# Análise léxica

**Thiago Alexsander**[1]

[1]Universidade Tecnológica Federal Do Paraná (UTFPR)
R. Rosalina Maria Ferreira, 1233 - Vila Carola, Campo Mourão - PR, 87301-899

[2]DACOM –
Universidade Tecnológica Federal Do Paraná – Campo Mourão, PR – Brazil

thiago.2014@alunos.utfpr.edu.br

***Abstract.***

***Resumo.*** *Este artigo tem como objetivo esclarecer e objetificar o desenvolvimento de uma análise léxica pertencente a um compilador da linguagem T++.*

## 1. Introdução

O primeiro estado de um compilador é percorrer o arquivo alvo, separando os caracteres em marcadores (*tokens*), por meio de expressões regulares cada *token* é identificado em símbolos especiais ou identificadores, este passo é denominado de Análise léxica.

A análise léxica foi desenvolvida utilizando a linguagem Python em conjunto com uma biblioteca especifica para o aprendizado de desenvolvimento de compiladores (PLY), essa biblioteca oferece ferramentas para realizar a análise léxica e sintática.

## 2. T++

A linguagem T++ é uma linguagem que oferece suporte aos tipos básicos de dados, como inteiro, flutuante, notação científica e *arrays*, porém não possui suporte a caracteres de texto, como *strings* e *char*. Outra característica da do T++ é que sua sintaxe é completamente em português, desse modo ajudando falantes da língua portuguesa a ter um primeiro contato com a programação.

## 3. Palavras reservadas

As palavras reservadas em T++ podem ser vista na Figura 1, a primeira coluna se refere a como a palavra é encontrada no arquivo alvo, já a segunda coluna é como essa palavra reservada é interpretada pela análise léxica.

Tabela 1. Palavras reservadas

| | |
|---|---|
| ’se’ | ”SE” |
| ’então’ | ’ENTAO’ |
| ’senão’ | ’SENAO’ |
| ’até’ | ’ATE’ |
| ’repita’ | ’REPITA’ |
| ’flutuante’ | ’FLUTUANTE’ |
| ’retorna’ | ’RETORNA’ |
| ’leia’ | ’LEIA’ |
| ’escreva’ | ’ESCREVA’ |
| ’inteiro’ | ’INTEIRO’ |
| ’fim’ | ’FIM’ |

**Figura 1. Palavras reservadas**

## 4. Lista de símbolos

A lista de símbolos é composta por *tokens* que representam alguma aritmética ou atribuição de valores dentro da linguagem.

**Tabela 1. Símbolos**

| |
|---|
| + |
| _ |
| * |
| / |
| = |
| , |
| := |
| < |
| > |
| >= |
| <= |
| ( |
| ) |
| : |
| [ |
| ] |
| && |
| ! |
| \|\|\| |

### 4.1. Funções auxiliares

Para o funcionamento satisfatório da análise léxica utilizou-se de funções para realizar a identificação do *tokens*, ID e símbolos.

A função t_ID é a identificadora de ID que usa da expressões regular [a-zA-ZÀ-ÿ_][a-zÀ-ÿ_0-9]* de forma a identificar se é uma palavra reservada.

A função t_NOTCIENTIFICA identifica se um dado número é em forma de notação cientifica fazendo o uso da expressão regular ([+-]?(d+)(.d+)([eE][+—-]?(d+)))

t_NUMFLUTUANTE identifica se o número é de ponto flutuante, usando a seguinte expressão regular ([+-]?d+)?((̇d)+)—((d))̇

t_NUMINTEIRO identifica um numero inteiro usando a expressão regular ([+-]?d+)

t_newline para identificar uma nova linha n+

## 5. Figures and Captions

t_error que acusa caso exista um erro em alguma linha do arquivo alvo.

## 6. Exemplos

Com uma entrada na linguagem T++ como mostra a Figura 2

```
inteiro: n
inteiro: b

inteiro fat( flutuante: a, inteiro: b)
    a:= 10.5
fim

inteiro fatorial(inteiro: n)
    inteiro: fat
    se n > 0 então {não calcula se n > 0}
        fat := 1
        repita
            fat := fat * n
            n := n - 1
        até n = 0
        retorna(fat) {retorna o valor do fatorial de n}
    senão
        retorna(0)
    fim
fim

inteiro principal()
    leia(n)
    escreva(fatorial(n))
    escreva(fat(1,1))
fim
```

**Figura 2. Exemplo entrada**

O resultado da análise léxica ficou como mostra a Figura 3

```
1   Tipo:[INTEIRO] Valor:[inteiro] Linha:[1]
2   Tipo:[DOISPONTOS] Valor:[:] Linha:[1]
3   Tipo:[ID] Valor:[n] Linha:[1]
4   Tipo:[INTEIRO] Valor:[inteiro] Linha:[2]
5   Tipo:[DOISPONTOS] Valor:[:] Linha:[2]
6   Tipo:[ID] Valor:[b] Linha:[2]
7   Tipo:[INTEIRO] Valor:[inteiro] Linha:[4]
8   Tipo:[ID] Valor:[fat] Linha:[4]
9   Tipo:[APAREN] Valor:[(] Linha:[4]
10  Tipo:[FLUTUANTE] Valor:[flutuante] Linha:[4]
11  Tipo:[DOISPONTOS] Valor:[:] Linha:[4]
12  Tipo:[ID] Valor:[a] Linha:[4]
13  Tipo:[VIRGULA] Valor:[,] Linha:[4]
14  Tipo:[INTEIRO] Valor:[inteiro] Linha:[4]
15  Tipo:[DOISPONTOS] Valor:[:] Linha:[4]
16  Tipo:[ID] Valor:[b] Linha:[4]
17  Tipo:[FPAREN] Valor:[)] Linha:[4]
18  Tipo:[ID] Valor:[a] Linha:[5]
19  Tipo:[ATRIBUICAO] Valor:[:=] Linha:[5]
20  Tipo:[NUMFLUTUANTE] Valor:[10.5] Linha:[5]
21  Tipo:[FIM] Valor:[fim] Linha:[6]
22  Tipo:[INTEIRO] Valor:[inteiro] Linha:[8]
23  Tipo:[ID] Valor:[fatorial] Linha:[8]
24  Tipo:[APAREN] Valor:[(] Linha:[8]
25  Tipo:[INTEIRO] Valor:[inteiro] Linha:[8]
26  Tipo:[DOISPONTOS] Valor:[:] Linha:[8]
27  Tipo:[ID] Valor:[n] Linha:[8]
28  Tipo:[FPAREN] Valor:[)] Linha:[8]
29  Tipo:[INTEIRO] Valor:[inteiro] Linha:[9]
30  Tipo:[DOISPONTOS] Valor:[:] Linha:[9]
31  Tipo:[ID] Valor:[fat] Linha:[9]
32  Tipo:[SE] Valor:[se] Linha:[10]
33  Tipo:[ID] Valor:[n] Linha:[10]
34  Tipo:[MAIOR] Valor:[>] Linha:[10]
35  Tipo:[NUMINTEIRO] Valor:[0] Linha:[10]
36  Tipo:[ENTAO] Valor:[então] Linha:[10]
```

**Figura 3. Resultado de uma análise**