Departamento de Computação (DECOM)
Universidade Federal de Ouro Preto (UFOP)
Construção de Compiladores I (BCC328)
Palloma Stéphanne Silva Brito
23 de setembro de 2018

1. Resumo

O objetivo deste trabalho é desenvolver um analisador léxico para uma pequena linguagem de programação. Trata-se de uma linguagem apresentada por Torben Ægidius Mogensen no capítulo 4 de seu livro *Introduction to Compiler Design.*

2. A linguagem Torben

A seguir, é apresentada uma gramática livre de contexto para a linguagem de programação proposta por Torben. Chamaremos de linguagem Torben. Nesta linguagem, um programa é uma lista de declarações de função. Cada função declara seu tipo de resultado e os tipos e nomes de seus argumentos. Os tipos são int (inteiro) e bool (booleano). A comparação é definida para os booleanos e para os inteiros, mas a adição é definida apenas para os inteiros.

Program → *Funs*	**PROGRAMA**
Funs → *Fun*	
Funs → *Fun Funs*	
Fun → *TypeId (TypeIds) = Exp*	**FUNÇÃO**
TypeId → *int* id	**TIPO INTEIRO**
TypeId → *bool* id	**TIPO BOOLEANO**
TypeIds → *TypeId*	
TypeIds → *TypeId , TypeIds*	
Exp → *num*	
Exp → *id*	
Exp → *Exp* + *Exp*	**OPERAÇÃO DE SOMA**
Exp → *Exp* < *Exp*	**OPERAÇÃO RELACIONAL (MENOR QUE)**
Exp → *if* Exp *then* Exp *else* Exp	**EXPRESSÃO CONDICIONAL**
Exp → id *(Exps)*	**CHAMADA DE FUNÇÃO**
Exp → *let* id = Exp *in* Exp	**EXPRESSÃO DE DECLARAÇÃO**
Exps → *Exp*	
Exps → *Exp, Exps*	

3. Análise Léxica

Os números inteiros são compostos por uma sequência de um ou mais dígitos decimais. Já os literais booleanos assumem os valores “true” ou “false”. Durante a análise léxica, os caracteres brancos são ignorados. Identificadores são sequências de letras maiúsculas ou minúsculas, dígitos decimais e sublinhados (_) e sempre se iniciam com uma letra. O analisador diferencia letras maiúsculas de minúsculas em um identificador.

4. Testes

Dentro do diretório *examples* existem alguns programas na linguagem torben que podem ser testados pelo analisador léxico. Abaixo estão outros exemplos.

Input	Output
if 5 < 6 then 8 + 9 then 7 + 6	2:1-2:3 IF 2:4-2:5 LITINT(5) 2:5-2:6 LT 2:7-2:8 LITINT(6) 2:9-2:13 THEN 2:14-2:15 LITINT(8) 2:16-2:17 PLUS 2:18-2:19 LITINT(9) 2:20-2:24 THEN 2:25-2:26 LITINT(7) 2:27-2:28 PLUS 2:29-2:30 LITINT(6)
Compiladores BCC328	3:1-3:13 ID(Compiladores) 3:14-3:20 ID(BCC328)

Obs.: o projeto do analisador léxico foi disponibilizado pelo professor José Romildo e pode ser acessado através do link: < https://github.com/romildo/torben-java/tree/5e04d18234f519520690cf977a37ceeb27f2913b >