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 del procesamiento de vinculación **2**

2. Especificación del procesamiento de comprobación de tipos **7**

Índice de figuras **13**

1 | Especificación del procesamiento de vinculación

```
var ts //Tabla de simbolos

vincula(prog(Blo)):
    ts = creaTS()
    vincula(Blo)

vincula(bloq(Decs,Insts)):
    abreAmbito(ts)
    recolectaDecs(Decs)
    vincula(Insts)
    cierraAmbito(ts)

recolectaDecs(si_decs(LDecs)):
    recolectaDecs(Ldecs)

recolectaDecs(no_decs()): noop

recolectaDecs(muchas_decs(LDecs,Dec)):
    recolectaDecs(LDecs)
    recolectaDec(Dec)

recolectaDecs(una_dec(Dec)):
    recolectaDec(Dec)

recolectaVars(muchas_var(LVar,Var)):
    recolectaVars(LVar)
    recolectaVar(Var)

recolectaVars(una_var(Var)):
    recolectaVar(Var)

recolectaVar(var(Tipo,id)):
    vincula(Tipo)
    if contiene(ts,id) then
        error
    else
        inserta(ts,id,$)
    end if

recolectaDec(dec_simple(Var)):
    recolectaVar(Var)

recolectaDec(dec_type(Var)):
    recolectaVar(Var)

recolectaDec(dec_proc(id,PFmls,Blo)):
    if contiene(ts,id) then
        error
    else
        inserta(ts,id,$)
    end if
    abreAmbito(ts)
    recolectaPFmls(PFmls)
```

```
        vincula(Blo)
        cierraAmbito(ts)

    vincula(tipo_array(Tipo,litEnt)):
        vincula(Tipo)

    vincula(tipo_punt(Tipo)):
        vincula(Tipo)

    vincula(tipo_bool()): noop

    vincula(tipo_int()): noop

    vincula(tipo_real()): noop

    vincula(tipo_string()): noop

    vincula(tipo_ident(id)):
        $.vinculo = vinculoDe(ts,id)
        if $.vinculo == false then
            error
        end if

    vincula(tipo_struct(LVar)):
        recolectaVars(LVar)

    vincula(si_inst(LInst)):
        vincula(LInst):

    vincula(no_inst()): noop

    vincula(muchas_inst(LInst,Inst)):
        vincula(LInst)
        vincula(Inst)

    vincula(una_inst(Inst)):
        vincula(Inst)

    recolectaPFmls(si_pformal(LPFml)):
        recolectaPFmls(LPFml)

    recolectaPFmls(no_pformal()): noop

    recolectaPFmls(muchos_pformal(LPFml,PFml)):
        recolectaPFmls(LPFml)
        recolectaPFml(PFml)

    recolectaPFmls(un_pformal(PFml))
        recolectaPFml(PFml)

    recolectaPFml(pformal_ref(Tipo,id)):
        vincula(Tipo)
        if contiene(ts,id) then
            error
        else
            inserta(ts,id,$)
        end if

    recolectaPFml(pformal_noref(Tipo,id)):
        vincula(Tipo)
```

```
        if contiene(ts,id) then
            error
        else
            inserta(ts,id,$)
        end if

    vincula(si_preales(LPReal)):
        vincula(LPReal)

    vincula(no_preales()): noop

    vincula(muchas_exp(LPReal,Exp)):
        vincula(LPReal)
        vincula(Exp)

    vincula(una_exp(Exp)):
        vincula(Exp)

    vincula(inst_eval(Exp)):
        vincula(Exp)

    vincula(inst_if(Exp,Blo)):
        vincula(Exp)
        vincula(Blo)

    vincula(inst_else(Exp,Blo1,Blo2)):
        vincula(Exp)
        vincula(Blo1)
        vincula(Blo2)

    vincula(inst_while(Exp,Blo)):
        vincula(Exp)
        vincula(Blo)

    vincula(inst_new(Exp)):
        vincula(Exp)

    vincula(inst_delete(Exp)):
        vincula(Exp)

    vincula(inst_read(Exp)):
        vincula(Exp)

    vincula(inst_write(Exp)):
        vincula(Exp)

    vincula(inst_call(id,PReales)):
        $.vinculo = vinculoDe(ts,Id)
        if $.vinculo == false then
            error
        end if
        vincula(PReales)

    vincula(inst_nl()): noop

    vincula(inst_blo(Blo)):
        vincula(Blo)

    vincula(exp_asig(Opnd0,Opnd1)):
        vincula(Opnd0)
```

```
        vincula(Opnd1)

    vincula(exp_menor(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_menIgual(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_mayor(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_mayIgual(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_igual(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_dist(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_sum(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_resta(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_mult(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_div(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_mod(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_and(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_or(Opnd0,Opnd1)):
        vincula(Opnd0)
        vincula(Opnd1)

    vincula(exp_menos(Exp)):
        vincula(Exp)

    vincula(exp_not(Exp)):
        vincula(Exp)
```

```
vincula(inst_index(Opnd0,Opnd1)):
    vincula(Opnd0)
    vincula(Opnd1)

vincula(exp_reg(Exp,id)):
    vincula(Exp)
    $.vinculo = vinculoDe(ts,Id)
    if $.vinculo == false then
        error
    end if

vincula(exp_ind(Exp)):
    vincula(Exp)

vincula(exp_true()): noop

vincula(exp_false()): noop

vincula(exp_litEnt(litEnt)): noop

vincula(exp_litReal(litReal)): noop

vincula(exp_litCad(litCad)): noop

vincula(exp_iden(id)):
    $.vinculo = vinculoDe(ts,Id)
    if $.vinculo == false then
        error
    end if

vincula(exp_null()): noop
```

