ANALIZADOR SINTÁCTICO TINY(1)

Grupo 29
Adrian Estevez Gallego
Alvaro Delgado Gutierrez

1. Especificación sintáctica de la gramática

Las palabras subrayadas se refieren a clases léxicas.

```
Prog -> SeccionDec SeccionInst
SeccionDec -> Decs &&
SeccionDec -> ε
Decs -> Decs ; Dec
Decs -> Dec
Dec -> var Tipo Variable
Dec -> type Tipo Variable
Dec -> proc Variable ( Params ) { Prog }
Tipo -> int
Tipo -> real
Tipo -> bool
Tipo -> string
Tipo -> Variable
Tipo -> array [ EntNum ] of Tipo
Tipo -> record { RecDecs }
Tipo -> pointer Tipo
RecDecs -> RecDecs ; RDec
RecDecs -> RDec
RDec -> Tipo Variable
Params -> ε
Params -> Params Param
Param -> Tipo Variable
Param -> Tipo & Variable
SeccionInst -> Insts
Insts -> Insts ; Inst
Insts -> Inst
Inst -> Expr0 = Expr0
Inst -> if Expr0 then OpInsts ELSE endif
Inst -> while Expr0 do OpInsts endwhile
Inst -> { Prog }
Inst -> read Expr0
Inst -> write Expr0
Inst -> nl
Inst -> new Expr0
Inst -> delete Expr0
Inst -> call Variable ( RParams )
OpInsts -> ε
```

```
OpInsts -> Insts
ELSE -> ε
ELSE -> else OpInsts
RParams -> ε
RParams -> RPars
RPars -> RPars , Expr0
RPars -> Expr0
Expr0 -> Expr1 + Expr0
Expr0 -> Expr1 - Expr1
Expr0 -> Expr1
Expr1 -> Expr1 Op1 Expr2
Expr1 -> Expr2
Expr2 -> Expr2 Op2 Expr3
Expr2 -> Expr3
Expr3 -> Expr4 Op3 Expr4
Expr3 -> Expr4
Expr4 -> - Expr5
Expr4 -> not Expr4
Expr4 -> Expr5
Expr5 -> Expr5 [ Expr0 ]
Expr5 -> Expr5 Op5 Variable
Expr5 -> Expr6
Expr6 -> * Expr6
Expr6 -> Expr7
Expr7 -> ( Expr0 )
Expr7 -> Variable
Expr7 -> EntNum
Expr7 -> RealNum
Expr7 -> true
Expr7 -> false
Expr7 -> null
Expr7 -> LitCad
Op1 -> and
Op1 -> or
Op2 -> <
Op2 -> >
Op2 -> <=
Op2 -> >=
Op2 -> ==
Op2 -> !=
Op3 -> *
Op3 -> /
Op3 -> %
Op5 -> .
```

```
Op5 -> ->
```

2. Transformaciones necesarias para obtener una gramática LL(1) equivalente

```
Prog -> SeccionDec SeccionInst
SeccionDec -> Decs &&
SeccionDec -> ε
Decs -> Dec ReDecs
ReDecs -> ; Dec ReDecs
ReDecs -> ε
Dec -> var Tipo Variable
Dec -> type Tipo Variable
Dec -> proc Variable ( Params ) { Prog }
Tipo -> int
Tipo -> real
Tipo -> bool
Tipo -> string
Tipo -> Variable
Tipo -> array [ EntNum ] of Tipo
Tipo -> record { RecordDecs }
Tipo -> pointer Tipo
RecordDecs -> RDec ReRecordDecs
ReRecordDecs -> ; RDec ReRecordDecs
ReRecordDecs -> ε
RDec -> Tipo Variable
Params -> ε
Params -> Param ReParams
ReParams -> , Param ReParams
ReParams -> ε
Param -> Tipo Ref Variable
Ref -> &
Ref -> ε
SeccionInst -> Insts
Insts -> Inst ReInsts
ReInsts -> ; Inst ReInsts
ReInsts -> ε
Inst -> Expr0 = Expr0
```

```
Inst -> if Expr0 then OpInsts ELSE endif
Inst -> while Expr0 do OpInsts endwhile
Inst -> read Expr0
Inst -> write Expr0
Inst -> nl
Inst -> new Expr0
Inst -> delete Expr0
Inst -> call Variable ( RParams )
Inst -> { Prog }
OpInsts -> ε
OpInsts -> Insts
ELSE -> ε
ELSE -> else OpInsts
RParams -> ε
RParams -> RPars
RPars -> Expr0 ReRPars
ReRPars -> , Expr0 ReRPars
ReRPars -> ε
Expr0 -> Expr1 ReExpr0
ReExpr0 -> + Expr0
ReExpr0 -> - Expr1
ReExpr0 -> ε
Expr1 -> Expr2 ReExpr1
ReExpr1 -> Op1 Expr2 ReExpr1
ReExpr1 -> ε
Expr2 -> Expr3 ReExpr2
ReExpr2 -> Op2 Expr3 ReExpr2
ReExpr2 -> ε
Expr3 -> Expr4 ReExpr3
ReExpr3 -> Op3 Expr4
ReExpr3 -> ε
Expr4 -> - Expr5
Expr4 -> not Expr4
Expr4 -> Expr5
Expr5 -> Expr6 ReExpr5
ReExpr5 -> Op5 Variable ReExpr5
ReExpr5 -> [ Expr0 ] ReExpr5
ReExpr5 -> ε
Expr6 -> * Expr6
Expr6 -> Expr7
Expr7 -> ( Expr0 )
Expr7 -> Variable
Expr7 -> EntNum
Expr7 -> RealNum
```

```
Expr7 -> true
Expr7 -> false
Expr7 -> null
Expr7 -> LitCad
Op1 -> and
Op1 -> or
Op2 -> <
Op2 -> >
Op2 -> <=
Op2 -> >=
Op2 -> ==
Op2 -> !=
Op3 -> *
Op3 -> /
Op3 -> %
Op5 -> .
Op5 -> ->
```