

UNIVERSIDADE FEDERAL DE VIÇOSA
CAMPUS DE FLORESTAL
CIÊNCIA DA COMPUTAÇÃO

VITOR HUGO OLIVERIA SILVA(3049)



COMPILADORES TRABALHO PRÁTICO 1

BELO HORIZONTE, MG

4 de outubro de 2020

VITOR HUGO OLIVERIA SILVA(3049)

COMPILADORES TRABALHO PRÁTICO 1



Esse será o relatório do primeiro trabalho de compiladores, no qual usaremos o gerador de analisador léxico flex para criar um analisador de acordo com as especificações do trabalho.

Orientador: Prof. Dr. Daniel Mendez Barbosa

BELO HORIZONTE, MG

4 de outubro de 2020

Resumo

O trabalho foi feito com o intuito de se desenvolver um analisador léxico baseado em lex capaz de reconhecer números inteiros (positivos e negativos), números decimais(positivos e negativos), placa de carro, palavra, Telefone e nome próprio completo.

Palavras-chaves: lex, lex and yacc, regex, flex, analisador léxico, compiladores.



Abstract

The work was done in order to develop a lexical analyzer based on lex capable of recognizing whole numbers (positive and negative), decimal numbers (positive and negative), license plate, word, telephone and full first name.

Key-words: lex, lex and yacc, regex, flex, lexical analyzer, compilers.



Sumário

1	Introdução	5
2	Desenvolvimento	6
2.1	Declarações	6
2.2	Regras de traduções	7
2.3	Funções auxiliares	10
3	Código e testes	11
3.1	Código	11
3.1.1	prints.h	11
3.1.2	lex.l	11
3.2	Testes e Resultados	13
3.2.1	Caso Base	13
3.2.1.1	Entrada	13
3.2.1.2	Resultado	13
3.2.2	Caso I	14
3.2.2.1	Resultado	14
4	Conclusão	19

1 Introdução

O desenvolvimento deste trabalho consiste na criação de um analisador léxico capaz de realizar o reconhecimento de padrões como números inteiros (positivos e negativos), números decimais(positivos e negativos), placa de carro, palavras , Telefone e nome próprio completo. Para isso foi utilizado os conceitos de expressões regulares vistos nas aulas, pois através destas era possível gerar regras para os reconhecimentos dos padrões citados acima. Além disso foi utilizado o software livre Flex como gerador de analisador léxico que se faz como uma alternativa ao lex.



2 Desenvolvimento

Para a criação do projeto no Flex deveria se seguir uma formatação já estruturada onde deve-se realizar as declarações das expressões, define, macros, cabeçalhos e constantes. Após a declaração deve-se realizar as regras de traduções nas quais são os padrões de expressões que desejamos reconhecer. Por fim, a última estrutura são as funções auxiliares na qual permite criar funções que podem ser chamadas dentro das regras de traduções para o auxílio em uma determinada tarefa, como também permite a criação do main do programa que será gerado como analisador léxico.

2.1 Declarações



```
%{
#include "prints.h"
%}

%option noyywrap

/* definicoes regulares */

delim [ \t\n]
ws {delim}+
upperCase [A-Z]
lowerCase [a-z]
bothCase [A-Za-z]
digit [0-9]
positive \+?{digit}+
negative \-?[1-9]{digit}*
float \.{digit}+
decimal {positive}{float}
decimalNegative \- {digit}+{float}
licensePlate {upperCase}{3}\-{digit}{4}
word {bothCase}+
phone {digit}{4}\-{digit}{4}
name {upperCase}{1}{bothCase}*
fullName ({name}[ ]){2}{name}+([ ]{name})?

%%
```

Figura 1 – Declarações

A primeira declaração realizada foi o header prints criado por mim onde declaro duas funções de apoio às redPrint e BluePrint, ambas possuem a funcionalidade de realizar um print de acordo com as respectivas cores descritas em seus títulos. A bluePrint será usada nas funções auxiliares para printar os lexemas reconhecidos no sistema, por outro lado o redPrint será usado para printar os lexemas não reconhecidos pelo sistema. Segue abaixo as declarações realizadas:

