

UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
LABORATÓRIO DE ALGORITMOS E ESTRUTURA DE DADOS 1

Relatório 1 – Sistema de Gerenciamento de Biblioteca

Anderson Carlos da Silva Moraes
Marilia Fonseca Andrade

Pau dos Ferros - RN

22/06/2025

Arquitetura e Organização

Para entender o projeto, sua arquitetura e organização são separadas em três categorias: lógica de estados, estrutura da arquitetura e lógica e funcionalidade dos módulos. Além disso, o código fonte está disponível em: [marifa16/Trabalho1_Lab_estrutural](https://github.com/marifa16/Trabalho1_Lab_estrutural).

Lógica de Estados

O projeto é construído usando lógica de máquina de estados, onde cada módulo é representado por um estado distinto. O estado pode mudar com base na entrada fornecida pelo usuário. O fluxo do programa é gerenciado pelo arquivo src/main.c por meio de um switch que aponta para o módulo selecionado, e o estado da enumeração definido em include/estruturas.h define os estados possíveis:

- MAIN_MENU
- BOOK_MENU
- EMPRESTIMO_MENU
- RELATORIO_MENU
- SAIR

Estrutura de Arquivos

- /bin: destinada ao arquivo executável gerado pelo projeto, representado pelo executável main;
- /data: armazena os arquivos CSV persistentes do projeto, com os arquivos: books.csv e emprestimos.csv.
- /include: contém os arquivos cabeçalho (.h) que declara os protótipos de funções e estruturas:
 - estruturas.h;
 - menus.h;
 - book_manager.h;
 - empréstimo_manager.h;
 - relatório.h;
 - auxiliar.h;
 - files_manager.

- /src: contém os arquivos fonte (.c) com os códigos e funcionalidades do projeto:
 - main.c;
 - menus.c;
 - user_handle.c;
 - book_manager.c;
 - emprestimo_manager.c;
 - relatorio.c;
 - auxiliar.c;
 - files_manager.c.
- Makefile: arquivo para automatizar o processo de compilação.
- README.md: arquivo de documentação do projeto com breve descrição do projeto e como se usar;

Conceitos e estruturas

Esta seção é dedicada a explicar a lógica e a necessidade de cada elemento do projeto, por direção e arquitetura; as funções são apresentadas na seção seguinte. Devido à complexidade e extensão dos módulos, a descrição visa fornecer o propósito geral de cada módulo, enfatizando suas responsabilidades na operação do sistema.

Include

Contém todos os arquivos (.h) do cabeçalho. Eles são responsáveis por anunciar as estruturas de dados (structs), enumerações (enums) e protótipos de funções que serão usados ao longo do projeto, permitindo o compartilhamento de vários estilos de codificação. Alguns arquivos de atenção são:

- **estruturas.h**: arquivo básico que descreve as estruturas de dados e listas usadas em todo o sistema, como:
 - Enums: state, genero, status e status_book.
 - Structs: book e empréstimo.
- **auxiliar.h**: estão incluídos exemplos de protótipos de funções auxiliares para manipulação de strings, comunicação de enumeração para string e validação de entradas (títulos, autores, ISBN).

Src

Contém todos os arquivos código-fonte (.c) onde a lógica do sistema é implementada. Cada entrada neste diretório corresponde a um módulo funcional específico. Alguns arquivos de atenção são:

- **main.c**: inicia o programa, configura o console para UTF-8 e entra no loop principal da máquina de estados, exibindo os menus apropriados com base na interação do usuário.
- **menus.c**: regula o fluxo de navegação entre o menu principal e os submenus para gerenciamento de livros, artigos e relatórios.
- **user_handle.c**: atua como uma camada intermediária que reúne e verifica as entradas do usuário antes de chamar as funções de gerenciamento (book_manager, emprestimo_manager).

- **book_manager.c**: gere todas as operações CRUD para livros, incluindo leitura e escrita, no arquivo books.csv.
- **emprestimo_manager.c**: criação e atualização do empréstimo, atualização do status do livro e salvamento dos dados em emprestimos.csv.
- **relatorio.c**: utiliza as funções para listar livros e artigos com base nos filtros selecionados pelo usuário (gênero, status).
- **auxiliar.c**: inclui funções de suporte como limpar_buffer, validar_titulo, validar_autor, get_maior_id e outras que melhoram a robustez e a legibilidade do código.
- **files_manager.c**: responsável por confirmar a existência do arquivos.csv na inicialização do sistema e criá-los com o cabeçalho apropriado, se necessário.

