

Practica 4:

Construcción de árboles de sintaxis abstracta

Grupo 11:

Youssef El Faqir El Rhazoui

Enrique Ávila Rodríguez

1. Conjunto de funciones constructoras

Prog: LDs x LIs \rightarrow Prog

LD_simp: String x String \rightarrow D

LD_comp: String x String x LDs \rightarrow LDs

LI_simp: String x Exp \rightarrow I

LI_comp: String x Exp x LIs \rightarrow LIs

Mas: Exp x Exp \rightarrow Exp

Menos: Exp x Exp \rightarrow Exp

And: Exp x Exp \rightarrow Exp

Or: Exp x Exp \rightarrow Exp

Distinto: Exp x Exp \rightarrow Exp

Igual: Exp x Exp \rightarrow Exp

Menor_que: Exp x Exp \rightarrow Exp

Menor_igual_que: Exp x Exp \rightarrow Exp

Mayor_que: Exp x Exp \rightarrow Exp

Mayor_igual_que: Exp x Exp \rightarrow Exp

Mul: Exp x Exp \rightarrow Exp

Div: Exp x Exp \rightarrow Exp

Not: Exp \rightarrow Exp

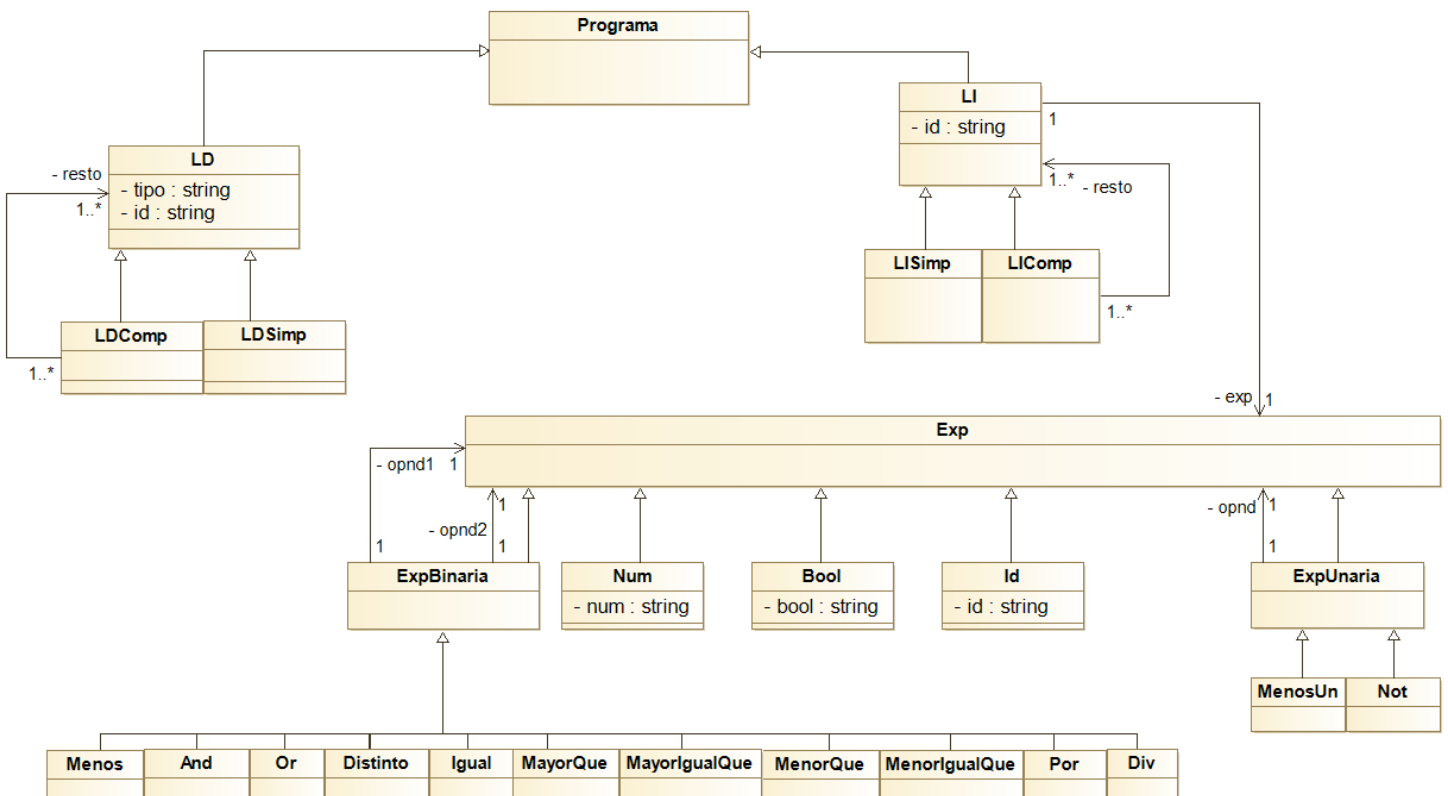
Menos_unario: Exp \rightarrow Exp

Num: String \rightarrow Exp

Bool: String \rightarrow Exp

Id: String \rightarrow Exp

2. Diagrama de clases



3. Gramática de atributos

Prog \rightarrow LDs && LIs

Prog.a = prog(LDs.a, LIs.a)