- **delim:** Essa declaração define o reconhecimento de espaços, quebra de linha e tabulação;

- **ws:** Reconhece um ou mais precedentes de delim;
- **upperCase:** Reconhece todas as letras maiúsculas;
- **lowerCase:** Reconhece todas as letras minúsculas;
- **bothCase:** Reconhece tanto letras maiúsculas como letras minúsculas;
- **digit:** Reconhece todos os dígitos de 0 a 9;
- **positive:** Reconhece um ou mais dígitos precedidos ou não pelo operador +;
- **negative:** Reconhece um ou mais dígitos precedidos pelo operador -;
- **float:** Reconhece um ou mais dígitos precedidos pelo operador .;
- **decimalNegative:** Reconhece todos os padrões definidos em positive concatenado com os padrões definidos em float;
- **licensePlate:** Reconhece todos os padrões definidos em negative concatenado com os padrões definidos em float;
- **word:** Reconhece um ou mais bothCase;
- **phone:** Reconhece 3 padrões seguidos de upperCase seguido pelo operador - e 4 digits;
- **name:** Reconhece um upperCase seguido de 0 ou mais bothCase;
- **fullName:** Reconhecer 3 name seguidos ou não por um espaço concatenados com um ou zero name;

2.2 Regras de traduções

As regras de traduções é o núcleo do programa, pois através delas o flex saberá quais lexemas reconhecer e qual a ordem de prioridade estabelecer. Suas declarações são bem simples, no qual deve-se redigir as expressões regulares ou utilizar as expressões já definidas e assim definir as funções em C do que realizar com os lexemas reconhecidos. Na figura abaixo segue as regras de tradução:


```
%%  
  
{ws}      ;  
  
{positive} {bluePrint("Foi encontrado um numero inteiro positivo.",yytext);}   
{negative} {bluePrint("Foi encontrado um numero inteiro negativo.",yytext);}   
{decimal} {bluePrint("Foi encontrado um numero com parte decimal.",yytext);}   
{decimalNegative} {bluePrint("Foi encontrado um numero com parte decimal negativo.",yytext);}   
{licensePlate} {bluePrint("Foi encontrado uma placa.",yytext);}   
{word} {bluePrint("Foi encontrado uma palavra.",yytext);}   
{phone} {bluePrint("Foi encontrado um telefone.",yytext);}   
{fullName} {bluePrint("Foi encontrado um nome proprio.",yytext);}   
  . {redPrint(yytext);}   
  
%%
```

Figura 2 – Regras de traduções

Na imagem acima vemos a ordem de definição das regras e as ações estabelecidas em cada uma dessas.

- **{ws}**: Essa regra define que todos os espaços em brancos serão ignorados, e não será realizado nenhuma ação;
- **{positive}**: Essa regra define o reconhecimento de todas as os números naturais positivos, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{negative}**: Essa regra define o reconhecimento de todas as os números naturais negativos, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{decimal}**: Essa regra define o reconhecimento de todas as os decimais positivos, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{decimalNegative}**: Essa regra define o reconhecimento de todas as os números decimais negativos, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{licensePlate}**: Essa regra define o reconhecimento de todas as as placas de carros no padrão de três letras maiúsculas um traço e 4 dígitos, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{word}**: Essa regra define o reconhecimento de todas os lexemas de palavras compostas por letras minúsculas sem acentos e letras minúsculas sem acentos, após o

reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;

- **{phone}**: Essa regra define o reconhecimento de todos os celulares compostos por 4 dígitos seguidos por um traço e mais 4 dígitos, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{fullName}**: Essa regra define o reconhecimento de todas as os nomes próprios compostos por 3 ou mais palavras que não necessariamente termine em espaços e a primeira letra seja maiúscula, após o reconhecimento do mesmo sua ação realizará uma impressão na tela do usuário com o lexema achado;
- **{.}**: Essa regra define o reconhecimento de todos os padrões que não forma possível reconhecer no sistema, após o reconhecimento do mesmo sua ação realizará uma impressão de cor avermelhada na tela do usuário com o lexema de erro;

