Análise Léxica da Linguagem TPP

Guilherme B. Del Ro[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR) – Campo Mourão, PR – BRASIL

guilhermerio@alunos.utfpr.edu.br

Abstract. *This document describe the lexical analyzer implemented using the pyhton library PLY, introducing a new didactic programming language called TPP (or T++), besides describing how the lexical algorithm works, the reserved words and the regular expressions, it will explain the creation of the TOKENS and their operation, resulting in one output file with the verified TOKENS of the input file with the TPP extension.*

Resumo. *Este documento descreve um analizador léxico implementado utilizando a biblioteca python PLY, introduzindo uma nova linguagem de programação didática chamada TPP (ou T++), além de descrever o funcionamento do algoritmo léxico, das palavras reservadas e das expressões regulares utilizadas, explicará a criação dos TOKENS e seu funcionamento resultando em um arquivo de saída com os TOKENS verificados de algum arquivo de entrada com a extenção TPP.*

1. Especificação da linguagem de programação TPP ou T++

Esta linguagem foi desenvolvida de forma simples, com objetivo de aprendizado e desenvolvimento do compilador, por isso suas palavras reservadas são em português.

Os tipos básicos suportados pela linguagem TPP são **inteiro** e **flutuante**, com arranjos uni e bidimensionais, ou seja, vetores (arrays) com uma ou duas dimensões em que são declarados como o exemplo: **tipo: identificador[dim]** ou **tipo: identificador[dim][dim]**. As variáveis locais podem ser de um dos tipos definidos anteriormente (inteiro ou flutuante).

A funções podem tem tipos, como por exemplo **inteiro** ou podem ter um tipo omitido, sendo assim retornarão o **tipo** void explicitamente. Ao ocorrerem erros, estes devem sempre ser avisados, e alguns especificados.

1.1. Palavras reservadas

Os tokens representam as palavras reservadas de forma unitária, sendo elas (veja Tabela 1):

Palavras Reservadas	TOKEN
se	SE
repita	REPITA
fim	FIM
leia	LEIA
retorna	RETORNA
escreva	ESCREVA
inteiro	INTEIRO
flutuante	FLUTUANTE
até	ATE
senão	SENAO
então	ENTAO

Tabela 1. Palavras reservadas e respectivos TOKENS da linguagem TPP

2. Especificação Formal dos Automatos

As expressões regulares são formadas por simbolos, sendo eles de operação, atribuição e entre outros, por operadores lógicos e operadores relacionais além das expressões do ID, números inteiros, numeros com ponto flutuante e notação cientifica que são mais complexos (veja Tabela 2 e 3).

Tokens	Expressões regulares simples
MAIS	r'\+'
MENOS	r'-'
MULTIPLICACAO	r'*'
DIVISAO	r'/'
ABRE_PARENTESE	r'\('
FECHA_PARENTESE	r'\)'
ABRE_COLCHETE	r''
FECHA_COLCHETE	r''
VIRGULA	r','
ATRIBUICAO	r':='
DOIS_PONTOS	r':'
E_LOGICO	r'&&'
OU_LOGICO	r'\—\—'
NEGACAO	r'!'
DIFERENCA	r'<>'
MENOR_IGUAL	r'<='
MAIOR_IGUAL	r'>='
MENOR	r'<'
MAIOR	r'>'
IGUAL	r'='

Tabela 2. Expressões regulares e respectivos TOKENS da linguagem TPP

TOKENS	Expressões regulares complexas
ID	r”(”+ letra + r”(”+ digito + r”+ — _ —”+ letra + r”)*)”
COMENTARIO	r”({((.—)*?)})”
NUM_INTEIRO	r”\d+”
NUM_PONTO_FLUTUANTE	r’\d+[eE][-+]?\d+—(\.\d+—\d+\.\d*)([eE][-+]?\d+)?’
NUM_NOTACAO_CIENTIFICA	r”(”+ sinal + r”([1-9])\.”+ digito + r”+[eE]”+ sinal + digito + r”+)”

Tabela 3. Expressões regulares complexas e respectivos TOKENS da linguagem

3. Implementação

3.1. Ferramenta utilizada

A ferramenta escolhida para a analáise léxica foi a biblioteca do python chamada **PLY** que utiliza *lex* e *yacc*, ou seja, gera um analizador léxico (*lex*) criando tokens a partir da entrada do programa, e então gera um analisador (*yacc*), porém, como as palavras reservadas e tokens foram criados no código, foi utilizado apenas o *yacc*.

