Relatório - Analisador Léxico

16 de Abril de 2023

Nome Completo	**Matrícula**
Lívia Pereira Ozório	201835011
Patrick Canto de Carvalho	201935026

Sumário

1 Introdução

Este trabalho objetiva criar um analisador léxico, que é o processo de converter uma sequência de caracteres em uma sequência de tokens, para a linguagem lang.

O analisador léxico foi desenvolvido na linguagem Java com uso da ferramenta Jflex. Para tal, foi necessário especificar a linguagem utilizando expressões regulares.

A estrutura do projeto consiste de duas partes: o analisador léxico gerado pelo JFlex e uma classe principal responsável por receber os parâmetros da linha de comando, abrir o arquivo desejado e solicitar a leitura dos tokens do arquivo pelo analisador.

Foi usada uma classe Token como representação interna, contendo informações básicas sobre cada token lido: o nome, o lexema lido e a sua localização no arquivo de entrada.

2 Regras da gramatica

A gramatica utilizada para a construção dos tokens utiliza-se de algumas categorias léxicas sendo elas:

- identificador = são palavras começadas com uma letra minuscula seguida de outras letras ou do caractere especial "_";
- tipo = são palavras começadas com uma letra maiúscula seguida de outras letras ou do caractere especial "_";
- inteiro = são palavras formadas por uma sequencia de um ou mais dígitos;
- ponto flutuante = são palavras formadas por uma sequencia de zero ou mais dígitos seguidos por um ponto e uma sequencia de um ou mais dígitos;
- caractere = são palavras formadas por um caractere entre aspas simples. Aceita caracteres especiais como '\r', \n, '\t', '\\', '\".
- comentário de linha = são palavras começadas com – seguidas por vários caracteres e terminadas com \r ou \n ou \r\n
- comentário por bloco = são palavras começadas por {- seguido de caracteres e terminadas com -}

As categorias de comentários, tanto o de linha quanto o de bloco, não geram tokens, são apenas ignorados apesar de lidos.

Na linguagem existem palavras reservadas, cujas regras devem ser inseridas antes das de categoria léxica, já que, se vierem depois, podem entram em outra categoria. Essas palavras são:

- true;
- false;
- null;
- data;
- Int;
- Float;

- Bool;
- Char;
- if;
- else;
- iterate;
- read;
- print;
- new.

E, por ultimo, são aceitos pela linguagem alguns símbolos de operadores e separadores, sendo eles:

- =
- ;
- *
- +
- -
- /
- %
- {
- }
- (
-)
- [
-]
- <
- >
- :
- ::
- .
- ,
- ==
- !=
- &&
- !

3 Tokens

A seguir os tokens que podem ser capturados pelo analisador léxico e as suas especificações:

TOKEN	**ESPECIFICAÇÃO**
EQ	=
SEMI	;
TIMES	*
PLUS	+
SUBTRACTION	-
DIVISION	/
MOD	%
BRACES_OPEN	{
BRACES_CLOSE	}
PARENTHESES_OPEN	(
PARENTHESES_CLOSE	)
BRACKETS_OPEN	[
BRACKETS_CLOSE	]
COLON	:
TWO_COLON	::
PERIOD	.
COMMA	,
MORE_THAN	>
LESS_THAN	<
EQ_LOGIC	==
DIFFERENT	!=
AND	&&
NEGATION	!
TRUE	true
FALSE	false
NULL	null
DATA	data
INT	Int
CHAR	Char
BOOL	Bool
FLOAT	Float
IF	if
ElSE	else
ITERATE	iterate
READ	read
PRINT	print
NEW	new
ID_VALUE	[a-z] [[a-z] | [A-Z] | [0-9] | _]*
INT_VALUE	[a-z] [[a-z] | [A-Z] | [0-9] | _]*
TYPE	[A-Z] [[a-z] | [A-Z] | [0-9] | _]*
CHAR_VALUE	'(.)' | '\r'| '\n' | '\t' | '\\' | '\"

Tabela 1: Tokenização

4 Saída do analisador léxico

Ao passar um arquivo de entrada pelo analisador léxico, é retornada uma string para cada palavra encontrada com a seguinte estrutura:

TOKEN(LINHA,COLUNA) "LEXEMA" <VALOR CASO SEJA UM NUMERO>

O token retornado identifica a palavra ao compilador, a linha e a coluna foram colocados para, posteriormente, quando for necessário gerar os erros de sintaxe e semântica, ser possível indicar onde está o erro. Enquanto o lexema e o valor serão importantes no momento de executar os comandos.

5 Como compilar

Para compilar, executar o arquivo compilar.sh, presente na pasta do projeto. Alternativamente, executar os comandos:

java -jar jflex-full-1.8.2.jar Lexer.jflex
javac Lexer.java
javac Token.java
javac Teste.java
jar cmvf MANIFEST.MF Teste.jar Lexer.class Token.class Teste.class TOKEN_TYPE.class

E, para executar o analisador:

java -jar Teste.jar <nome do arquivo a ser analisado>