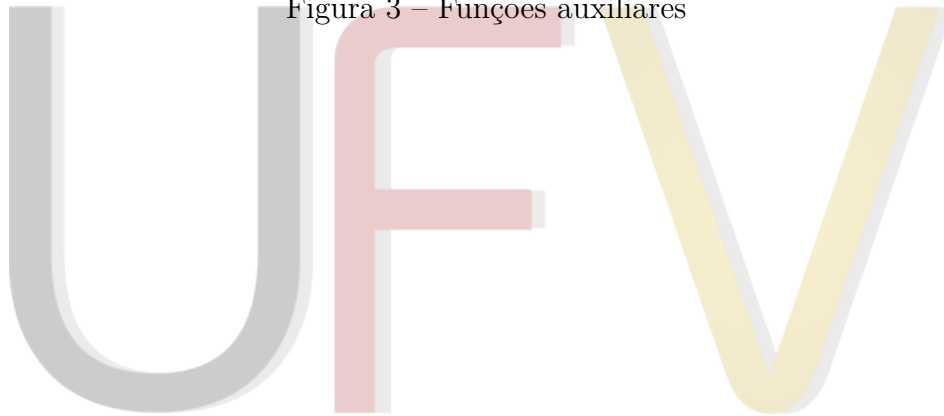
Algo interessante de ressaltar sobre uma versão antiga do código, onde para reconhecer fullname usava-se a seguinte expressão `(name[]?)3name?`, em determinadas situações essa expressão poderia servir tanto para word quanto fullname, nesse caso 3 palavras que possua uma letra maiúscula e um conjunto de letras minúsculas concatenadas, exemplo: VitorHugoOliveria. Todavia de acordo com a descrição de como deve se reconhecer cada padrão o lexema acima deve ser reconhecido com uma palavra, já que os nomes próprios devem ter espaços entre si, porém o programa conseguia distinguir bem cada uma desses padrões graças a ordem de declaração das regras de tradução.

2.3 Funções auxiliares

As funções auxiliares como o próprio nome já relata auxiliam na construção do analisador léxico, geralmente são usadas para fazer o controle da tabela de tokens. No caso do programa em análise a única função que foi declarada foi o main. Aqui vale uma outra ressalva que apesar da escolha de se declarar o main, poderia apenas passar o parametro -ll no momento de executar o flex que ele já se utilizaria de um main genérico que também serviria para o nosso caso. Segue as funções auxiliares utilizadas:

```
%%  
  
int main(void)  
{  
    /* Call the lexer, then quit. */  
    yylex();  
    return 0;  
}
```

Figura 3 – Funções auxiliares



3 Código e testes

3.1 Código

3.1.1 prints.h

```
void bluePrint(char* string, char* lexema);

void redPrint(char* lexema);

void bluePrint(char* string, char* lexema){
    printf("%s LEXEMA:\e[1;34m %s \e[0m \n",string,lexema);
}

void redPrint(char* lexema){
    printf("\e[1;31mNão foi possível encontrar o LEXEMA: %s \e[0m\n",lexema);
}
```

3.1.2 lex.l

```
%{

#include "prints.h"

%}

%option noyywrap

/* definicoes regulares */

delim [ \t\n]
ws {delim}+
upperCase [A-Z]
lowerCase [a-z]
bothCase [A-Za-z]
digit [0-9]
positive \+?{digit}+
```

```
negative \-[1-9]{digit}*
float \.{digit}+
decimal {positive}{float}
decimalNegative \-{digit}+{float}
licensePlate {upperCase}{3}\-{digit}{4}
word {bothCase}+
phone {digit}{4}\-{digit}{4}
name {upperCase}{1}{bothCase}*
fullName ({name}[ ]){2}{name}+([ ]{name})?

%%

{ws} ;

{positive} {bluePrint("Foi encontrado um numero inteiro positivo.",yytext);}

{negative} {bluePrint("Foi encontrado um numero inteiro negativo.",yytext);}

{decimal} {bluePrint("Foi encontrado um numero com parte decimal.",yytext);}

{decimalNegative} {bluePrint("Foi encontrado um numero com parte decimal negativo."

{licensePlate} {bluePrint("Foi encontrado uma placa.",yytext);}

{word} {bluePrint("Foi encontrado uma palavra.",yytext);}

{phone} {bluePrint("Foi encontrado um telefone.",yytext);}

{fullName} {bluePrint("Foi encontrado um nome proprio.",yytext);}

. {redPrint(yytext);}

%%

int main(void)
{
    /* Call the lexer, then quit. */
    yylex();
}
```

```
    return 0;  
}
```