LDs \rightarrow LDs; D

LDs₀.a = ldCompuesta(D.tipo, D.id, LDs₁.a)

LDs \rightarrow D

LDs.a = ldSimple(D.tipo, D.id)

LIs \rightarrow LIs; I

LIs₀.a = liCompuesta(I.id, I.exp, LIs₁.a)

LIs \rightarrow I

LIs.a = liSimple(I.id, I.exp)

$D \rightarrow \text{tipo identificador}$

$D.\text{tipo} = \text{tipo.lex}$

$D.\text{id} = \text{identificador.lex}$

$I \rightarrow \text{identificador} = \text{Exp0}$

$I.\text{id} = \text{identificador.lex}$

$I.\text{exp} = \text{Exp0.a}$

$\text{Exp0} \rightarrow \text{Exp0 Op0 Exp1}$

$\text{Exp0}_0.\text{a} = \text{mkexp}(\text{Op0.op}, \text{Exp0}_1.\text{a}, \text{Exp1.a})$

$\text{Exp0} \rightarrow \text{Exp1}$

$\text{Exp0.a} = \text{Exp1.a}$

$\text{Exp1} \rightarrow \text{Exp2 and Exp1}$

$\text{Exp1}_0.\text{a} = \text{and}(\text{Exp2.a}, \text{Exp1}_1.\text{a})$

$\text{Exp1} \rightarrow \text{Exp2 or Exp2}$

$\text{Exp1.a} = \text{or}(\text{Exp2}_0.\text{a}, \text{Exp2}_1.\text{a})$

$\text{Exp1} \rightarrow \text{Exp2}$

$\text{Exp1.a} = \text{Exp2.a}$

$\text{Exp2} \rightarrow \text{Exp3 Op2 Exp3}$

$\text{Exp2.a} = \text{mkexp}(\text{Op2.op}, \text{Exp3}_0.\text{a}, \text{Exp3}_1.\text{a})$

$\text{Exp2} \rightarrow \text{Exp3}$

$\text{Exp2.a} = \text{Exp3.a}$

$\text{Exp3} \rightarrow \text{Exp3 Op3 Exp4}$

$\text{Exp3}_0.\text{a} = \text{mkexp}(\text{Op3.op}, \text{Exp3}_1.\text{a}, \text{Exp4.a})$

$\text{Exp3} \rightarrow \text{Exp4}$

$\text{Exp3.a} = \text{Exp4.a}$

$\text{Exp4} \rightarrow - \text{Exp4}$

$\text{Exp4}_0.\text{a} = \text{menos_unario}(\text{Exp4}_1.\text{a})$

$\text{Exp4} \rightarrow \text{not Exp5}$

$\text{Exp4.a} = \text{not}(\text{Exp5.a})$

$\text{Exp4} \rightarrow \text{Exp5}$

$\text{Exp4.a} = \text{Exp5.a}$

$\text{Exp5} \rightarrow \text{numero}$

Exp5.a = num(numero.lex)

Exp5 → booleano

Exp5.a = bool(booleano.lex)

Exp5 → identificador

Exp5.a = id(identificador.lex)

Exp5 → (Exp0)

Exp5.a = Exp0.a

Op0 → +

Op0.op = “+”

Op0 → -

Op0.op = “-”

Op2 → !=

Op2.op = “!=”

Op2 → ==

Op2.op = “==”

Op2 → <

Op2.op = “<”

Op2 → <=

Op2.op = “<=”

Op2 → >

Op2.op = “>”

Op2 → >=

Op2.op = “>=”

Op3 → *

Op3.op = “*”

Op3 → /

Op3.op = “/”

Definimos la función mkexp como sigue:

```
fun mkexp(op, opnd1,opnd2) {  
    switch(op) {  
        "+" => return suma(opnd1,opnd2)  
        "-" => return resta(opnd1,opnd2)  
        "!=" => return distinto(opnd1,opnd2)  
        "==" => return igual(opnd1,opnd2)  
        "<" => return menorQue(opnd1,opnd2)  
        "<=" => return menorIgualQue(opnd1,opnd2)  
        ">" => return mayorQue(opnd1,opnd2)  
        ">=" => return mayorIgualQue(opnd1,opnd2)  
        "*" => return mul(opnd1,opnd2)  
        "/" => return div(opnd1,opnd2)  
    }  
}
```

