

Especificação dos tokens - Linguagem L--

Lucas Montenegro e Letícia Medeiros

2020.1

Universidade Federal de Alagoas - Instituto de Computação
Compiladores
Professor Alcino Dall Igna Junior

Conteúdo

1	Especificações dos tokens da linguagem	3
1.1	Linguagem utilizada	3
1.2	Lista de Tokens	3
1.2.1	Expressões Regulares Auxiliares	3
1.3	Expressões Terminais	3
1.3.1	Main	3
1.3.2	Identificador	3
1.3.3	Delimitadores	3
1.3.4	Atribuição	4
1.3.5	Função	4
1.3.6	Leitura e Escrita	4
1.3.7	Palavras Reservadas de estruturas condicionais e de repetição	4
1.3.8	Operadores	4
1.3.9	Tipos	5
1.3.10	Constantes	5
1.3.11	Token Não reconhecido	5

1 Especificações dos tokens da linguagem

1.1 Linguagem utilizada

Para o desenvolvimento do analisador léxico foi utilizada a linguagem Java.

1.2 Lista de Tokens

```
public enum TokenClass {  
    UNKNOWN, MAIN, ID, TYPEVOID, TYPEINT, TYPEFLOAT, TYPECHAR, TYPE-  
    BOOL, TYPESTRING, TYPEARRAY, NULL, BREAK, CONSTINT, CONSTFLOAT, CONST-  
    BOOL, CONSTCHAR, CONSTSTRING, OPTADD, OPTPOW, OPTMULT, OPTDIV, OPT-  
    SUB, OPTMOD, OPTCONCAT, ATRIB, OPTLESS, OPTLESSEQ, OPTGREAT, OPTGRE-  
    ATEQ, OPTEQ, OPTNOTEQ, OPTAND, OPTOR, OPTNOT, CONDIF, CONDELSE, CON-  
    DELSE, LOOPWHILE, LOOPFOR, READ, WRITE, SEPARATOR, PARAMINIT, PARA-  
    MEND, ARRINIT, ARREND, FUNCDEF, FUNCRETURN, FUNCINIT, FUNCEND, SEN-  
    TENEND, ENDFILE  
}
```

1.2.1 Expressões Regulares Auxiliares

Segue abaixo as expressões regulares auxiliares:

- letra: [a - zA- Z]
- dígito: [0 - 9]
- símbolo: [' ' | ' ; ' | ' , ' | ' . ' | ' : ' | ' ? ' | ' ~ ' | ' ! ' | ' + ' | ' - ' | ' * ' | ' \ ' | ' / ' | ' _ ' | ' % ' | ' @ ' | ' & ' | ' # ' | ' \$ ' | ' < ' | ' > ' | ' = ' | ' (' | ') ' | ' [' | '] ' | ' { ' | ' } ' | ' | ' | ' \ ' | ' ^ ' | ' \n '];

1.3 Expressões Terminais

1.3.1 Main

MAIN: 'main'

1.3.2 Identificador

ID: ('{letra}') (('{letra}' | '{dígito}' | '_'*)

1.3.3 Delimitadores

Terminador

SENTENCEEND: ';'

ENDFILE: '< <EOF> >'

Separador

SEPARATOR: ','

Escopos

FUNCINIT: '{'

FUNCEND: '}'

Parâmetros

PARAMINIT: '('

PARAMEND: ')'

Array

ARRINIT: '['

ARREND: ']'

1.3.4 Atribuição

ATRIB: '='

1.3.5 Função

FUNCDEF: 'function'

FUNCRETUR: 'return'

1.3.6 Leitura e Escrita

READ: 'read'

WRITE: 'write'

1.3.7 Palavras Reservadas de estruturas condicionais e de repetição

CONDIF: 'if'

CONDELSIF: 'elsif'

CONDELSE: 'else'

LOOPFOR: 'for'

LOOPWHILE: 'while'

BREAK: 'break'

1.3.8 Operadores**Lógicos**

OPTAND: '&'

OPTNOT: '!'

OPTOR: '||'

Relacionais

OPTLESS: '<'

OPTLESSEQ: '<='
OPTGREAT: '>'
OPTGRATEQ: '>='
OPTEQ: '=='
OPTNOTEQ: '!='

Aritméticos

OPTADD: '+'
OPTSUB: '-'
OPTPOW: '^'
OPTMULT: '*'
OPTDIV: '/'
OPTMOD: '%'

1.3.9 Tipos

TYPEINT: 'int'
TYPEFLOAT: 'float'
TYPECHAR: 'char'
TYPEBOOL: 'bool'
YPESTRING: 'string';
TYPEARRAY: 'array';
TYPEVOID: 'void';

1.3.10 Constantes

CONSTINT: '{dígito}'+
CONSTFLOAT: ('{dígito}')+\.('{dígito}')+
CONSTBOOL: ('*true*' | '*false*')
CONSTCHAR: (' \ ') ('{letra}' | '{símbolo}' | '{dígito}') (' \ ')
CONSTSTRING: ('\"") (('{letra}' | '{símbolo}' | '{dígito}')+) ('\"")
NULL: 'null'

1.3.11 Token Não reconhecido

UNKNOWN: Token inválido.