3.2 Testes e Resultados

3.2.1 Caso Base

3.2.1.1 Entrada

875878 -3355456 abc5464 abc-5464 ABC-5464 453-2345 9486-0847 Daniel Mendes Barbosa 32.345 Palavra Qualquer 3567-3224 Daniel Mendes Barbosa Daniel Mendes Barbosa Menezes200

3.2.1.2 Resultado

Foi encontrado um numero inteiro positivo. LEXEMA: 875878
Foi encontrado um numero inteiro negativo. LEXEMA: -3355456
Foi encontrado uma palavra. LEXEMA: abc
Foi encontrado um numero inteiro positivo. LEXEMA: 5464
Foi encontrado uma palavra. LEXEMA: abc
Foi encontrado um numero inteiro negativo. LEXEMA: -5464
Foi encontrado uma placa. LEXEMA: ABC-5464
Foi encontrado um numero inteiro positivo. LEXEMA: 453
Foi encontrado um numero inteiro negativo. LEXEMA: -2345
Foi encontrado um telefone. LEXEMA: 9486-0847
Foi encontrado um nome proprio. LEXEMA: Daniel Mendes Barbosa
Foi encontrado um numero com parte decimal. LEXEMA: 32.345
Foi encontrado uma palavra. LEXEMA: Palavra
Foi encontrado uma palavra. LEXEMA: Qualquer
Foi encontrado um telefone. LEXEMA: 3567-3224
Foi encontrado um nome proprio. LEXEMA: Daniel Mendes Barbosa Daniel
Foi encontrado uma palavra. LEXEMA: Mendes
Foi encontrado uma palavra. LEXEMA: Barbosa
Foi encontrado uma palavra. LEXEMA: Menezes
Foi encontrado um numero inteiro positivo. LEXEMA: 200

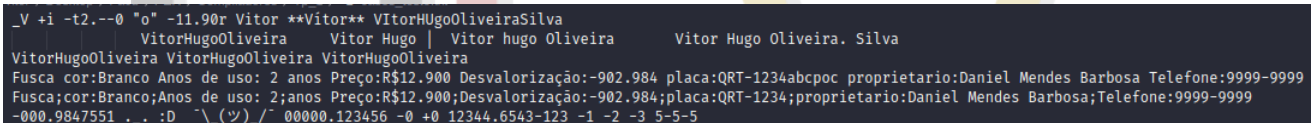
3.2.2 Caso I

```

_V +i -t2.--0 "o" -11.90r Vitor **Vitor** VitorHugoOliveiraSilva
      VitorHugoOliveira      Vitor Hugo | Vitor hugo Oliveira
      Vitor Hugo Oliveira. Silva
      VitorHugoOliveira VitorHugoOliveira VitorHugoOliveira
Fusca cor:Branco Anos de uso: 2 anos Preço:R$12.900 Desvalorização:-902.984
placa:QRT-1234abcpoc proprietario:Daniel Mendes Barbosa Telefone:9999-9999
Fusca;cor:Branco;Anos de uso: 2;anos Preço:R$12.900;Desvalorização:-902.984;
      placa:QRT-1234;proprietario:Daniel Mendes Barbosa;Telefone:9999-9999
-000.9847551 ._. :D \_()_/ 00000.123456 -0 +0 12344.6543-123 -1 -2 -3 5-5-5

