UNIVERSIDADE FEDERAL DE ALAGOAS - UFAL
INSTITUTO DE COMPUTAÇÃO - IC

ERIC DOS SANTOS COELHO

# ESPECIFICAÇÃO DA GRAMÁTICA ESC

Trabalho apresentado à disciplina de Compiladores, ministrada pelo professor Alcino Dall Igna Junior.

MACEIÓ
2019.1

# Sumário

## Tipo de analisador sintático

Analisador descendente LL(1) preditivo recursivo

## Gramática

***Início***

```
Início = ListaFunc Início
       | ListaProc Início
       | ListaVar Início
       | ε

ListaFunc  = Função ListaFunc | ε

ListaProc  = Proc ListaProc | ε

ListaVar   = Variável ListaVar | ε
```

***Função***

```
Função = 'func' Tipo 'id' '(' Parâmetros ')' Bloco
       | 'func' Tipo 'id' '(' Parâmetros ')' ';'
```

***Procedimento***

```
Proc = 'proc' 'id' '(' Parâmetros ')' Bloco
     | 'proc' 'id' '(' Parâmetros ')' ';'
```

***Variável***

```
Variável   = Tipo ListaId ';'
           | 'const' Tipo ListaAtr ';'

ListaAtr   = ListaAtr ',' 'id' Atribuição
           | ListaAtr ',' 'id' Array Atribuição
           | 'id' Atribuição
           | 'id' Array Atribuição
```

***Identificadores***

```
ListaId    = ListaId ',' Id
           | Id

Id = 'id' | 'id' Array
   | 'id' Atribuição | 'id' Array Atribuição
   | 'id' '(' Argumentos ')'
```

```
Array = '[' ']' | '[' ExprBool ']'
```

***Bloco***

```
Bloco = '{' ListaSentenças '}'
```

***Sentenças***

```
ListaSentenças = Sentença ListaSentenças | ε

Sentença   = If | While | For | Desvio | Return
           | Entrada | Saída | Id ';' | ListaVar

Entrada = 'input' Argumentos ';'

Saída = 'print' CteStr Argumentos ';'

If     = 'if' '(' ExprBool ')' Bloco ElseIf Else
ElseIf = 'else if' '(' ExprBool ')' Bloco ElseIf
       | ε
Else   = 'else' Bloco
       | ε

While = 'while' '(' ExprBool ')' Bloco

For ='for' Id 'in' '(' ExprBool ',' ExprBool ')' 'step' ExprBool Bloco

Desvio = 'break' ';'

Return = 'return' ExprBool ';'

Atribuição = OpAtribuição ExprBool
```

***Parâmetros***

```
Parâmetros = ListaParam | ε
ListaParam = ListaParam ',' Tipo Id
           | Tipo Id
```

***Argumentos***

```
Argumentos = ListaArgs | ε
ListaArgs  = ListaArgs ',' ExprBool
           | ExprBool
```

***Tipos***

```
Tipo       = 'int' | 'float' | 'char' | 'bool' | 'string'
```

***Expressões***

```
ExprBool   = ExprBool OpLogic TermoBool
           | TermoBool

TermoBool  = ExprConcat OpRelac ExprConcat
           | ExprConcat
           | '!' TermoBool

ExprConcat = ExprConcat OpConcat ExprAritm
           | ExprAritm

ExprAritm  = ExprAritm OpAritm TermoAritm
           | TermoAritm

TermoAritm = TermoAritm OpMult FatorAritm
           | FatorAritm

FatorAritm = '(' Tipo ')' ExprBool
           | Id
           | CteInt
           | CteFloat
           | CteChar
           | CteBool
           | CteStr
           | '(' ExprBool ')'
           | '[' ListaArray ']'

ListaArray  = ListaArray ',' ExprBool
            | ExprBool
```

***Operadores***

```
OpUnario   = '-' | '!'
OpRelac    = '>' | '<' | '>=' | '<=' | '==' | '!='
OpLogic    = 'and' | 'or'
OpAdit     = '+' | '-'
OpMult     = '*' | '/' | '%'
OpAtr      = '='
OpConcat   = '++'
```

