UNIVERSIDADE FEDERAL DE RORAIMA
CENTRO DE CIÊNCIA DA COMPUTAÇÃO
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RELATÓRIO ANALISADOR LÉXICO

BOA VISTA – RR
2023.1

GUILHERME LUCAS PEREIRA BERNARDO

RELATÓRIO ANALISADOR LÉXICO

Relatório de trabalho apresentado ao Centro de Ciência da Computação do Curso de Ciência da Computação da Universidade Federal de Roraima, como requisito parcial para obtenção de notas da disciplina **CONSTRUÇÃO DE COMPILADORES** – DCC605 —, sob orientação do professor Dr. Luciano F. Silva.

BOA VISTA – RR

2023.1

SUMÁRIO

1. INTRODUÇÃO

Este trabalho tem como objetivo apresentar as conclusões obtidas pelo autor acerca do desenvolvimento de um separador de lexemas, como requisitado na aula do dia 16 de março, onde foi apresentado a tarefa. Porém, conforme foi-se estudando acerca do assunto, o autor então decidiu expandir o assunto e conceituar um analisador léxico, próxima fase do processo de compilação de um programa.

2. RELATÓRIO

a. Concepção

O projeto de analisador léxico apresentado neste relatório se dá, baseado no projeto do professor Daniel Lucrédio da UFSCAR onde ele cria um analisador léxico com uma tabela de símbolos padronizada, a seguir é mostrada a informações detalhadas dos símbolos.

O trabalho se deu pela linguagem Java, ao qual tanto o autor do relatório quanto Lucrédio tem maior proficiência no desenvolvimento de aplicações. Aliados da IDE Netbeans, o programa é case-sensitive em relação a suas palavras reservadas, além de ser baseado em uma linguagem similar à um pseudocódigo baseado em portugol.

Contendo dois buffers de caracteres que leem um arquivo de texto ao mesmo tempo, o analisador léxico é capaz de identificar o início de diferentes lexemas e determinar seus tamanhos e tipos com base em uma operação simples de ifs envoltas de um loop que só termina com uma condição de parada específica.

b. Tabela de Símbolos

expressão regular	token	categoria do token
Dígito = [0-9]		
Letra = [a-zA-Z_]		
NumInt = D+	NumInt	Número inteiro
NumReal = D+\.D+	NumReal	Número real
Var = L(L|D)*	Var	Identificador
TEXT = ".*"	Cadeia	Constantes de Texto
PCInteiro = INT	PCInteiro	Palavra reservada
PCReal = REAL	PCReal	Palavra reservada
BOOLEAN = BOOLEAN	BOOLEAN	Palavra reservada
PCSe = SE	PCSe	Palavra reservada

PCEntao = ENTAO	PCEntao	Palavra reservada
PCEnquanto = ENQUANTO	PCEnquanto	Palavra reservada
PCLer = LER	PCLer	Palavra reservada
PCAlgoritmo = ALGORITMO	PCAlgoritmo	Palavra reservada
PCDeclaracoes = DECLARACOES	PCDeclaracoes	Palavra reservada
PCA	PCAlgoritmo	Palavra reservada
PCImprimir = IMPRIMIR	PCImprimir	Palavra reservada
PCInicio = INICIO	PCInicio	Palavra reservada
PCFim = FIM	PCFim	Palavra reservada
RETURN = return	RETURN	Palavra reservada
% = //.*		Comentário
PCAtribuir = '='	PCATRIBUIR	Operador de Atribuição
(Delimitador) Delim = :	Delim	Simbolo especial
+	OpAritSoma	Operador aritmético
-	OpAritSub	Operador aritmético
*	OpAritMult	Operador aritmético
/	OpAritDiv	Operador aritmético
E	OpBoolE	Operador lógico
OU	OpBoolOu	Operador lógico
<	OpRelMenor	Operador de comparação
<=	OpRelMenorIgual	Operador de comparação
>	OpRelMaior	Operador de comparação
>=	OpRelMaiorIgual	Operador de comparação
==	OpRelIgual	Operador de comparação
!=	OpRelDif	Operador de comparação
(	AbrePar	Símbolo especial
)	FechaPar	Simbolo especial
Fim	Fim	Sinal de Parada

