## Sumário

[**Sumário**](#_hkllq2mjrst) **1**

[**Linguagem utilizada**](#_9vftmnmy0r8z) **2**

[**Enumeração das categorias dos tokens**](#_t4kj9i5zos71) **2**

[**Expressões regulares auxiliares**](#_71c0mbpjgt6z) **3**

[**Categorias**](#_3dn4rnub6ds5) **3**

### **Linguagem utilizada**

Os analisadores léxico e sintático serão implementados em Python.

### **Enumeração das categorias dos tokens**

from enum import Enum, auto

class TokenCategory(Enum):

ID = auto()

TYPEBOOL = auto()

TYPECHAR = auto()

TYPESTRING = auto()

TYPEINT = auto()

TYPEFLOAT = auto()

OPASUM = auto()

OPAMINUS = auto()

OPADIV = auto()

OPAMULT = auto()

OPRLESS = auto()

OPRGREAT = auto()

OPREQL = auto()

OPRLESSEQL = auto()

OPRGREATEQL = auto()

OPRNOTEQL = auto()

ARRAYBEGIN = auto()

ARRAYEND = auto()

PARAMSBEGIN = auto()

PARAMSEND = auto()

FCBEGIN = auto()

FCEND = auto()

FCDEF = auto()

FCRETURN = auto()

LINEEND = auto()

SEPARATOR = auto()

ASSIGN = auto()

TYPEASSIGN = auto()

FCTYPEVOID = auto()

CONSTLINT = auto()

CONSTLSTRING = auto()

CONSTLBOOL = auto()

CONSTLFLOAT = auto()

CONSTLCHAR = auto()

### **Expressões regulares auxiliares**

| letra | [a-zA-Z] |
| --- | --- |
| dígito | [0-9] |

### **Categorias**

| **Categoria** | **Lexema** |
| --- | --- |
| ID | ('{letra}')(('{letra}’ | ‘{dígito}' | ‘\_’)\*) |
| TYPEBOOL | ‘boolean’ |
| TYPECHAR | ‘char’ |
| TYPESTRING | ‘string’ |
| TYPEINT | ‘int’ |
| TYPEFLOAT | ‘float’ |
| OPASUM | ‘+’ |
| OPAMINUS | ‘-’ |
| OPADIV | ‘/’ |
| OPAMULT | ‘\*’ |
| OPRLESS | ‘<’ |
| OPRGREAT | ‘>’ |
| OPREQL | ‘==’ |
| OPRLESSEQL | ‘<=’ |
| OPRGREATEQL | ‘>=’ |
| OPRNOTEQL | ‘!=’ |
| ARRAYBEGIN | ‘[’ |
| ARRAYEND | ‘]’ |
| PARAMSBEGIN | ‘(’ |
| PARAMSEND | ‘)’ |
| FCBEGIN | ‘begin’ |
| FCEND | ‘end’ |
| FCDEF | ‘def’ |
| FCRETURN | ‘return’ |
| LINEEND | ‘;’ |
| SEPARATOR | ‘,’ |
| ASSIGN | ‘=’ |
| TYPEASSIGN | ‘:’ |
| FCTYPEVOID | ‘void’ |
| CONSTLINT | '{dígito}'+ |
| CONSTLSTRING | (‘\”’)((‘{letra}’ | ‘{dígito}’)+)(‘\”’) |
| CONSTLBOOL | (‘true’ | ‘false’) |
| CONSTLFLOAT | (‘{dígito}’)+\.(‘{dígito}’)+ |
| CONSTLCHAR | (‘\’’)(‘{letra}’ | ‘{dígito}’)(‘\’’) |
|  |  |