# Gramática LL(1)

***Início***

```
Início = ListaFunc Início
       | ListaProc Início
       | ListaVar Início
       | ε

ListaFunc  = Função ListaFunc | ε

ListaProc  = Proc ListaProc | ε

ListaVar   = Variável ListaVar | ε
```

***Função***

```
Função  = 'func' Tipo 'id' '(' Parâmetros ')' FunçãoF
FunçãoF = Bloco | ';'
```

***Procedimento***

```
Proc  = 'proc' 'id' '(' Parâmetros ')' ProcF
ProcF = Bloco | ';'
```

***Variável***

```
Variável   = Tipo ListaId ';'
           | 'const' Tipo ListaAtr ';'

ListaAtr   = 'id' ListaAtrF ListaAtrR
ListaAtrR  = ',' 'id' ListaAtrF ListaAtrR
           | ε

ListaAtrF  = Atribuição
           | Array Atribuição
```

***Identificadores***

```
ListaId    = Id ListaIdR
ListaIdR   = ',' Id ListaIdR
           | ε

Id = 'id' IdF
IdF = '(' Argumentos ')' | Atribuição | Array IdFF | ε
IdFF = Atribuição | ε
```

```
Array  = '[' ArrayF
ArrayF = ']' | ExprBool ']'
```

***Bloco***

```
Bloco = '{' ListaSentenças '}'
```

***Sentenças***

```
ListaSentenças = Sentença ListaSentenças | ε

Sentença   = If | While | For | Desvio | Return | Entrada | Saída
           | Id ';' | ListaVar

Entrada = 'input' Argumentos ';'

Saída = 'print' CteStr Argumentos ';'

If     = 'if' '(' ExprBool ')' Bloco ElseIf Else
ElseIf = 'else if' '(' ExprBool ')' Bloco ElseIf
       | ε
Else   = 'else' Bloco
       | ε

While = 'while' '(' ExprBool ')' Bloco

For ='for' Id 'in' '(' ExprBool ',' ExprBool ')' 'step' ExprBool Bloco

Desvio = 'break' ';'

Return = 'return' ExprBool ';'

Atribuição = OpAtribuição ExprBool
```

***Parâmetros***

```
Parâmetros = ListaParam | ε

ListaParam  = Tipo Id ListaParamR
ListaParamR = ',' Tipo Id ListaParamR
            | ε
```

***Argumentos***

```
Argumentos = ListaArgs | ε
```

```
ListaArgs  = ExprBool ListaArgsR
ListaArgsR = ',' ExprBool ListaArgsR
           | ε
```

***Tipos***

```
Tipo       = 'int' | 'float' | 'char' | 'bool' | 'string'
```

***Expressões***

```
ExprBool   = TermoBool ExprBoolR
ExprBoolR  = OpLogic TermoBool ExprBoolR | ε

TermoBool  = ExprConcat TermoBoolF
           | '!' TermoBool
TermoBoolF = OpRelac ExprConcat | ε

ExprConcat  = ExprAritm ExprConcatR
ExprConcatR = OpConcat ExprAritm ExprConcatR | ε

ExprAritm  = TermoAritm ExprAritmR
ExprAritmR = OpAritm TermoAritm ExprAritmR | ε

TermoAritm  = FatorAritm TermoAritmR
TermoAritmR = OpMult FatorAritm TermoAritmR | ε

FatorAritm  = '(' FatorAritmF
            | Id
            | CteInt
            | CteFloat
            | CteChar
            | CteBool
            | CteStr
            | '[' ListaArray ']'
FatorAtirmF = Tipo ')' ExprBool
            | ExprBool ')'


ListaArray  = ExprBool ListaArrayR
ListaArrayR = ',' ExprBool ListaArrayR | ε
```

***Operadores***

```
OpUnario   = '-' | '!'
OpRelac    = '>' | '<' | '>=' | '<=' | '==' | '!='
OpLogic    = 'and' | 'or'
```

```
OpAdit     = '+' | '-'
OpMult     = '*' | '/' | '%'
OpAtr      = '='
OpConcat   = '++'
```