UNIVERSIDADE FEDERAL DE ALAGOAS - UFAL
COMPILADORES 2018.2
INSTITUTO DE COMPUTAÇÃO - IC

IAGO BARBOZA DE SOUZA

Especificação da gramática de linguagem HI

Maceió
Abril de 2019

1. Ajustes nas categorias dos tokens

Algumas categorias foram agrupadas em uma única categoria e outras removidas com o objetivo de simplificar a gramática e consequentemente a análise.

1.1. Agrupamentos

- logicOp
  - andLogicOp ('*and*')
  - orLogicOp ('*or*')

- relOp
  - ineqRelOp ('<' | '>' | '<=' | '>=')
  - EqRelOp ('==' | '!=')

1.2. Remoções

- main ('main') - Todos os nomes de função são classificados como *TokenCategory.ID*

2. Gramática LL(1)

```
Program = LDecl

LDecl = Decl LDeclR
LDeclR = Decl LDeclR
LDeclR = epsilon

Decl = VarDecl
Decl = FuncDecl

VarDecl = 'var' Type 'id' VarDeclAux '#'
VarDeclAux = '[' AritExpr ']' Atr
VarDeclAux = Atr
```

```
Atr = '=' ConcatExp
Atr = epsilon

FuncDecl = 'function' FType 'id' '(' LParam ')' '{' LSent
'}'

FType = 'none'
FType = Type
Type = 'int'
Type = 'float'
Type = 'char'
Type = 'string'
Type = 'boolean'

LParam = Param LParamR
LParam = espison

LParamR = ',' Param LParamR
LParamR = epsilon

Param = Type 'id' ParamR

ParamR = '[' ']'
ParamR = epsilon

LSent = Sent LSent
LSent = epsilon

Sent = VarDecl
Sent = Command

Command = 'id' CommandR '#'
Command = Print '#'
Command = Read '#'
Command = If
Command = While
Command = Repeater
Command = Return
```

```
CommandR = Atr
CommandR = '[' AritExpr ']' Atr
CommandR = '(' LArg ')'

LArg = Arg LArgR
LArg = epsilon

LArgR = ',' Arg LArgR
LArgR = epsilon

Arg = ConcatExp

Print = 'print' '(' ConcatExp ')'

Read = 'read' '(' LReadParam ')'

LReadParam = ReadParam LReadParamR
LReadParamR = ',' ReadParam LReadParamR
LReadParamR = epsilon

ReadParam = 'id' ReadParamR
ReadParamR = '[' AritExp' ]'
ReadParamR = epsilon

If = 'if' '(' BooleanExp ')' '{' LSent '}' Else

Else = 'else' '{' LSent '}'
Else = epsilon

While = 'while' '(' BooleanExp ')' '{' LSent '}'

Repeater = repeater '(' 'id' 'of' AritExp 'to' AritExp ','
Step ')' '{' LSent '}'

Step = AritExp
Step = epsilon

Return = 'return' ConcatExp '#'
```

```
ConcatExp = BooleanExp ConcatExpR
ConcatExpR = '++' BooleanExp ConcatExpR
ConcatExpR = epsilon

BooleanExp = BooleanTerm BooleanExpR
BooleanExpR = 'logicOp' BooleanTerm BooleanExpR
BooleanExpR = epsilon

BooleanTerm = AritExp BooleanTermR
BooleanTermR = 'relOp' AritExp BooleanTermR
BooleanTermR = epsilon

AritExp = AritTerm AritExpR
AritExpR = 'addAritOp' AritTerm AritExpR
AritExpR = epsilon

AritTerm = AritFact AritTermR
AritTermR = 'multAritOp' AritFact AritTermR
AritTermR = epsilon

AritFact = 'id' AritFatcR
AritFact = '-' AritFact
AritFact = 'intNumConst'
AritFact = 'decNumConst'
AritFact = 'charConst'
AritFact = 'stringConst'
AritFact = 'logicConst'

AritFactR = '(' LArg ')'
AritFactR = '[' AritExp ']'
AritFactR = epsilon
```