Gramática Livre de Contexto

S = DeclaId S
S = FunDecla S
S = ProcDecla S
S = ε

DeclaId = Type LId ‘;’
DeclaId = ‘const’ Type LId ‘;’

Type = ‘int’ | ‘float’ | ‘bool’ | ‘char’ | ‘string’

LId = LId ‘,’ ‘id’ ‘=’ Ec
LId = LId ‘,’ ‘id’
Lid = Lid ‘,’ Id
LId = ‘id’ ‘=’ Ec
LId = Id
LId = ‘id’

Id = ‘id’ ‘[‘ Ea ‘]’
Id = ‘id’

Param = Param ‘,’ Ec
Param = Ec
Param = ε

FunDecla = ‘fun’ Type FunName ‘(‘ ParamDecla‘)’ Body
FunName = ‘id’ | ‘main’

ParamDecla = ParamDecla ‘,’ Type Id
ParamDecla = Type Id
ParamDecla = Type ‘id’
ParamDecla = ε

IdOrFun = Id
IdOrFun = ‘id’ ‘(‘ Param ‘)’
IdOrFun = ‘id’

ProcDecla = ‘proc’ FunName ‘(‘ ParamDecla ‘)’ Body

Body = ‘{‘ BodyPart ‘}’

BodyPart = DeclaId BodyPart
BodyPart = Command BodyPart
BodyPart = ‘id’ ‘(‘ Param ‘)’ ‘;’ BodyPart
BodyPart = ‘return’ Return ‘;’
BodyPart = IdAtr ‘;’ BodyPart
BodyPart = ε

IdAtr = IdAtr ‘,’ ‘=’ Ec
IdAtr = IdAtr ‘,’ ‘id’ ‘[‘ Ea ‘]’ ‘=’ Ec
IdAtr = ‘id’ ‘=’ Ec
IdAtr = ‘id’ ‘[‘ Ea ‘]’ ‘=’ Ec

Return = Ec // quando for funcao
Return = ε // quando for procedimento

Command = ‘print’ ‘(‘ ‘CT_STRING’ PrintParam ‘)’ ‘;’
Command = ‘println’ ‘(‘ ‘CT_STRING’ PrintParam ‘)’ ‘;’
Command = ‘read’ ‘(‘ ReadLParam ‘)’ ‘;’
Command = ‘while’ ‘(‘ Eb ‘)’ Body
Command = ‘for’ ‘(‘ ‘RW_INT’ ‘id’ ‘:’ Ea ‘,’ Ea ‘,’ Ea ‘)’ Body
Command = ‘if’ ‘(‘ Eb ‘)’ Body
Command = ‘if’ ‘(‘ Eb ‘)’ Body ‘else’ Body

PrintParam = ‘,’ Eb PrintParam
PrintParam = ε

ReadParam = ReadParam ‘,’ Id
ReadParam = Id

Expressões:
Ec = Ec ‘OP_CONC’ Eb
Ec = Eb

Eb = Eb ‘OP_OR’ Tb
Eb = Tb
Tb = Tb ‘OP_AND’ Fb
Tb = Fb
Fb = Fb OpRel Ra
Fb = ‘OP_NOT’ Fb
Fb = Ra
Ra = Ra OpRel Ea
Ra = Ea
Ea = Ea ‘OP_AD’ Ta
Ea = Ea ‘OP_SUB’ Ta
Ea = Ta
Ta = Ta ‘OP_MULT’ Pa
Ta = Ta ‘OP_DIV’ Pa
Ta = Pa
Pa = Pa ‘OP_MOD’ Fa
Pa = Fa
Fa = ‘(‘ Ec ‘)’
Fa = ‘OP_SUB’ Fa
Fa = IdOrFun | ‘CT_INT’ | ‘CT_FLOAT’ | ‘CT_STRING’ | ‘CT_CHAR’
OpRel = ‘OP_GREATER’ | ‘OP_LESS’ | ‘OP_GREATEREQ’ | ‘OP_LESSEQ’| ‘OP_REL’ | ‘OP_RELNOT’