```

Devido a formatação do texto em PDF não foi possível inserir o texto exatamente como foi teste, devido a isso abaixo segue uma imagem do texto original:



```

_V +i -t2.--0 "o" -11.90r Vitor **Vitor** VitorHugoOliveiraSilva
      VitorHugoOliveira      Vitor Hugo | Vitor hugo Oliveira      Vitor Hugo Oliveira. Silva
      VitorHugoOliveira VitorHugoOliveira VitorHugoOliveira
Fusca cor:Branco Anos de uso: 2 anos Preço:R$12.900 Desvalorização:-902.984 placa:QRT-1234abcpoc proprietario:Daniel Mendes Barbosa Telefone:9999-9999
Fusca;cor:Branco;Anos de uso: 2;anos Preço:R$12.900;Desvalorização:-902.984;placa:QRT-1234;proprietario:Daniel Mendes Barbosa;Telefone:9999-9999
-000.9847551 ._. :D \_()_/ 00000.123456 -0 +0 12344.6543-123 -1 -2 -3 5-5-5

```

Figura 4 – Caso de teste I

3.2.2.1 Resultado

```

Não foi possível encontrar o LEXEMA: _
Foi encontrado uma palavra. LEXEMA: V
Não foi possível encontrar o LEXEMA: +
Foi encontrado uma palavra. LEXEMA: i
Não foi possível encontrar o LEXEMA: -
Foi encontrado uma palavra. LEXEMA: t
Foi encontrado um numero inteiro positivo. LEXEMA: 2
Não foi possível encontrar o LEXEMA: .
Não foi possível encontrar o LEXEMA: -
Não foi possível encontrar o LEXEMA: -
Foi encontrado um numero inteiro positivo. LEXEMA: 0
Não foi possível encontrar o LEXEMA: "
Foi encontrado uma palavra. LEXEMA: o
Não foi possível encontrar o LEXEMA: "
Foi encontrado um numero com parte decimal negativo. LEXEMA: -11.90
Foi encontrado uma palavra. LEXEMA: r

```

Foi encontrado uma palavra. LEXEMA: Vitor
Não foi possível encontrar o LEXEMA: *
Não foi possível encontrar o LEXEMA: *
Foi encontrado uma palavra. LEXEMA: V
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Foi encontrado uma palavra. LEXEMA: tor
Não foi possível encontrar o LEXEMA: *
Não foi possível encontrar o LEXEMA: *
Foi encontrado uma palavra. LEXEMA: VitorHUGoOliveiraSilva
Foi encontrado uma palavra. LEXEMA: VitorHugoOliveira
Foi encontrado uma palavra. LEXEMA: Vitor
Foi encontrado uma palavra. LEXEMA: Hugo
Não foi possível encontrar o LEXEMA: |
Foi encontrado uma palavra. LEXEMA: Vitor
Foi encontrado uma palavra. LEXEMA: hugo
Foi encontrado uma palavra. LEXEMA: Oliveira
Foi encontrado um nome proprio. LEXEMA: Vitor Hugo Oliveira
Não foi possível encontrar o LEXEMA: .
Foi encontrado uma palavra. LEXEMA: Silva
Foi encontrado um nome proprio. LEXEMA: VitorHugoOliveira VitorHugoOliveira
VitorHugoOliveira
Foi encontrado uma palavra. LEXEMA: Fusca
Foi encontrado uma palavra. LEXEMA: cor
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma palavra. LEXEMA: Branco
Foi encontrado uma palavra. LEXEMA: Anos
Foi encontrado uma palavra. LEXEMA: de
Foi encontrado uma palavra. LEXEMA: uso
Não foi possível encontrar o LEXEMA: :
Foi encontrado um numero inteiro positivo. LEXEMA: 2
Foi encontrado uma palavra. LEXEMA: anos
Foi encontrado uma palavra. LEXEMA: Pre
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Foi encontrado uma palavra. LEXEMA: o
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma palavra. LEXEMA: R
Não foi possível encontrar o LEXEMA: \$
Foi encontrado um numero com parte decimal. LEXEMA: 12.900