2 | Especificación del procesamiento de comprobación de tipos

```
tipado(prog(Blo)):
    tipado(Blo)
    $.tipo = Blo.tipo

tipado(bloq(Decs,Insts)):
    tipado(Decs)
    tipado(Insts)
    $.tipo = ambos-ok(Decs.tipo,Insts.tipo)

tipado(muchas_decs(LDecs,Dec)):
    tipado(LDecs)
    tipado(Dec)
    $.tipo = ambos-ok(LDecs.tipo,Dec.tipo)

tipado(una_dec(Dec)):
    tipado(Dec)
    $.tipo = Dec.tipo

tipado(muchas_var(LVar,Var)):
    tipado(LVar)
    tipado(Var)
    $.tipo = ambos-ok(LVar.tipo,Var.tipo)

tipado(una_var(Var)):
    tipado(Var)
    $.tipo = Var.tipo

tipado(var(Tipo,id)):
    tipado(Tipo)
    $.tipo = Tipo.tipo

tipo(dec_simple(Var)):
    tipado(Var)
    $.tipo = Var.tipo

tipo(dec_type(Var)):
    tipado(Var)
    $.tipo = Var.tipo

tipo(dec_proc(id,PFmls,Blo)):
    tipado(PFmls)
    tipado(Bloq)
    $.tipo = ambos-ok(PFmls.tipo,Blo.tipo)

tipado(tipo_array(Tipo,litEnt)):
    tipado(Tipo)
    $.tipo = Tipo.tipo

tipado(tipo_punt(Tipo)):
    tipado(Tipo)
    $.tipo = Tipo.tipo

tipado(tipo_bool()): $.tipo = bool
```

```
tipado(tipo_int()): $.tipo = int

tipado(tipo_real()): $.tipo = real

tipado(tipo_string()): $.tipo = string

tipado(tipo_ident(id)):
    let $.vinculo = Dec_var(T,I) in
        $.tipo = T
    end let

tipado(tipo_struct(LVar)):
    tipado(LVar)
    $.tipo = LVar.tipo

tipado(muchas_inst(LInst,Inst)):
    tipado(LInst)
    tipado(Inst)
    $.tipo = ambos-ok(LInst.tipo,Inst.tipo)

tipado(una_inst(Inst)):
    tipado(Inst)
    $.tipo = Inst.tipo

tipado(muchos_pformal(LPFml,PFml)):
    tipado(LPFml)
    tipado(PFml)
    $.tipo = ambos-ok(LPFml.tipo,PFml.tipo)

tipado(un_pformal(PFml))
    tipado(PFml)
    $.tipo = PFml.tipo

tipado(pformal_ref(Tipo,id)):
    tipado(Tipo)
    $.tipo = Tipo.tipo

recolectaPFml(pformal_noref(Tipo,id)):
    tipado(Tipo)
    $.tipo = Tipo.tipo

tipado(muchas_exp(LPReal,Exp)):
    tipado(LPReal)
    tipado(Exp)
    $.tipo = ambos-ok(LPReal.tipo,Exp.tipo)

tipado(una_exp(Exp)):
    tipado(Exp)
    $.tipo = Exp.tipo

tipado(inst_eval(Exp)):
    tipado(Exp)
    $.tipo = Exp.tipo

tipado(inst_if(Exp,Blo)):
    tipado(Exp)
    tipado(Blo)
    $.tipo = ambos-ok(Exp.tipo,Blo.tipo)

tipado(inst_else(Exp,Blo1,Blo2)):
```

```
        tipado(Exp)
        tipado(Blo1)
        tipado(Blo2)
        if Exp.tipo == ok ^ Blo1 == ok ^ Blo2 == ok then
            $.tipo = ok
        else
            return error
        end if

    tipado(inst_while(Exp,Blo)):
        tipado(Exp)
        tipado(Blo)
        $.tipo = ambos-ok(Exp.tipo,Blo.tipo)

    tipado(inst_new(Exp)):
        tipado(Exp)
        $.tipo = Exp.tipo

    tipado(inst_delete(Exp)):
        tipado(Exp)
        $.tipo = Exp.tipo

    tipado(inst_read(Exp)):
        tipado(Exp)
        $.tipo = Exp.tipo

    tipado(inst_write(Exp)):
        tipado(Exp)
        $.tipo = Exp.tipo

    tipado(inst_call(id,PReales)):
        tipado(PReales)
        $.tipo = PReales.tipo

    tipado(inst_nl()): $.tipo = nl

    tipado(inst_blo(Blo)):
        tipado(Blo)
        $.tipo = Blo.tipo

    tipado(exp_asig(Opnd0,Opnd1)):
        tipado(Opnd0)
        tipado(Opnd1)