3.2. Explicação do código

Após a definição das expressões regulares e da criação de uma lista com todos os tokens utilizados na linguagem, definimos a relação de cada expressão regular com seu devido token. Então é feita a criação do lexer e é verificado se o formato do arquivo é TPP e caso for, o arquivo é aberto e enviado para o lexer (código de entrada) para que seja separado em tokens e impresso na tela.

3.3. Exemplos de saída do sistema

Ao executarmos o lexer com o código de entrada ”teste-003.tpp”(ver figura 1), será obtido uma lista de tokens referente as palavras reservadas e/ou expressões presentes (ver Figura 2).

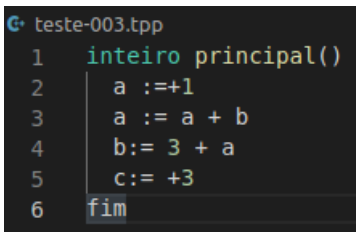

Figura 1. Código de teste: teste-003.tpp

Observa-se que para executar o programa utilizamos o seguinte comando: **python3 lexer.py teste-003.tpp**, onde o ultimo arquivo necessariamente deve conter a extensão **.tpp**.

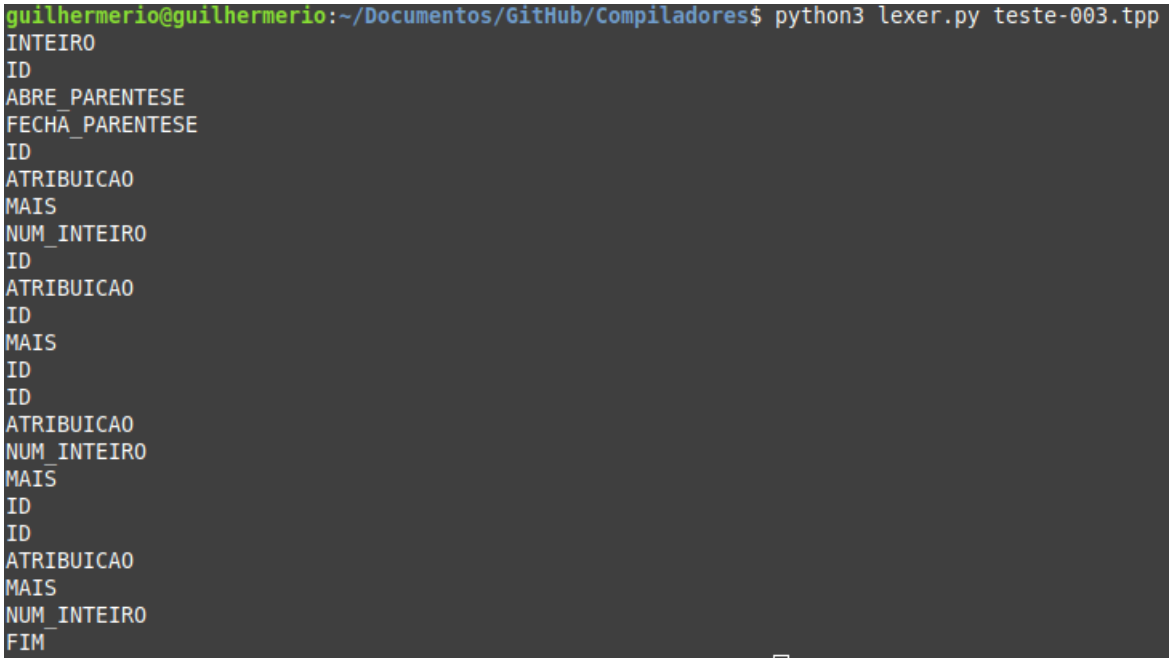

Figura 2. Saída gerada utilizando código de teste