Gramática Livre de Contexto

Ponto de Início de Execução:

```
S = DeclId S
S = FunDecl S
S = ProcDecl S
S = &
```

Declaração de Variáveis:

```
DeclId = Type LId
DeclId = ‘const’ Type LId
LId = ‘id’ IdAttr LIdr
LIdr = ‘,’ ‘id’ IdAttr LIdr
LIdr = &
Type = ‘int’ | ‘float’| ‘bool’ | ‘char’ | ‘string’
```

Id:

```
Id = ‘id’ ArrayOpt
Id = ‘id’ FunCall
IdAttr = ArrayOpt AttrOpt
IdAttr = FunCall
```

Array:

```
ArrayOpt = ’[‘ Ea ‘]’
ArrayOpt = &
```

Atribuição:

```
AttrOpt = ‘opAttrib’ Ec
AttrOpt = &
```

Declaração de Funções:

```
FunDecl = ‘funDef’ Type FunName Param Body
FunName = ‘id’ | ‘main’
Param = ‘(‘ LParam ‘)’
LParam = Type ‘id’ ArrayOpt LParamr
LParam = &
LParamr = ‘,’ Type ‘id’ ArrayOpt LParamr
LParamr = &
LParamCall = Ec LParamCallr
```

```
      LParamCallr = ‘,’ Ec LParamCallr
      LParamCallr = &
      FunCall = ‘(‘ LParamCall ‘)’
      Return = Ec
      Return = &
```

Declaração de Procedimento:

```
      ProcDecl = ‘procDef’ ‘id’ Param Body
```

Corpo da Função / Procedimento:

```
      Body = ‘{‘ BodyPart ‘}’
      BodyPart = DeclId BodyPart
      BodyPart = LId BodyPart
      BodyPart = Command BodyPart
      BodyPart = ‘funRet’ Return ‘;’
      BodyPart = &
```

Comandos:

```
      PrintParam = ‘,’ LParamCall
      PrintParam = &
      Command = ‘print’ ‘(‘ Ec PrintParam ‘)’ ‘;’
      Command = ‘scan’ ‘(‘ LParamCall ‘)’ ‘;’
      Command = ‘whileLoop’ ‘(‘ Eb ‘)’ Body
      Command = ‘forLoop’ ‘(‘ ‘typeInt’ ‘id’ ‘:’ ‘(‘ Ec ‘,’ Ec ‘,’
Ec ‘)’ ’)’ Body
      Command = ‘if’ ‘(‘ Eb ‘)’ Body Ifr
      Ifr = ‘condElseIf’ ‘(‘ Eb ‘)’ Body Ifr
      Ifr = ‘condElse’ Body
      Ifr = &
```

Expressões:

```
      1. Ec = Ec ‘opConcat’ Fc
      2. Ec = Fc
      3. Fc = ‘strConst’
      4. Fc = ‘charConst’
      5. Fc = Eb
      6. Eb = Eb ‘opOr’ Tb
      7. Eb = Tb
      8. Tb = Tb ‘opAnd’ Fb
      9. Tb = Fb
      10. Fb = Fb OpRel Ra
```

11. Fb = ‘opNot’ Fb
12. Fb = ‘cteBool’
13. Fb = Ra
14. Ra = Ra OpRel Ea
15. Ra = Ea
16. Ea = Ea ‘opAdd’ Ta
17. Ea = Ea ‘opSub’ Ta
18. Ea = Ta
19. Ta = Ta ‘opMult’ Pa
20. Ta = Ta ‘opDiv’ Pa
21. Ta = Pa
22. Pa = Pa ‘opPow’ Fa
23. Pa = Fa
24. Fa = ‘(‘ Ec ‘)’
25. Fa = ‘opSub’ Fa
26. Fa = Id | ‘cteInt’ | ‘cteFloat’
27. OpRel = ‘opGreater’ | ‘opLesser’ | ‘opGreq’ | ‘opLeq’