        if es-designador(Opnd0) then
            if compatibles(Opnd0.tipo, Opnd1.tipo) then
                $.tipo = ok
            else
                aviso-error(Opnd0.tipo,Opnd1.tipo)
                $.tipo = error
            end if
        else
            error
            $.tipo = error
        end if

    tipado(exp_menor(Opnd0,Opnd1)):
        tipado-bin(Opnd0,Opnd1)

    tipado(exp_menIgual(Opnd0,Opnd1)):
        tipado-bin(Opnd0,Opnd1)
```

```
tipado(exp_mayor(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_mayIgual(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_igual(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_dist(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_sum(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_resta(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_mult(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_div(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_mod(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_and(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_or(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_menos(Exp)):
    tipado(Exp)
    $.tipo = Exp.tipo

tipado(exp_not(Exp)):
    tipado(Exp)
    $.tipo = Exp.tipo

tipado(inst_index(Opnd0,Opnd1)):
    tipado-bin(Opnd0,Opnd1)

tipado(exp_reg(Exp,id)):
    tipado(Exp)
    tipado-bin(Opnd0,Opnd1)

tipado(exp_ind(Exp)):
    tipado(Exp)

tipado(exp_true()): $.tipo = true

tipado(exp_false()): $.tipo = false

tipado(exp_litEnt(litEnt)): $.tipo = literalEntero

tipado(exp_litReal(litReal)): $.tipo = literalReal
```

```
tipado(exp_litCad(litCad)): $.tipo = literalCadena

tipado(exp_iden(id)):
    let $.vinculo = Dec_var(T,I) in
        $.tipo = T
    end let

tipado(exp_null()): $.tipo = null

tipado(elem1(E)):
    tipado(E)
    if ref!(E.tipo) = par(T,_) then
        Acc.tipo = T
    else
        aviso-error(T)
        Acc.tipo = error
    end if

tipado(elem2(E)):
    tipado(E)
    if ref!(E.tipo) = par(_,T) then
        Acc.tipo = T
    else
        aviso-error(T)
        Acc.tipo = error
    end if

ambos-ok(T0,T1):
    if T0 == ok ^ T1 == ok then
        return ok
    else
        return error
    end if

aviso-error(T0,T1):
    if T0 != error ^ T1 != error then
        error
    end if

aviso-error(T):
    if T != error then
        error
    end if

ref!(T):
    if T == Ref(I) then
        let T.vinculo = Dec_tipo(T',I) in
            return ref!(T')
        end let
    else
        return T
    end if

tipado-bin(E0,E1,E):
    tipado(E0)
    tipado(E1)
    E.tipo = tipo-bin(E0.tipo,E1.tipo)

tipo-bin(T0,T1):
    if compatibles(T0,T1)  then
```

```
        return T0
    else
        aviso-error(T0,T1)
        return error
    end if

compatibles(T1,T2):
    let T1' = ref!(T1) ^ T2' = ref!(T2) in
        if T1' == T2' then
            return true;
        elsif T1' == par(T1_a,T1_b) ^ T2' == par(T2_a,T2_b)then
            return compatibles(T1_a,T2_a) ^ compatibles(T1_b,T2_b)
        else
            return false
        end if
    end let

es-designador(E):
    return E = id(v) v E = elem1(E') v E = elem2(E')
```

Índice de figuras