *rtipo*.**a**
rtipo $\longrightarrow$ [**literalEntero**] *rtipo*
| $rtipo_1$.**ah** = **tipo_array**($rtipo_0$.**ah**, **literalEntero.lex**)
| $rtipo_0$.**a** = $rtipo_1$.**a**
*tipo*2 $\longrightarrow$ ^*tipo*2
| *tipo*2.**a** = **tipo_punt**(*tipo*2.**a**)
*tipo*2 $\longrightarrow$ *tipo*3
| *tipo*2.**a** = *tipo*3.**a**
*tipo*3 $\longrightarrow$ **bool**
| *tipo*3.**a** = **tipo_bool**
*tipo*3 $\longrightarrow$ **int**
| *tipo*3.**a** = **tipo_int**
*tipo*3 $\longrightarrow$ **real**
| *tipo*3.**a** = **tipo_real**
*tipo*3 $\longrightarrow$ **string**
| *tipo*3.**a** = **tipo_string**
*tipo*3 $\longrightarrow$ **identificador**
| *tipo*3.**a** = **tipo_ident**(**identificador.lex**)
*tipo*3 $\longrightarrow$ **struct** { *lista-variables* }
| *tipo*3.**a** = **tipo_struct**(*lista-variables*.**a**)
instrucciones $\longrightarrow$ *lista-instrucciones*
| *instrucciones*.**a** = **si_inst**(*lista-instrucciones*.**a**)

$instrucciones \longrightarrow \epsilon$
| $instrucciones.\mathbf{a} = \mathbf{no\_inst}()$
$lista\text{-}instrucciones \longrightarrow instruccion\ rlista\text{-}inst$
| $rlista\text{-}inst.\mathbf{ah} = \mathbf{una\_inst}(instruccion.\mathbf{a})$
| $lista\text{-}instrucciones.\mathbf{a} = rlista\text{-}inst.\mathbf{a}$
$rlista\text{-}inst \longrightarrow ;\ instruccion\ rlista\text{-}inst$
| $rlista\text{-}inst_1.\mathbf{ah} = \mathbf{muchas\_inst}(rlista\text{-}inst_0.\mathbf{ah}, instrucciones.\mathbf{a})$
| $rlista\text{-}inst_0.\mathbf{a} = rlista\text{-}inst_1.\mathbf{a}$
$rlista\text{-}inst \longrightarrow \epsilon$
| $rlista\text{-}inst.\mathbf{a} = rlista\text{-}inst.\mathbf{ah}$
$par\text{-}formales \longrightarrow (lista\text{-}par\text{-}formal)$
| $par\text{-}formales.\mathbf{a} = \mathbf{si\_pformal}(lista\text{-}par\text{-}formal.\mathbf{a})$
$par\text{-}formales \longrightarrow ()$
| $par\text{-}formales.\mathbf{a} = \mathbf{no\_pformal}()$
$lista\text{-}par\text{-}formal \longrightarrow par\text{-}formal, lista\text{-}par\text{-}formal$
| $lista\text{-}par\text{-}formal_0.\mathbf{a} = \mathbf{muchos\_pformal}(par\text{-}formal.\mathbf{a}, lista\text{-}par\text{-}formal_1.\mathbf{a})$
$lista\text{-}par\text{-}formal \longrightarrow par\text{-}formal$
| $lista\text{-}par\text{-}formal.\mathbf{a} = \mathbf{un\_pformal}(par\text{-}formal.\mathbf{a})$
$par\text{-}formal \longrightarrow tipo\ \&\mathbf{identificador}$
| $par\text{-}formal.\mathbf{a} = \mathbf{pformal\_ref}(tipo.\mathbf{a}, \mathbf{indentificador.lex})$
$par\text{-}formal \longrightarrow tipo\ \mathbf{identificador}$
| $par\text{-}formal.\mathbf{a} = \mathbf{pformal\_noref}(tipo.\mathbf{a}, \mathbf{indentificador.lex})$
$par\text{-}reales \longrightarrow (lista\text{-}par\text{-}real)$
| $par\text{-}reales.\mathbf{a} = \mathbf{si\_preales}(lista\text{-}par\text{-}real.\mathbf{a})$
$par\text{-}reales \longrightarrow ()$
| $par\text{-}reales.\mathbf{a} = \mathbf{no\_preales}()$
$lista\text{-}par\text{-}real \longrightarrow E0, lista\text{-}par\text{-}real$
| $lista\text{-}par\text{-}real_0.\mathbf{a} = \mathbf{muchas\_exp}(E0.\mathbf{a}, lista\text{-}par\text{-}real_1.\mathbf{a})$
