Universidade Tiradentes
Ciência da Computação

Thiago Estombelo Llapa
Ramon Costa da Guia
Luiz Felipe de Araujo Menezes

PROJETO COMPILADORES

LINGUAGEM DE DIÁLOGO PARA CHATBOTS

Aracaju - SE
2025

Thiago Estombelo Llapa
Ramon Costa da Guia
Luiz Felipe de Araujo Menezes

PROJETO COMPILADORES
LINGUAGEM DE DIÁLOGO PARA CHATBOTS

Atividade sobre Linguagem de Diálogo para Chatbots, apresentada como requisito parcial para avaliação da disciplina Compiladores, ministrada pela Prof.ª Layse Santos Souza, no 2º semestre de 2025.

Aracaju - SE
2025

Sumário

1 INTRODUÇÃO

A etapa inicial do desenvolvimento de um compilador é responsável por transformar a entrada textual de um programa em uma sequência estruturada de símbolos significativos. Essa fase, denominada análise léxica, realiza a leitura do código-fonte caractere por caractere com o objetivo de identificar padrões, reconhecer tokens e eliminar elementos irrelevantes, como espaços em branco e comentários. Segundo Aho, Sethi e Ullman (2007), essa etapa atua como a base para todo o processo de compilação, garantindo que o analisador sintático receba dados coerentes e organizados.

O presente trabalho descreve o desenvolvimento do analisador léxico referente à Etapa 01 do projeto da disciplina de Compiladores, realizado pelo Grupo 02. A implementação foi desenvolvida em Python, utilizando expressões regulares e autômatos, técnicas amplamente citadas na literatura de construção de compiladores (WIRTH, 1996).

2 JUSTIFICATIVA

A criação de um analisador léxico próprio permite compreender profundamente o funcionamento estrutural de um compilador, tornando evidente como linguagens de programação são decompostas e interpretadas pelas máquinas. Essa etapa é essencial, pois erros não tratados durante a análise léxica podem ser propagados para a análise sintática e semântica, comprometendo todo o processo (AHO; SETHI; ULLMAN, 2007).

Optou-se por implementar o analisador manualmente, sem uso de ferramentas automatizadas como Lex ou Flex, devido ao caráter didático da disciplina. Tal escolha possibilita a prática direta com expressões regulares, autômatos e tabelas de transição, o que fortalece o entendimento dos mecanismos de reconhecimento de padrões, como afirma Louden (2003).

3 OBJETIVOS

3.1 OBJETIVO GERAL

Desenvolver um analisador léxico funcional capaz de identificar e classificar tokens de uma linguagem pré-definida, estruturando-os para uso nas próximas etapas da construção de um compilador.

3.2 OBJETIVOS ESPECÍFICOS

1. Definir e documentar o conjunto de tokens da linguagem, estabelecendo suas expressões regulares;
2. Construir um autômato capaz de reconhecer os padrões definidos;
3. Implementar, em Python, um módulo de varredura eficiente para leitura e classificação dos tokens;
4. Organizar o projeto de forma modular;
5. Validar o funcionamento do analisador por meio de casos de teste documentados;

4 METODOLOGIA

A metodologia adotada iniciou-se com a especificação dos tokens da linguagem-alvo, abrangendo identificadores, números, operadores, delimitadores e palavras reservadas, conforme propõe Aho et al. (2007). Em seguida, foram elaboradas expressões regulares responsáveis por descrever cada categoria léxica, tomando como base princípios de reconhecimento formal de padrões (WIRTH, 1996).

Posteriormente, iniciou-se a implementação do analisador léxico em Python. O código foi estruturado em módulos que realizam a leitura do arquivo-fonte, a identificação de padrões e a classificação dos tokens encontrados. Casos de teste foram utilizados para verificar o reconhecimento correto dos elementos da linguagem, incluindo situações específicas que exigiram ajustes manuais, como a diferenciação entre operadores compostos e simples.

5 RESULTADOS E DISCUSSÕES

A implementação produzida demonstrou capacidade de reconhecer corretamente os tokens definidos na linguagem proposta para a Etapa 01. Os testes realizados validaram o reconhecimento de identificadores, números inteiros, operadores aritméticos, operadores relacionais e delimitadores. Além disso, observou-se que a modularização do código facilitou o processo de depuração e compreensão estrutural do analisador.

O uso de expressões regulares mostrou-se eficiente para identificar padrões simples, corroborando com os princípios descritos por Louden (2003). Entretanto, determinados casos exigiram refinamento lógico, evidenciando a importância do controle manual em implementações didáticas, quando comparadas a soluções automatizadas.

6 CONSIDERAÇÕES FINAIS

A realização da Etapa 01 permitiu ao grupo compreender, na prática, os mecanismos fundamentais envolvidos na análise léxica. A implementação de um analisador próprio proporcionou maior entendimento sobre expressões regulares, autômatos e estruturas formais de reconhecimento de padrões, conforme discutido pelos autores clássicos da área (AHO; SETHI; ULLMAN, 2007; WIRTH, 1996).

O código desenvolvido mostrou-se funcional e suficientemente robusto para servir como ponto de partida para as próximas fases do projeto. A partir da base léxica estruturada, será possível avançar para a análise sintática e, posteriormente, para a análise semântica, completando o desenvolvimento de um compilador simples.

7 REFERÊNCIAS

Livro clássico ("Dragon Book"): AHO, Alfred V.; SETHI, Ravi; ULLMAN, Jeffrey D. Compilers: Principles, Techniques, and Tools. 2. ed. Boston: Addison-Wesley, 2007.

Construção de Compiladores (livro didático): LOUDEN, Kenneth C. Compiler Construction: Principles and Practice. Boston: PWS Publishing Company, 2003.

Autômatos e gramáticas (base teórica): HOPCROFT, John E.; MOTWANI, Rajeev; ULLMAN, Jeffrey D. Introduction to Automata Theory, Languages, and Computation. 3. ed. Boston: Addison-Wesley, 2006.

Fundamentos de construção de compiladores (alternativa mais curta): WIRTH, Niklaus. Compiler Construction. Addison-Wesley, 1996.

Linguagens formais e autômatos (apoio para teoria léxica): LEWIS, Harry R.; PAPADIMITRIOU, Christos H. Elements of the Theory of Computation. 2. ed. Upper Saddle River: Prentice Hall, 1997.