Funcionalidades

Esta seção consiste em determinar onde cada requisito da atividade avaliativa é implementada no projeto.

Módulo de Livros

Este módulo concentra-se em todas as operações relacionadas ao acervo de livros da biblioteca.

- **Adicionar novos livros ao acervo:** Permite o cadastro de um novo livro com título, autor, ISBN e gênero. O livro é marcado como "Disponível".
- **Eliminar livros existentes:** Com base em seu ID, um livro pode ser encontrado e removido do sistema.
- **Atualizar informações do livro:** Permite que os usuários alterem detalhes do livro registrados anteriormente, como título, autor, ISBN ou gênero.
- **Busca de livros:** Oferece diversas opções de busca, incluindo por ID, título, autor, ISBN ou gênero, e exibe uma lista dos resultados encontrados.

Módulo de Empréstimos

Este módulo supervisiona todo o processo de venda e distribuição de livros.

- **Registrar um novo empréstimo:** Um usuário (leitor) pode solicitar um empréstimo. O sistema vincula o leitor a um livro assim que o livro estiver com o status "Disponível". O status do livro é atualizado para "Emprestado".
- **Marcando um livro como devolvido:** Quando um livro é devolvido, o status do empréstimo é marcado como "Concluído" e o status do livro se torna "Disponível" para que possa ser reutilizado por outro usuário.
- **Buscar empréstimos pelo nome do leitor:** O sistema permite listar todos os resultados relacionados a um determinado leitor.

Módulo de Relatórios

Este módulo permite a extração de informações compiladas sobre as operações e o acervo da biblioteca.

- **Listar livros por status:** Fornece uma lista de livros de acordo com seu status atual (disponível, indisponível ou emprestado).
- **Listar livros por gênero:** Permite ao usuário filtrar e visualizar todos os livros que pertencem a um gênero específico (ficção, didático e biografia) ou a todos os gêneros.

Especificações Técnicas

Ponteiros

No projeto o uso de ponteiro é visto em:

- passagem de structs como parâmetro, para evitar cópias desnecessárias de dados, as funções recebem ponteiros para as estruturas.
- Retorno de ponteiros em funções de busca retornam um ponteiro para um array alocado dinamicamente levando em consideração os resultados. Também pode-se manipular arquivos, como no tipo FILE * é um ponteiro usado para identificar e manipular fluxos de arquivos CSV. Como podemos ver nas Figura 1 e Figura 2.

```
void create_book(book *temp_book)
{
    int id = get_maior_id(books_file) + 1;
    temp_book->stats_bk = DISPONIVEL;

    FILE *arquivo = fopen(books_file, "a"); // Abre o arquivo para adicionar no final
    if (arquivo == NULL)
    {
        printf("Erro ao abrir o arquivo '%s' para escrita.\n", books_file);
        return;
    }

    fprintf(arquivo, "%d,%s,%s,%d,%d\n", id, temp_book->title, temp_book->autor, temp_book->ISBN, temp_book->gen, temp_book->stats_bk); // Escreve o novo id
    fclose(arquivo);
}
```

Figura 1. Passagem de structs. Autoria própria

```
void create_book(book *temp_book);
```

Figura 2. Retorno de ponteiros. Autoria própria

Manipulação de Strings com Ponteiros

A parte da manipulação de strings é feita majoritariamente com ponteiros (char *), o que permite o processamento eficiente de nomes, títulos e outros textos. Um exemplo óbvio é a função personalizada strcasestr, que busca uma substring dentro de uma string, classificando os caracteres um por um. Como podemos ver na Figura 3.

```

char *strcasestr(const char *haystack, const char *needle)
{
    if (!*needle)
    {
        return (char *)haystack;
    }
    for (; *haystack; haystack++)
    {
        if (tolower((unsigned char)*haystack) == tolower((unsigned char)*needle))
        {
            const char *h, *n;
            for (h = haystack, n = needle; *h && *n; h++, n++)
            {
                if (tolower((unsigned char)*h) != tolower((unsigned char)*n))
                {
                    break;
                }
            }
            if (!*n)
            {
                return (char *)haystack;
            }
        }
    }
    return NULL;
}