$lista\text{-}par\text{-}real \longrightarrow E0$
| $lista\text{-}par\text{-}real.\mathbf{a} = \mathbf{una\_exp}(E0.\mathbf{a})$
$instruccion \longrightarrow eval$
| $instruccion.\mathbf{a} = eval.\mathbf{a}$
$instruccion \longrightarrow E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_eval}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{if}\ E0\ \ bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_if}(E0.\mathbf{a}, bloque.\mathbf{a})$
$instruccion \longrightarrow \mathbf{if}\ E0\ \ bloque\ \mathbf{else}\ bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_else}(E0.\mathbf{a}, bloque.\mathbf{a}, bloque.\mathbf{a})$
$instruccion \longrightarrow \mathbf{while}\ E0\ bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_while}(E0.\mathbf{a}, bloque.\mathbf{a})$
$instruccion \longrightarrow \mathbf{new}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_new}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{delete}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_delete}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{read}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_read}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{write}\ E0$
| $instruccion.\mathbf{a} = \mathbf{inst\_write}(E0.\mathbf{a})$
$instruccion \longrightarrow \mathbf{call\ identificador}\ par\text{-}reales$
| $instruccion.\mathbf{a} = \mathbf{inst\_call}(\mathbf{identificador.a}, par\text{-}reales.\mathbf{a})$
$instruccion \longrightarrow \mathbf{nl}$
$instruccion \longrightarrow bloque$
| $instruccion.\mathbf{a} = \mathbf{inst\_blo}(bloque.\mathbf{a})$
$eval \longrightarrow @\ E0$
| $eval.\mathbf{a} = E0.\mathbf{a}$
$E0 \longrightarrow E1 = E0$
| $E0_0.\mathbf{a} = \mathbf{exp\_asig}(E1.\mathbf{a}, E0_1.\mathbf{a})$
$E0 \longrightarrow E1$
| $E.\mathbf{a} = E1.\mathbf{a}$
$E1 \longrightarrow E2\ rE1$

$|$ $\quad rE1.\mathbf{ah} = E2.a$
$|$ $\quad E1.\mathbf{a} = rE1.\mathbf{a}$
$rE1 \longrightarrow OP1\ E2\ rE1$
$|$ $\quad rE1_1.\mathbf{ah} = \mathbf{mkop}(OP1.\mathbf{op}, rE1_0.\mathbf{ah}, E2.\mathbf{a})$
$|$ $\quad rE1_0.\mathbf{a} = rE1_1.\mathbf{a}$
$rE1 \longrightarrow \epsilon$
$|$ $\quad rE1.\mathbf{a} = rE1.\mathbf{ah}$
$E2 \longrightarrow E3 - E3\ rE2$
$|$ $\quad rE1.\mathbf{ah} = \mathbf{exp\_resta}(E3_0.\mathbf{a}, E3_1.\mathbf{a})$
$|$ $\quad E2.\mathbf{a} = rE2.\mathbf{a}$
$E2 \longrightarrow E3\ rE2$
$|$ $\quad rE2.\mathbf{ah} = E3.\mathbf{a}$
$|$ $\quad E2.\mathbf{a} = rE2.\mathbf{a}$
$rE2 \longrightarrow + E3\ rE2$
$|$ $\quad rE2_1.\mathbf{ah} = \mathbf{exp\_suma}(rE2_0.\mathbf{ah}, E3.\mathbf{a})$
$|$ $\quad rE2_0.\mathbf{a} = rE2_1.\mathbf{a}$
$rE2 \longrightarrow \epsilon$
$|$ $\quad rE2.\mathbf{a} = rE2.\mathbf{ah}$
$E3 \longrightarrow E4\ \ and\ \ E3$
$|$ $\quad E3_0.\mathbf{a} = \mathbf{exp\_and}(E4.\mathbf{a}, E3_1.\mathbf{a})$
$E3 \longrightarrow E4\ \ or\ \ E4$
$|$ $\quad E3.\mathbf{a} = \mathbf{exp\_or}(E4_0.\mathbf{a}, E4_1.\mathbf{a})$
$E3 \longrightarrow E4$
$|$ $\quad E3.\mathbf{a} = E4.\mathbf{a}$
$E4 \longrightarrow E5\ rE4$
$|$ $\quad rE4.\mathbf{ah} = E5.a$
$|$ $\quad E4.\mathbf{a} = rE4.\mathbf{a}$
$rE4 \longrightarrow OP4\ E5\ rE4$
