UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

Especificações dos Tokens **Valcode**

Luís Antônio da Silva Nascimento, Vinicius Monteiro Pontes

Maceió, AL - 2021

1. Development Language

Valcode uses Python (3.8.5 version) to implement its lexical and syntactic analyzer.

2. Tokens category enum

```
class TokenCategory(Enum):
    IDENTIFIER = auto()
    RW_INIT = auto()
    OP_PAR = auto()
    CL_PAR = auto()
    OP_CBRA = auto()
    CL_CBRA = auto()
    OP_BRA = auto()
    CL_BRA = auto()
    RW_FN = auto()
    RW_INT = auto()
    RW_FLOAT = auto()
    RW_BOOL = auto()
    RW_CHAR = auto()
    RW_NULL = auto()
    RW_VOID = auto()
    RW_STR = auto()
    RW_RTN = auto()
    RW_IF = auto()
    RW_ELSE = auto()
    RW_WHILE = auto()
    RW_FOR = auto()
    RW_VAR = auto()
    RW_AND = auto()
    RW_OR = auto()
    RW_NOT = auto()
    RW_INPUT = auto()
    RW_PRINT = auto()
    SB_COLON = auto()
    SB_SCOLON = auto()
    SB_COMMA = auto()
    SB_ASGN = auto()
    OPR_ADD = auto()
    OPR_SUB = auto()
    OPR_MUL = auto()
    OPR_DIV = auto()
    OPR_UNG = auto()
    OPR_FDIV = auto()
    OPR_EQUAL = auto()
    OPR_GT = auto()
    OPR_LT = auto()
```

```
OPR_GTE = auto()
OPR_LTE = auto()
CL_INT = auto()
CL_FLOAT = auto()
CL_BOOL = auto()
CL_CHAR = auto()
CL_STR = auto()
```

3. Auxiliary Regular Expressions

Integer literal constants: /[+-]?\d+$/
Float literal constants: /[+-]?\d\.\d+/
Bool literal constants: /true|false/
Char literal constants: /^”.”$/
String literal constants: /^”.*”$/
Identifier: /^[A-Za-z]+[A-Za-z0-9_]*/

3. Lexemes Regular Expressions

Identifiers	
IDENTIFIER	Identifier

Reserved words	
RW_FN	‘function’
RW_RTN	‘return’
RW_IF	‘if’
RW_ELSE	‘else’
RW_WHILE	‘while’
RW_FOR	‘for’
RW_INT	‘int’
RW_FLOAT	‘float’
RW_CHAR	‘char’
RW_STR	‘string’

RW_BOOL	‘bool’
RW_INPUT	‘input’
RW_PRINT	‘print’
RW_NULL	‘null’
RW_VOID	‘void’
RW_INIT	‘init’

Operators	
OPR_ASGN	‘=’
OPR_EQ	‘==’
OPR_DIFF	‘!=’
OPR_ADD	‘+’
OPR_SUB	‘-’
OPR_MUL	‘*’
OPR_DIV	‘/’
OPR_FDIV	‘//’
OPR_MOD	‘%’
OPR_GT	‘>’
OPR_LT	‘<’
OPR_GTE	‘>=’
OPR_LTE	‘<=’
OPR_AND	‘and’
OPR_OR	‘or’
OPR_NOT	‘not’
OPR_UNG	‘ung’

Delimiters	
OP_CBRA	‘{’
CL_CBRA	‘}’
OP_PAR	‘(‘
CL_PAR	‘)’
OP_BRA	‘[‘
CL_BRA	‘]’
SB_SCOLON	‘;’
SB_COMMA	‘,’

Literal constants	
CT_BOOL	bool literal constant
CT_INT	int literal constant
CT_FLOAT	float literal constant
CT_CHAR	char literal constant
CT_STRING	string literal constant