Davi Herick

Maria Gabriela

Rebeca Soares

# CADASTRAMENTO DE CARROS

## LUA

BETIM

2025

## LINGUAGEM UTILIZADA

A linguagem utilizada para desenvolver este programa foi a Lua, uma linguagem de programação criada no Brasil. Embora amplamente aplicável, Lua se destaca especialmente na criação de jogos e sistemas embarcados — sistemas projetados para realizar tarefas específicas com alta eficiência. Sendo leve, flexível e tendo facilidade na integração com outras linguagens ela é ideal para projetos que exigem desempenho e modularidade. Lua também é bastante conhecida por sua capacidade de ser incorporada em outros sistemas, como motores de jogos, permitindo a criação de scripts e extensões dinâmicas. Entre os jogos conhecidos que utilizam Lua, destacam-se World of Warcraft, SimCity 4 e GTA RP.

Jogo eletrônico entre outros.

## CONTEXTUALIZAÇÃO DA LINGUAGEM LUA

Sendo uma linguagem brasileira, criada em 1993 na PUC-Rio, por uma equipe de pesquisadores brasileiros. Seu desenvolvimento teve como objetivo atender à demanda por uma linguagem de script que fosse leve, rápida e fácil de embutir em outras aplicações, especialmente em projetos de engenharia e automação industrial. Desde então, Lua tem ganhado destaque internacional pela sua simplicidade e eficiência.

## CADASTRAMENTO DE CARROS

O projeto escolhido foi um sistema de cadastramento de carros, tendo um menu com:

1. **Cadastrar carro**: permite o cadastramento do carro, com seu nome e ano, permitindo apenas um carro com o mesmo nome.
2. **Listar carros:** lista todos os carros cadastrados.
3. **Remover carro**: remove o carro que foi encontrado a partir do nome digitado.
4. **Buscar carros:** busca o carro a partir de seu nome.
5. **Editar carro:** edita as informações do carro (nome e ano).
6. **Limpar lista:** remove todos os carros já cadastrados.
7. **Sair:** Fecha o sistema.

## SINTAXE DA LINGUAGEM LUA

### Declaração de Variáveis

Em Lua, as variáveis são geralmente declaradas com a palavra-chave *“local”*, o que limita seu escopo ao bloco onde foram criadas. As estruturas de dados mais utilizadas são as tabelas, representadas por chaves {}, que funcionam como listas, dicionários ou objetos.

### Declaração de Funções

As funções são definidas utilizando a palavra '*function*' e finalizadas com *'end'*. Diferente do Python, que utiliza '*def'*, Lua delimita claramente o início e fim do bloco da função com palavras-chave.

**Estruturas Condicionais**

Lua usa '*if', 'then', 'elseif', 'else'* e '*end* 'para criar estruturas de decisão. A palavra '*then'* inicia o bloco que será executado caso a condição seja verdadeira, '*elseif*' permite adicionar novas condições, e '*end*' finaliza toda a estrutura condicional.

**Laços de Repetição (Loops)**

Para percorrer listas ou tabelas, Lua oferece os iteradores '*ipairs*' e' *pairs*'. O '*ipairs*' é usado para percorrer índices numéricos sequenciais, enquanto '*pairs*' percorre todos os elementos, independentemente da chave. Esses laços são semelhantes ao for com enumerate em Python.

**Entrada e Saída de Dados**

A entrada de dados em Lua é realizada através de '*io.read*', com diferentes modos de leitura:

- *n: lê um número,
- *l: lê uma linha,
- *a: lê tudo até o final da entrada.

Para saída de dados, Lua utiliza print ou '*io.write*.'

**Manipulação de Tabelas**

As tabelas em Lua são estruturas poderosas que permitem armazenar qualquer tipo de dado. As principais funções para manipulá-las são:

- **table.insert**: adiciona um valor ao final da tabela,
- **table.remove:** remove um valor de um índice específico,
- **#tabela:** retorna a quantidade de elementos.

**Métodos de String**

Lua disponibiliza diversas funções nativas para manipulação de textos:

- **string.lower:** converte a *string* para letras minúsculas,
- **string.gsub:** substitui trechos da *string* com base em um padrão,
- **string.find:** localiza a posição de uma *substring* dentro da *string.*

**Loops Infinitos**

Lua permite a criação de loops infinitos, que são úteis em sistemas que exigem execução contínua até uma condição externa ser atendida. Isso pode ser feito com as *estruturas ‘while’ ‘true’,’ for’* ou ’ *repeat’*.