$|$ $\quad rE4_1.\mathbf{ah} = \mathbf{mkop}(OP4.\mathbf{op}, rE4_0.\mathbf{ah}, E5.\mathbf{a})$
$|$ $\quad rE4_0.\mathbf{a} = rE4_1.\mathbf{a}$
$rE4 \longrightarrow \epsilon$
$|$ $\quad rE4.\mathbf{a} = rE4.\mathbf{ah}$
$E5 \longrightarrow OP5\ \ E6$
$|$ $\quad E5.\mathbf{a} = \mathbf{mkop}(OP5.\mathbf{op}, E6.\mathbf{a})$
$E5 \longrightarrow E6$
$|$ $\quad E5.\mathbf{a} = E6.\mathbf{a}$
$E6 \longrightarrow E7\ rE6$
$|$ $\quad rE6.\mathbf{ah} = E7.a$
$|$ $\quad E6.\mathbf{a} = rE6.\mathbf{a}$
$rE6 \longrightarrow OP6\ rE6$
$|$ $\quad rE6_1.\mathbf{ah} = \mathbf{mkop}(rE6_0.\mathbf{ah}, OP6.\mathbf{op})$
$|$ $\quad rE6_0.\mathbf{a} = rE6_1.\mathbf{a}$
$rE6 \longrightarrow \epsilon$
$|$ $\quad rE6.\mathbf{a} = rE6.\mathbf{ah}$
$E7 \longrightarrow \mathbf{true}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_true}(\mathbf{true.lex})$
$E7 \longrightarrow \mathbf{false}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_false}(\mathbf{false.lex})$
$E7 \longrightarrow \mathbf{literalEntero}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_litEnt}(\mathbf{literalEntero.lex})$
$E7 \longrightarrow \mathbf{literalReal}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_litReal}(\mathbf{literalReal.lex})$
$E7 \longrightarrow \mathbf{literalCadena}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_litCad}(\mathbf{literalCadena.lex})$
$E7 \longrightarrow \mathbf{identificador}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_iden}(\mathbf{identificador.lex})$
$E7 \longrightarrow \mathbf{null}$
$|$ $\quad E7.\mathbf{a} = \mathbf{exp\_null}(\mathbf{null.lex})$
$E7 \longrightarrow (\ E0\ )$
$|$ $\quad E7.\mathbf{a} = E.\mathbf{a}$

$OP1 \longrightarrow <$
| $OP1$.**op** = “ < ”
$OP1 \longrightarrow <=$
| $OP1$.**op** = “ <= ”
$OP1 \longrightarrow >$
| $OP1$.**op** = “ > ”
$OP1 \longrightarrow >=$
| $OP1$.**op** = “ >= ”
$OP1 \longrightarrow ==$
| $OP1$.**op** = “ == ”
$OP1 \longrightarrow !=$
| $OP1$.**op** = “ **!=** ”
$OP4 \longrightarrow *$
| $OP4$.**op** = “ ***** ”
$OP4 \longrightarrow /$
| $OP4$.**op** = “ / ”
$OP4 \longrightarrow \%$
| $OP4$.**op** = “ **%** ”
$OP5 \longrightarrow -$
| $OP5$.**op** = “ **-** ”
$OP5 \longrightarrow$ **not**
| $OP5$.**op** = “ **not** ”
$OP6 \longrightarrow [\ E0\ ]$
| $OP6$.**op** = $E0$.**a**
$OP6 \longrightarrow$ **identificador**
| $OP6$.**op** = **identificador.lex**
$OP6 \longrightarrow \hat{}$
| $OP6$.**op** = “^”

4 | Especificación del procesamiento de impresión

```
imprime(prog(Bloque):
    imprime(Bloque)

imprime(Bloque(Decs,Instr)):
    print"{"
    nl
    imprime(Decs)
    nl
    imprime (Instr)
    nl
    print"}"
    nl

imprime(si_decs(LDecs):
    imprime(Ldecs)
imprime(no_decs(LDecs): noop
imprime(muchas_decs(Ldecs, Dec):
    imprime(ldecs)
    print ","
    imprime(Dec)
imprime(una_dec(Dec):
    imprime(Dec)

imprime(muchas_var(lvar, Dec):
    imprime(lvar)
    print ","
    imprime(Dec)
imprime(una_var(Dec)
```

```
imprime(si_inst(lInstr):
    imprime(linstr)
imprime(no_inst(): noop
imprime(muchas_inst(linstr, instr):
    imprime(linstr)
    print ","
    imprime(instr)
imprime(una_inst(intr):
    imprime(instr)