```

Figura 3. Manipulação de Strings com Ponteiros. Autoria Própria

Alocação Dinâmica de Memória

As funções de realloc e free são usadas para gerar vetores dinâmicos, permitindo que o programa se adapte à quantidade de dados sem um limite definido. A função realloc é usada em nossas funções de busca para aumentar o tamanho dos resultados para cada novo item encontrado. Para evitar lapsos de memória, a memória atribuída aos resultados da busca é liberada após o uso. Como podemos ver nas Figura 4 e Figura 5.

```

if (found)
{
    (*matchs)++;
    // Realoca memória para o novo livro encontrado
    book *temp = realloc(temp_books, (*matchs) * sizeof(book));
    if (temp == NULL)
    {
        printf("Erro de alocação de memória!\n");
        free(temp_books);
        fclose(arquivo);
        return NULL;
    }
    temp_books = temp;
    // Copia o livro encontrado para o final do array dinâmico
    temp_books[(*matchs) - 1] = aux_book;
}

```

Figura 4. Uso de realloc. Autoria própria

```

if (count > 0)
{
    printf("\nDigite o ID do livro que deseja remover: ");
    int id_del = 0;
    scanf(" %d", &id_del);
    Limpar_buffer();

    delete_book(id_del);
}

free(books); // libera os livros da função searching_book

```

Figura 5. Uso de free. Autoria própria

Vetores Dinâmicos

Ponteiros são usados no projeto para manipular strings de forma eficiente, como na função de busca de texto. Ele cria vetores dinâmicos que crescem ao longo do tempo usando realloc, alocando memória apenas quando necessário e evitando desperdícios para proteger os resultados dessas buscas. Por fim, para salvar os dados com segurança, o primeiro sistema armazena todas as alterações em um arquivo temporário e só então

substitui o arquivo original, uma técnica que protege contra corrupção de dados. Como podemos ver na Figura 6.

```
if (found)
{
    (*matchs)++;
    // Realoca memória para o novo livro encontrado
    book *temp = realloc(temp_books, (*matchs) * sizeof(book));
    if (temp == NULL)
    {
        printf("Erro de alocação de memória!\n");
        free(temp_books);
        fclose(arquivo);
        return NULL;
    }
    temp_books = temp;
    // Copia o livro encontrado para o final do array dinâmico
    temp_books[(*matchs) - 1] = aux_book;
}
```

Figura 6. Vetor dinâmico. Autoria própria

Tipos Estruturados

Estruturas como books e emprestimo funcionam como moldes que reúnem todas as informações de uma entidade, como: título e autor de um livro. Para tornar o código mais seguro, legível e fácil de entender, enumerações como status_book são usadas para dar nomes claros a estados e categorias. Como podemos ver nas Figura 7 e Figura 8.

```
typedef struct
{
    int id;
    char Leitor[100];
    book Livro;
    struct tm data;
    status stats;
} emprestimo;
```

Figura 7. Uso de uma Struct no projeto. Autoria própria.

```

typedef enum
{
    CONCLUIDO,
    EM_ANDAMENTO,
    CANCELADO,
} status;

```

Figura 8. Uso de Enum no projeto. Autoria própria.

Manipulação de Arquivos

A persistência dos dados no projeto é alcançada por meio da manipulação de arquivos CSV. Para operações de leitura e escrita, as funções de leitura usam fgets para capturar uma linha inteira do documento e sscanf para extrair os campos, enquanto a escrita de novos dados é feita com fprintf. Uma abordagem mais segura é usada para as tarefas de atualização e exclusão de dados: o sistema cria um temporário onde os dados são copiados e alterados; ao final do processo, o temporário original é removido com a função remove, e o temporário é renomeado para o nome original, completando a operação. Como podemos ver nas Figura 9 e Figura 10.

```

while (fgets(Linha, sizeof(Linha), arquivo))
{
    int genero_int, status_int;
    if (sscanf(Linha, "%d,%99[^,],%99[^,],%13[^,],%d,%d",
               &temp_book.id, temp_book.title, temp_book.autor, temp_book.ISBN, &genero_int, &status_int) == 6)
    {
        if (temp_book.id == id)
        {
            temp_book.gen = (genero)genero_int;
            temp_book.stats_bk = (status)status_int;
            fclose(arquivo);
            return temp_book; // Retorna o livro encontrado
        }
    }
}

```

Figura 9. Leitura e Escrita. Autoria própria

```
fclose(arquivo);
fclose(temp);

remove(books_file);
rename("temp.csv", books_file);

if (executado)
{
    printf("\nLivro com ID %d foi removido com sucesso.\n", id);
}
else
{
    printf("\nAVISO: Livro com ID %d não foi encontrado para remoção.\n", id);
}
```

Figura 10. Atualizar e Deletar. Autoria própria

Persistência de Dados

Todas as informações sobre livros e empréstimos são salvas em arquivos CSV localizados no diretório de dados. Isso garante que os dados não sejam perdidos quando o programa for encerrado.