Foi encontrado uma palavra. LEXEMA: Desvaloriza
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Foi encontrado uma palavra. LEXEMA: o
Não foi possível encontrar o LEXEMA: :
Foi encontrado um numero com parte decimal negativo. LEXEMA: -902.984
Foi encontrado uma palavra. LEXEMA: placa
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma placa. LEXEMA: QRT-1234
Foi encontrado uma palavra. LEXEMA: abcpoc
Foi encontrado uma palavra. LEXEMA: proprietario
Não foi possível encontrar o LEXEMA: :
Foi encontrado um nome proprio. LEXEMA: Daniel Mendes Barbosa Telefone
Não foi possível encontrar o LEXEMA: :
Foi encontrado um telefone. LEXEMA: 9999-9999
Foi encontrado uma palavra. LEXEMA: Fusca
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: cor
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma palavra. LEXEMA: Branco
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: Anos
Foi encontrado uma palavra. LEXEMA: de
Foi encontrado uma palavra. LEXEMA: uso
Não foi possível encontrar o LEXEMA: :
Foi encontrado um numero inteiro positivo. LEXEMA: 2
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: anos
Foi encontrado uma palavra. LEXEMA: Pre
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Foi encontrado uma palavra. LEXEMA: o
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma palavra. LEXEMA: R
Não foi possível encontrar o LEXEMA: \$
Foi encontrado um numero com parte decimal. LEXEMA: 12.900
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: Desvaloriza

Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Foi encontrado uma palavra. LEXEMA: o
Não foi possível encontrar o LEXEMA: :
Foi encontrado um numero com parte decimal negativo. LEXEMA: -902.984
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: placa
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma placa. LEXEMA: QRT-1234
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: proprietario
Não foi possível encontrar o LEXEMA: :
Foi encontrado um nome proprio. LEXEMA: Daniel Mendes Barbosa
Não foi possível encontrar o LEXEMA: ;
Foi encontrado uma palavra. LEXEMA: Telefone
Não foi possível encontrar o LEXEMA: :
Foi encontrado um telefone. LEXEMA: 9999-9999
Foi encontrado um numero com parte decimal negativo. LEXEMA: -000.9847551
Não foi possível encontrar o LEXEMA: .
Não foi possível encontrar o LEXEMA: _
Não foi possível encontrar o LEXEMA: .
Não foi possível encontrar o LEXEMA: :
Foi encontrado uma palavra. LEXEMA: D
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA: \
Não foi possível encontrar o LEXEMA: _
Não foi possível encontrar o LEXEMA: (
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:)
Não foi possível encontrar o LEXEMA: _
Não foi possível encontrar o LEXEMA: /
Não foi possível encontrar o LEXEMA:
Não foi possível encontrar o LEXEMA:
Foi encontrado um numero com parte decimal. LEXEMA: 00000.123456
Não foi possível encontrar o LEXEMA: -

Foi encontrado um numero inteiro positivo. LEXEMA: 0
Foi encontrado um numero inteiro positivo. LEXEMA: +0
Foi encontrado um numero com parte decimal. LEXEMA: 12344.6543
Foi encontrado um numero inteiro negativo. LEXEMA: -123
Foi encontrado um numero inteiro negativo. LEXEMA: -1
Foi encontrado um numero inteiro negativo. LEXEMA: -2
Foi encontrado um numero inteiro negativo. LEXEMA: -3
Foi encontrado um numero inteiro positivo. LEXEMA: 5
Foi encontrado um numero inteiro negativo. LEXEMA: -5
Foi encontrado um numero inteiro negativo. LEXEMA: -5

Os espaços vazios são caracteres especiais que não foi possível mostrar.



4 Conclusão

Apesar da densa complexidade vista nas aulas para construção de um compilador, quando quebrado em partes como no caso do analisador léxico, é perceptível a facilidade de se tratar e construir essas pequenas partes, além de que facilita a manutenção e otimização do código.

A utilização do flex foi outra abordagem interessante de se implementar, pois complementou a teoria vista em sala, mostrando na prática como é possível usar geradores de analisadores léxicos que com apenas as expressões regulares já gera tokens que podem ser passados para análise semântica.

Por fim com poucas linhas de códigos foi possível construir um analisador capaz de reconhecer os padrões propostos de formas simples e eficaz.