imprime(dec_simple(Tipo, Id):
    imprime(Tipo)
    print ","
    imprime(Id)


imprime(dec_proc(Id, Par-formal,Bloque):
    imprime(Id)
    print ","
    imprime(Par-formal)
```

```
    imprime(Bloque)

imprime(Tipo_array(Tipo, N):
    imprime(Tipo)
    print ","
    imprime(litEnt)

imprime(Tipo2):

imprime(Tipo_punt(Tipo2):
    imprime(Tipo2)

imprime(Tipo3):

imprime(Tipo_bool(bool)):
    print "<"
    print Bool
    print ">"
imprime(Tipo_int(int)):
    print "<"
    print int
    print ">"
imprime(Tipo_real(real)):
    print "<"
    print real
    print ">"
imprime(Tipo_string(string)):
    print "<"
    print string
    print ">"

imprime(Tipo_ident(Id)):
    print Id

imprime(Tipo_struct(lvar)):
    print "<"
    print struct
    print ">"
    print "{"
    imprime(lvar)
    print "}"

imprime(si_pformal(l-par-formal):
    imprime(l-par-formal)

imprime(no_pformal():noop

imprime(muchos_pformal(par-formal,l-par-formal):
    imprime(par-formal)
    print ","
    imprime(l-par-formal)

imprime(un_pformal(par-formal):
    imprime(par-formal)

imprime(pformal-ref(Tipo,Id):
    imprime(Tipo)
    nl
    imprime(id)
```

```
imprime(pformal_noref(Tipo, Id):
    imprime(Tipo)
    nl
    imprime(id)

imprime(si_preales(l-par-real):
    imprime(l-par-real)

imprime(no_preales()):noop

imprime(muchas_exp(Exp, l-par-real):
    imprime(Exp)
    nl
    print ","
    nl
    imprime(l-par-real)
    nl

imprime(una_exp(Exp):
    imprime(Exp)
    nl

imprime(Eval)

imprime(inst_eval(Exp)):
    imprime(Exp)

imprime(inst_if(Exp, Bloque):
    print "<"
    print if
    print ">"
    nl
    imprime(Exp)
    nl
    imprime(Bloque)
    nl

imprime(inst_else(Exp, Bloque, Bloque):
    print "<"
    print else
    print ">"
    nl
    imprime(Exp)
    nl
    imprime(Bloque)
    nl

imprime(inst_while(Exp, Bloque):
    print "<"
    print while
    print ">"
    nl
    imprime(Exp)
    nl
    imprime(Bloque)
    nl

imprime(inst_new(Exp)):
    print "<"
    print new
```

```
    print ">"
    nl
    imprime(Exp)
    nl


imprime(inst_delete(Exp)):
    print "<"
    print delete
    print ">"
    nl
    imprime(Exp)
    nl

imprime(inst_read(read)):
    print "<"
    print read
    print ">"
    nl
    imprime(Exp)
    nl

imprime(inst_write(write)):
    print "<"
    print write
    print ">"
     nl
    imprime(Exp)
    nl


imprime(inst_call(Id, par-real):
    print "<"
    print call
    print ">"
    nl
    imprime(Id)
    imprime(par-real)
    nl

imprime(inst_blo(Bloque):
    imprime(Bloque)
    nl

imprime(dec_proc(Id, par-formales, Bloque)):
    print "<"
    print proc
    print ">"
    imprime(Id)
    imprime(par-formales)
    imprime(Bloque)
    nl

imprime(dec_type(tipo, Id)):
    print "<"
    print type
    print ">"
    nl
    imprime(tipo)
    imprime(Id)
```

```
    nl

imprime(nl):
    print "<"
    print nl
    print ">"
    nl

imprime(exp_and(Exp4, Exp3)):
    print "<"
    print and
    print ">"
imprime(exp_or(or)):
    print "<"
    print or
    print ">"
imprime("not"(not)):
    print "<"
    print not
    print ">"
imprime(exp_true(true)):
    print "<"
    print true