Manual de Uso

Módulo Livros

Ao acessar [1] - **Gerenciar livros** no menu principal. Este método permite o gerenciamento completo do acervo.

1. Adicionar um Livro

Esta opção direciona o usuário para o catálogo de um novo livro. O processo é interativo e requer confirmação em cada etapa.

- Passo 1: Selecione a opção [1] - **Adicionar livro** no menu "Gerenciamento de livros".
- Passo 2: O sistema solicitará o título do livro. Após a inserção, ele imprimirá o título digital e solicitará uma confirmação ([1] - SIM, [2] - NÃO, [3] - SAIR).
- Passo 3: O mesmo procedimento de inserção e verificação é repetido para o nome do autor e o código ISBN, que deve conter 13 dígitos.
- Passo 4: O sistema exibirá uma lista de opções para o gênero ([1] - Ficção, [2] - Didático, [3] - Biografia). O usuário deve inserir o número correspondente. Após a confirmação final, o livro é salvo no sistema.
- Passo 5: "livro adicionado com sucesso!" será exibido.

2. Remover Livro

Esta opção permite a exclusão de um livro do acervo.

- Passo 1: Escolha uma opção [2] - **Remover livro**.
- Passo 2: O sistema exibirá um menu de busca (função searching_book). O usuário deve escolher um critério (ID, Título, Autor, etc.) e fornecer as informações para a busca.
- Passo 3: Uma lista dos livros encontrados será apresentada ao final.
- Passo 4: O sistema solicitará que você "Digite o ID do livro que deseja remover:". O usuário deve inserir o ID correspondente na lista.
- Passo 5: O livro será removido e uma mensagem de sucesso será exibida.

3. Atualizar Livro

Permitir a alteração do conteúdo de um livro previamente cadastrado.

- Passo 1: Escolha uma opção **[3] — Atualizar**.
- Passo 2: O sistema solicitará a atualização do ID do livro: "Digite o ID do livro que deseja atualizar:".
- Passo 3: Se o livro for encontrado, um menu de atualização (upp_book) será exibido, permitindo a seleção do campo a ser alterado (**[1] – Título, [2] - Autor, etc.**).
- Passo 4: Após escolher um campo, o usuário insere as novas informações. O procedimento pode ser repetido em outros campos.
- Passo 5: Para finalizar, o usuário deve escolher a opção **[5] — Salvar Alterações e Sair**. As alterações permanecerão no arquivo.

Módulo Empréstimos

Acessado pela opção **[2] - Gerenciar Empréstimos** no menu principal.

1. Registrar Novo Empréstimo

Procedimento para cadastrar um exemplar de um livro.

- Passo 1: Selecione **[1] – Registrar novo empréstimo**.
- Passo 2: O sistema solicitará o nome do usuário (leitor) e solicitará confirmação.
- Passo 3: Em seguida, o usuário será direcionado ao menu de busca de livros para localizar o título desejado.
- Passo 4: O sistema solicitará o "ID do livro que deseja pegar emprestado" após a busca.
- Passo 5: Se o status do livro for "Disponível", o sistema o verificará internamente. Caso contrário, uma mensagem de erro será enviada e o processo de seleção de livros será reiniciado.
- Passo 6: Uma confirmação final sobre os termos do lote será fornecida. Após a confirmação, o lote será cadastrado com sucesso.

2. Marcar devolução

Procedimento para registrar a devolução de um livro.

- Passo 1: Escolha a opção **[2] - Marcar devolução**.

- Passo 2: O sistema solicitará o nome do usuário para encontrar o livro com esse nome (função searching_emp).
- Passo 3: Uma lista de atividades do usuário será apresentada.
- Passo 4: O sistema solicitará "Digite o ID do lote que deseja devolver:". O usuário deve inserir o ID da lista.
- Passo 5: O sistema atualizará automaticamente o status do lote para "Concluído" e o status do livro correspondente para "Disponível".

3. Listar empréstimo por status

Permite ao usuário visualizar a lista de todos filtrados por seu status (Concluído, Em Andamento e Cancelado).

Módulo Relatórios

O Módulo Relatório do menu principal é acessado através da opção **[3] – Módulo Relatório**.

- Passo 1: Dentro deste menu, o usuário pode escolher entre **[1] - Listar livros por status do livro** e **[2] - Listar livros por gênero**.
- Passo 2: Caso escolha a opção **[2] - Gerenciar Empréstimos**, o usuário acessa o menu de empréstimos.
- Passo 3: Se for a opção **[3] - Listar empréstimos por status** executa a função list_emp_stats para gerar o relatório correspondente.