GUSTAVO BOTEZINI, NYCOLLAS WILDNEY DOS SANTOS BARBOSA DA FONSECA

RECONHECEDOR LÉXICO NA LÍNGUA FICTÍCIA DOVAHZUL

CHAPECÓ
2025

RESUMO:

Este trabalho tem como objetivo a implementação de um reconhecedor léxico da língua fictícia Dohvazul capaz de identificar palavras-chave de uma linguagem simples. O trabalho aborda a definição de tokens, construção de gramáticas e validação da implementação.

INTRODUÇÃO:

Um analisador léxico é um programa que implementa um autômato finito, reconhecendo (ou não) strings como símbolos válidos de uma linguagem. Esse em específico utiliza a língua fictícia *Dohvazul* presente na franquia de jogos The Elder Scrolls da desenvolvedora *Bethesda*.

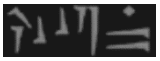
O propósito deste trabalho é exclusivamente acadêmico, (mesmo sendo cômico ter uma “linguagem de programação meme” com esse estilo). Foram escolhidas algumas das palavras mais conhecidas do Dohvazul para atuarem como palavras-chave do analisador, possibilitando a aplicação prática dos conceitos de compiladores.

REFERENCIAL TEÓRICO:

Os compiladores, responsáveis por traduzir programas escritos em linguagens de alto nível para um formato compreensível pelo computador, são compostos por diversas etapas de processamento. A primeira delas é a análise léxica, cujo objetivo principal é identificar os lexemas presentes no código-fonte e classificá-los em unidades denominadas tokens. Esse processo consiste na leitura sequencial dos caracteres do programa, agrupando-os em padrões reconhecíveis, como palavras reservadas, identificadores e operadores. O analisador léxico, além de gerar a sequência de tokens que servirá de entrada para a análise sintática, também pode interagir com a tabela de símbolos, registrando informações relevantes a respeito de identificadores e constantes. Dessa forma, desempenha papel fundamental na organização e estruturação inicial dos elementos de um programa, possibilitando a correta interpretação nas etapas subsequentes da compilação.

https://johnidm.gitbooks.io/compiladores-para-humanos/content/part1/lexical-analysis.html

Segue abaixo os tokens e seus significados:

PALAVRA ORIGINAL	SIGNIFICADO	REPRESENTAÇÃO	ORIGINAL
KO	em	>	
KEL	Elder Scroll	<	
HON	ouvir	+	
JUN	reis	-	
FUS	força	==	
HIM	seu/sua	!=	
FAH	para	for	
LOS	é	if	
FOD	quando	while	
AAN	um/uma	or	
ANRK	e	and	
NUST	eles	not	

IMPLEMENTAÇÃO

A implementação foi realizada em Python, utilizando uma estrutura originalmente desenvolvida para Autômato de Pilha (PDA). No entanto, como não há operações de inserção ou remoção na pilha, o autômato se comporta de forma equivalente a um Autômato Finito Determinístico (AFD), suficiente para o reconhecimento léxico da linguagem Dovahzul.

A solução foi organizada em módulos para facilitar a manutenção e a extensibilidade:

- Classe AF (afd.py): Núcleo do autômato com algoritmo de reconhecimento.
- Delta (delta.py): Função de transição definida como dicionário (grafo dirigido).

- Main (main.py): Configuração de parâmetros e execução de casos de teste.

Algoritmo de Funcionamento

1. Leitura da Entrada: O programa recebe uma sequência de palavras (separadas por espaço ou delimitador #).
2. Processamento por Palavra: Cada palavra inicia no estado inicial q0.
3. Transições Caractere a Caractere: A navegação entre estados é realizada consultando o dicionário de transições.
4. Validação Final: Caso a palavra termine em um estado de aceitação, é reconhecida como token válido. Caso contrário, é rejeitada.
5. Registro de Execução: Caminhos percorridos e tokens reconhecidos são armazenados em uma tabela de símbolos.

Estruturas de Dados

- AUTÔMATO (grafo): Representado por um dicionário {estado: {caractere: próximo_estado}}.
- Tabela de Símbolos (TS): Armazena tuplas no formato (linha, palavra, token).
- Fita de Execução: Lista de estados percorridos em cada análise.

Saída

- Lista de tokens reconhecidos.
- Caminho percorrido pelo autômato.
- Indicação de aceitação ou rejeição da palavra.

AUTÔMATO FINITO:

LINK DAS PLANILHAS