    print ">"
imprime(exp_false(false)):
    print "<"
    print false
    print ">"
imprime(exp_null(null)):
    print "<"
    print null
    print ">"

imprime(lit_Ent(N)):
    print N
imprime(lit_Real(R)):
    print R
imprime(lit_Cad(C)):
    print C
imprime (EOF(eof)):
    print "<"
    print EOF
    print ">"

imprime(exp_suma(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,"+",Opnd1,2,3)
imprime(exp_resta(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,"-",Opnd1,3,3)
imprime(exp_and(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,"and",Opnd1,4,3)
imprime(exp_or(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,"or",Opnd1,4,4)
imprime(exp_mult(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,"*",Opnd1,4,5)
imprime(exp_div(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    /",Opnd1,4,5)
imprime(exp_mod(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    %",Opnd1,4,5)

imprime(exp_men(Opnd0,Opnd1)):
```

```
    imprimeExpBin(Opnd0,    <",Opnd1,1,2)
imprime(exp_menIgual(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    <=",Opnd1,1,2)
imprime(exp_mayor(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    >",Opnd1,1,2)
imprime(exp_mayIgual(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    >=",Opnd1,1,2)
imprime(exp_igual(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    ==",Opnd1,1,2)
imprime(exp_dist(Opnd0,Opnd1)):
    imprimeExpBin(Opnd0,    !=",Opnd1,1,2)

imprime(exp_menos(Opnd0)):
    imprimeExpBin(    -",Opnd0,6)
imprime(exp_not(Opnd0)):
    imprimeExpBin(  not  ",Opnd0,6)
imprime(exp_index(Opnd0)):
    imprimeExpBin(  Exp  ",Opnd0,6)
imprime(exp_reg(Opnd0)):
    imprimeExpBin(  Id  ",Opnd0,6)
imprime(exp_ind(Opnd0)):
    imprimeExpBin("^",Opnd0,6)

imprimeExpUnario(Opnd0,Op,np0):
    print " "++Op++" "
    imprimeOpnd(Opnd0,np0)

imprimeExpBin(Opnd0,Op,Opnd1,np0,np1):
    imprimeOpnd(Opnd0,np0)
    print " "++Op++" "
    imprimeOpnd(Opnd1,np1)

imprimeOpnd(Opnd,MinPrior):
    if prioridad(Opnd) < MinPrior
        print    (
    end if
    imprime(Opnd)
    if prioridad(Opnd) < MinPrior
        print    )
    end if


prioridad(exp_suma(_,_)): return 2
prioridad(exp_resta(_,_)): return 2
prioridad(exp_and(_,_)): return 3
prioridad(exp_or(_,_)): return 3
prioridad(exp_mul(_,_)): return 4
prioridad(exp_div(_,_)): return 4
prioridad(exp_mod(_,_)): return 4

prioridad(exp_men(_,_)): return 1
prioridad(exp_menorIgual(_,_)): return 1
prioridad(exp_mayor(_,_)): return 1
prioridad(exp_mayIgual(_,_)): return 1
prioridad(exp_igual(_,_)): return 1
prioridad(exp_dist(_,_)): return 1


prioridad(exp_menos(_)): return 5
prioridad(exp_not(_)): return 5
```

```
prioridad(exp_index(_)): return 6
prioridad(exp_reg(_)): return 6
prioridad(exp_ind(_)): return 6
```

Índice de figuras

Temporary page!

LaTeX was unable to guess the total number of pages correctly. As there was some unprocessed data that should have been added to the final page this extra page has been added to receive it.

If you rerun the document (without altering it) this surplus page will go away, because LaTeX now knows how many pages to expect for this document.