\. significa literalmente .		
* significa zero ou mais		
. sozinho significa que pode ser qualquer caracter		

c. Testes

Foram feitos dois testes, o primeiro mais simples, com apenas duas linhas de instrução e o segundo mais completo, implementando em pseudocódigo um algoritmo de fatorial:

i. primeiro teste:

```
x = y + 5.0
real z
FIM
```

A saída do programa será:

```
<Var,x>
<OpRelIgual,=>
<Var,y>
<OpAritSoma,+>
<NumReal,5.0>
<PCReal,REAL>
<Var,z>
<PCFim,FIM>
```

ii. segundo teste:

```
:DECLARACOES
argumento:INT
fatorial:INT

:ALGORITMO
% Calcula o fatorial de um número inteiro
LER argumento
ATRIBUIR argumento A fatorial
SE argumento = 0 ENTAO ATRIBUIR 1 A fatorial
```

```
ENQUANTO argumento < 1000

   INICIO
          ATRIBUIR  fatorial  *  (argumento  -  1)  A
fatorial
      ATRIBUIR argumento - 1 A argumento
   FIM
IMPRIMIR fatorial
```

A saída do programa será:

```
<Delim,:>
<PCDeclaracoes,DECLARACOES>
<Var,argumento>
<Delim,:>
<PCInteiro,INT>
<Var,fatorial>
<Delim,:>
<PCInteiro,INT>
<Delim,:>
<PCAlgoritmo,ALGORITMO>
<PCLer,LER>
<Var,argumento>
<PCAtribuir,ATRIBUIR>
<Var,argumento>
<PCA,A>
<Var,fatorial>
<PCSe,SE>
<Var,argumento>
<OpRelIgual,=>
<NumInt,0>
<PCEntao,ENTAO>
<PCAtribuir,ATRIBUIR>
<NumInt,1>
<PCA,A>
<Var,fatorial>
<PCEnquanto,ENQUANTO>
<Var,argumento>
<OpRelMenor,<>
<NumInt,1000>
<PCInicio,INICIO>
<PCAtribuir,ATRIBUIR>
<Var,fatorial>
<OpAritMult,*>
<AbrePar,(>
<Var,argumento>
<OpAritSub,->
<NumInt,1>
<FechaPar,)>
<PCA,A>
<Var,fatorial>
<PCAtribuir,ATRIBUIR>
<Var,argumento>
<OpAritSub,->
<NumInt,1>
<PCA,A>
<Var,argumento>
<PCFim,FIM>
<PCImprimir,IMPRIMIR>
<Var,fatorial>
```

3. CONSIDERAÇÕES FINAIS

Considerando as informações adquiridas pelas aulas e pela pesquisa necessária para desenvolver este analisador léxico, conclui-se que este é um exemplo perfeito de como programas de computadores podem se tornar complexos e que o desenvolvimento dos mesmos deve ser efetuado com cuidado e preparo.

No caso de um compilador, em sua primeira fase, é crucial termos definido sua gramática com a qual será trabalhada nas análises do mesmo, sem um estudo aprofundado não teremos como fabricar um programa com a qualidade necessária para futuro uso.

Para adquirir mais informações acerca do projeto, pode-se entrar no link GuilhermeBn198/DCC605-Compilers para poder visualizar o projeto e o código.

4. REFERÊNCIAS

AHO, Alfred V; SETHI, Ravi; ULMAN, Jeffrey D. **Compiladores - Princípios Técnicas e Ferramentas. 1995.** Editora Pearson.

Construção de Compiladores - Aula 03: Tópico 02 - Análise Léxica - Parte 2. Daniel Lucrédio(Criador). Local: youtube, 18 de fev. de 2020.

LUCRÉDIO, Daniel. Cursocompiladores. Disponível em: dlucredio/cursocompiladores (github.com). Acesso em: 11 de mar. de 2023