4. Acondicionamiento para imp descendente

Prog \rightarrow Sec_Dec && LIs

Prog.a = prog(LDs.a, LIs.a)

LDs \rightarrow D PDec

PDec.ah = ldSimple(D.tipo, D.id)

LDs.a = PDec.a

PDec \rightarrow ; D PDec

$PDec_1.a = ldCompuesta(PDec_0.ah, D.a)$

$PDec_0.a = PDec_1.a$

PDec $\rightarrow \varepsilon$

PDec.a = PDec.ah

LIs \rightarrow I PIns

PIns.ah = liSimple(I.id, I.exp)

LIs.a = PIns.a

$PIns \rightarrow ; I PIns$

$PIns_1.a = liCompuesta(PIns_0.ah, I.a)$

$PIns_0.a = PIns_1.a$

$PIns \rightarrow \varepsilon$

$PIns.a = PIns.ah$

$D \rightarrow \text{tipo identificador}$

$D.tipo = tipo.lex$

$D.id = identificador.lex$

$I \rightarrow \text{identificador} = Exp0$

$I.id = identificador.lex$

$I.exp = Exp0.a$

$Exp0 \rightarrow Exp1 RExp0$

$RExp0.ah = Exp1.v$

$Exp0.v = RExp0.v$

$RExp0 \rightarrow Op0 Exp1 RExp0$

$RExp0_1.ah = mkexp(Op0.op, RExp0_0.ah, Exp1.v)$

$RExp0_0.a = RExp0_1.a$

$RExp0 \rightarrow \varepsilon$

$RExp0.a = RExp0.ah$

$Exp1 \rightarrow Exp2 RExp1$

$RExp1.ah = Exp2.a$

$Exp1.a = RExp1.a$

$RExp1 \rightarrow \text{and } Exp1$

$RExp1.a = \text{and}(RExp.ah, Exp1.a)$

$RExp1 \rightarrow \text{or } Exp2$

$RExp1.a = \text{or}(RExp.ah, Exp2.a)$

$RExp1 \rightarrow \varepsilon$

$RExp1.a = RExp1.ah$

$Exp2 \rightarrow Exp3 RExp2$

$RExp2.ah = Exp3.a$

$Exp2.a = RExp2.a$

$\text{RExp2} \rightarrow \text{Op2 Exp3}$

$\text{RExp2.a} = \text{mkexp}(\text{Op2.op}, \text{RExp2.ah}, \text{Exp3.a})$

$\text{RExp2} \rightarrow \varepsilon$

$\text{RExp2.a} = \text{RExp2.ah}$

$\text{Exp3} \rightarrow \text{Exp4 RExp3}$

$\text{RExp3.ah} = \text{Exp4.a}$

$\text{Exp3.a} = \text{RExp3.a}$

$\text{RExp3} \rightarrow \text{Op3 Exp4 RExp3}$

$\text{RExp3}_1.\text{ah} = \text{mkexp}(\text{Op3.op}, \text{RExp3}_0.\text{ah}, \text{Exp4.a})$

$\text{RExp3}_0.\text{a} = \text{RExp3}_1.\text{a}$

$\text{RExp3} \rightarrow \varepsilon$

$\text{RExp3.a} = \text{RExp3.ah}$

$\text{Exp4} \rightarrow - \text{Exp4}$

$\text{Exp4}_0.\text{a} = \text{menos_unario}(\text{Exp4}_1.\text{a})$

$\text{Exp4} \rightarrow \text{not Exp5}$

$\text{Exp4.a} = \text{not}(\text{Exp5.a})$

$\text{Exp4} \rightarrow \text{Exp5}$

$\text{Exp4.a} = \text{Exp5.a}$

$\text{Exp5} \rightarrow \text{numero}$

$\text{Exp5.a} = \text{num}(\text{numero.lex})$

$\text{Exp5} \rightarrow \text{booleano}$

$\text{Exp5.a} = \text{bool}(\text{booleano.lex})$

$\text{Exp5} \rightarrow \text{identificador}$

$\text{Exp5.a} = \text{id}(\text{identificador.lex})$

$\text{Exp5} \rightarrow (\text{Exp0})$

$\text{Exp5.a} = \text{Exp0.a}$

$\text{Op0} \rightarrow +$

$\text{Op0.op} = "+"$

$\text{Op0} \rightarrow -$

$\text{Op0.op} = "-"$

$\text{Op2} \rightarrow !=$

Op2.op = “!=”

Op2 → ==

Op2.op = “==”

Op2 → <

Op2.op = “<”

Op2 → <=

Op2.op = “<=”

Op2 → >

Op2.op = “>”

Op2 → >=

Op2.op = “>=”

Op3 → *

Op3.op = “*”

Op3 → /

Op3.op = “/”