Relatório do Projeto Luify: Gerenciador de Músicas em C (CLI)

Larissa Vitória De Oliveira Alves Santos

Matricula:

Data: 22 de julho de 2025

1. Introdução ao Luify

O Luify é um projeto desenvolvido em linguagem C que simula um gerenciador de músicas via linha de comando (CLI). O objetivo principal é demonstrar as operações fundamentais de um sistema de gerenciamento de dados, comumente conhecido como **CRUD** (Create, Read, Update, Delete), aplicadas ao contexto de metadados de músicas (título, artista, álbum, ano).

É crucial entender que, apesar do nome "Luify" evocar plataformas de streaming como o Spotify, este projeto **não inclui interface gráfica (GUI), funcionalidades de reprodução de áudio, recursos de rede (streaming online) ou um sistema de banco de dados robusto**. Ele foca na lógica de manipulação e persistência de dados em um ambiente de terminal, servindo como uma base educacional para compreensão de conceitos de programação em C e manipulação de arquivos em baixo nível.

1. Estrutura e Componentes do Código

O código-fonte do Luify é modular e estruturado para facilitar a compreensão e manutenção.

2.1. Bibliotecas e Macros

- **stdio.h**: Essencial para todas as operações de entrada e saída (leitura de teclado, impressão na tela).
- **stdlib.h**: Fornece funções de propósito geral, como alocação de memória e controle do programa.
- **string.h**: Indispensável para a manipulação de strings (títulos, nomes de artistas, etc.), incluindo cópia, comparação e busca.
- **MAX_MUSICAS (100)**: Define o limite máximo de músicas que o sistema pode armazenar em um array estático.
- **MAX_STR (100)**: Especifica o tamanho máximo para strings (como título, artista), prevenindo *buffer overflows*.

2.2. Estrutura de Dados Musica

```
typedef struct {
    int id;
    char titulo[MAX_STR];
```

```
    char artista[MAX_STR];

    char album[MAX_STR];

    int ano;

} Musica;
```

A estrutura Musica é o "modelo" fundamental para cada registro de música. Ela encapsula todos os atributos de uma música (ID único, título, artista, álbum e ano de lançamento) em uma única unidade de dados.

2.3. Variáveis Globais

- **Musica musicas[MAX_MUSICAS]**: Um array global que atua como o repositório de dados em memória para todas as músicas cadastradas.
- **int num_musicas**: Mantém a contagem atual de músicas armazenadas no array.
- **int proximo_id**: Gerencia a atribuição de IDs sequenciais e únicos para cada nova música, garantindo a identificação exclusiva.

2.4. Funções Principais

O programa é dividido em funções claras, cada uma com uma responsabilidade específica:

- **main()**: Ponto de entrada do programa. Responsável por carregar dados, iniciar o menu principal e salvar os dados ao encerrar.
- **menu_principal()**: Exibe as opções para o usuário e coordena as chamadas para outras funções com base na escolha.
- **cadastrar_musica()**: Adiciona uma nova música ao array, solicitando os detalhes ao usuário. Implementa lógica para pré-preencher dados (usado para a música inicial).
- **listar_musicas()**: Percorre e exibe todos os detalhes das músicas cadastradas.
- **editar_musica()**: Permite a modificação dos atributos de uma música específica, identificada por seu ID.
- **buscar_musica()**: Implementa uma funcionalidade de pesquisa por título ou artista.
- **remover_musica()**: Exclui uma música do array, realocando os elementos subsequentes para manter a integridade da lista.
- **salvar_musicas_em_arquivo()**: Salva o estado atual do programa (músicas, num_musicas, proximo_id) em um arquivo binário (luify_data.dat). Isso garante a **persistência dos dados**, evitando perdas ao fechar o programa.
- **carregar_musicas_de_arquivo()**: Carrega os dados do arquivo binário ao iniciar o programa, restaurando o estado anterior.
- **adicionar_musica_inicial()**: Função específica para cadastrar a "pior música que possa imaginar" (definida como "O Grito Silencioso do Caos" de "A Orquestra do Desespero", álbum "Fragmentos Incompreensíveis", ano 2019) automaticamente na primeira execução ou quando o arquivo de dados não é encontrado.

3. Exemplo da "Pior Música que Possa Imaginar"

Para demonstrar o funcionamento do Luify, uma música de exemplo foi incorporada:

- **Título**: O Grito Silencioso do Caos
- **Artista**: A Orquestra do Desespero
- **Álbum**: Fragmentos Incompreensíveis
- **Ano**: 2019

Essa música é adicionada automaticamente ao sistema se nenhum dado anterior for encontrado ao iniciar. Ela serve para ilustrar as funcionalidades de listagem, edição, busca e remoção sem a necessidade de um cadastro manual inicial.

4. Como Compilar e Executar o Luify

Para interagir com o Luify, siga os passos abaixo usando um terminal e um compilador C (como o GCC).

4.1. Pré-requisitos

- **Compilador C (GCC)**: Certifique-se de ter o GCC instalado em seu sistema operacional (Linux, macOS, Windows com MinGW/Git Bash).

4.2. Passos

1. **Salve o Código**
2. Salve o código-fonte fornecido (o conteúdo do luify_manager.c) em um arquivo chamado luify_manager.c em uma pasta de sua escolha.
3. **Abra o Terminal**
4. Navegue até a pasta onde você salvou o arquivo luify_manager.c.
5. **Compile o Código**
6. Execute o comando de compilação:
7. gcc luify_manager.c -o luify_manager
8. Este comando criará um arquivo executável chamado luify_manager (ou luify_manager.exe no Windows).
9. **Execute o Programa**
10. Inicie o Luify com o comando:
11. ./luify_manager
12. (No Windows, você pode precisar digitar luify_manager.exe ou simplesmente luify_manager).

4.3. Interação

Após a execução, o menu principal do Luify aparecerá no terminal, permitindo que você escolha entre cadastrar, listar, editar, buscar, remover músicas ou sair. O programa salvará automaticamente suas alterações no arquivo luify_data.dat ao ser encerrado corretamente.

1. Licença e Contribuições

Este projeto é disponibilizado sob a **Licença MIT**, o que significa que é de código aberto e permite o uso, modificação e distribuição para fins educacionais e comerciais. Detalhes completos podem ser encontrados no arquivo LICENSE no repositório do GitHub.

Contribuições são bem-vindas! Se você tiver ideias para aprimoramentos, sugestões de novas funcionalidades ou quiser corrigir bugs, encorajamos a abertura de *issues* ou o envio de *pull requests* no repositório do GitHub.