Documentação do Código de Empréstimo de Livros

Esta documentação detalha, linha a linha, todas as partes do código C que implementa um sistema básico de cadastro, aluguel, devolução, renovação e pagamento de multas para até cinco livros.

---

1. Diretivas e Constantes Globais

```
#include <stdio.h>
#include <string.h>

#define VALOR_MULTA 5.0 // Valor da multa por dia de atraso
#define SENHA  "admin123" // Senha para acessar o sistema
```

- `#include <stdio.h>`: importa funções de entrada e saída padrão (como `printf`, `scanf`).
- `#include <string.h>`: importa funções de manipulação de strings (como `strlen`, `strcpy`, `strcmp`).
- `#define VALOR_MULTA 5.0`: define valor fixo da multa diária (R$ 5,00).
- `#define SENHA "admin123"`: define, como literal, a senha de administrador para cadastrar livros.

---

2. Declaração de Variáveis Globais para até 5 Livros

Para cada livro (1 a 5), há um conjunto de variáveis que armazenam:

- **`tituloX`** (`char[50]`): título do livro.
- **`locadorX`** (`char[50]`): nome da pessoa que alugou.
- **`diasRestantesX`** (`int`): prazo (em dias) original ou remanescente para devolução.
- **`locadoX`** (`int`): flag (0 ou 1) indicando se está alugado.
- **`nomemultadoX`** (`char[50]`): nome da pessoa multada ao devolver com atraso.

- **`valormultaX`** (`float`): valor da multa calculada.

// Livro 1
typedef struct { char titulo[50]; char locador[50]; int diasRestantes; int locado; char nomemultado[50]; float valormulta; } Livro;

Livro livro1 = {"", "", 0, 0, "", 0.0f};
Livro livro2 = {"", "", 0, 0, "", 0.0f};
Livro livro3 = {"", "", 0, 0, "", 0.0f};
Livro livro4 = {"", "", 0, 0, "", 0.0f};
Livro livro5 = {"", "", 0, 0, "", 0.0f};

> Observação: na versão original, as variáveis são separadas por livro (`titulo1`, `locador1`, ...), aqui apresentamos agrupadas em `struct` apenas para clareza.

---

3. Função `temNumero`

```
int temNumero(const char texto[]) {
    for (int i = 0; texto[i] != '\0'; i++) {
        if (texto[i] >= '0' && texto[i] <= '9') {
            return 1; // Tem número
        }
    }
    return 0; // Não tem número
}
```

- **Objetivo**: verifica se a string `texto` contém qualquer dígito numérico.
- **Lógica**:
    1. Percorre cada caractere até o terminador `'\0'`.
    2. Se encontrar caractere `'0'..'9'`, retorna 1 (verdadeiro).
    3. Se terminar o laço sem encontrar, retorna 0.

---

4. Função `estaMultado`

```
int estaMultado(const char nome[50]) {
```

```
    return ((strcmp(nome, nomemultado1) == 0 && valormulta1 > 0) ||
            (strcmp(nome, nomemultado2) == 0 && valormulta2 > 0) ||
            (strcmp(nome, nomemultado3) == 0 && valormulta3 > 0) ||
            (strcmp(nome, nomemultado4) == 0 && valormulta4 > 0) ||
            (strcmp(nome, nomemultado5) == 0 && valormulta5 > 0));
}
```

- **Objetivo**: checar se o `nome` informado está pendente de multa em algum dos cinco livros.
- **Lógica de negócio**:
    - Para cada livro de 1 a 5:
        - compara `nome` com `nomemultadoX`
        - verifica se `valormultaX > 0`
    - Se qualquer condição for verdadeira, retorna 1, senão 0.

---

5. Função `estaAdicionado`

```
int estaAdicionado(const char titulo[50]) {
    return (strcmp(titulo, titulo1) == 0 || strcmp(titulo, titulo2) == 0 ||
            strcmp(titulo, titulo3) == 0 || strcmp(titulo, titulo4) == 0 ||
            strcmp(titulo, titulo5) == 0);
}
```

- **Objetivo**: verificar se um livro com aquele `titulo` já foi cadastrado.
- **Lógica**: compara o `titulo` com cada variável `tituloX`; retorna 1 se algum igual.

---

6. Função `adicionarLivro`

```
void adicionarLivro() {
    char senha[20];
    char titulo[50];
    int diasDevolucao;

    printf("Digite a senha para adicionar um livro: ");
```

```
    scanf("%s", senha);
    if(strcmp(senha, SENHA) != 0) { ... }
    // Se senha correta, solicita título e dias
    getchar();
    fgets(titulo, 50, stdin);
    // Remove '\n' final
    size_t len = strlen(titulo);
    if (len > 0 && titulo[len-1] == '\n') titulo[len-1] = '\0';

    if (estaAdicionado(titulo)) { ... }

    printf("Digite o número de dias para devolução:  \n");
    scanf("%d", &diasDevolucao);

    // Insere no primeiro slot livre (tituloX vazio)
    if(strlen(titulo1) == 0) { strcpy(titulo1, titulo); diasRestantes1 = diasDevolucao; printf(...); }
    else if(...) // repete para 2,3,4,5
    else { printf("Limite de livros atingido!\n"); }
}
```

- **Regra de negócio**:
    - **Autenticação**: solicita senha e compara com `SENHA`.
    - **Evita duplicação**: retorna se já cadastrado.
    - **Cadastro sequencial**: preenche o primeiro espaço disponível.
    - **Limite**: só até 5 livros.
- **Tratamento de entrada**:
    - Usa `getchar()` para limpar o buffer antes de `fgets()`.
    - Remove o `\n` deixado por `fgets`.

---

7. Função `listar_livros`

```
void listar_livros() {
    printf("\n--------------LIVROS--------------\n");
    printf("1- %s\n", titulo1);
    ... até 5
    printf("----------------------------------\n");
    getchar();
```

```
}
```

- **Objetivo**: exibir todos os títulos cadastrados (mesmo vazios).
- Usa `getchar()` para pausar até ENTER.

---

8. Função `alugarLivro`

```
void alugarLivro() {
    int escolha;
    listar_livros();
    printf("Escolha o número do livro que deseja alugar: ");
    scanf("%d", &escolha);
    switch (escolha) {
      case 1:
        if(strlen(titulo1) == 0) { printf("Livro não cadastrado!\n"); break; }
        if(locado1 == 1) { printf("Esse livro já está alugado por %s ..."); break; }
        getchar();
        do {
            printf("Digite seu nome: ");
            fgets(nomeLocador, 50, stdin);
            // remove '\n', checa se tem número
            if (temNumero(nomeLocador)) printf("Nome inválido!\n");
            if(estaMultado(nomeLocador)) { printf(...); return; }
        } while (temNumero(nomeLocador));
        strcpy(locador1, nomeLocador);
        locado1 = 1;
        printf("Livro '%s' alugado por '%s'. Prazo de %d dias.\n");
        break;
      // Cases 2 a 5 seguem mesmo padrão
      default: printf("Opção inválida!\n");
    }
    printf("\nPRESSIONE ENTER PARA VOLTAR AO MENU PRINCIPAL...");
    getchar(); getchar();
}
```

- **Regras de negócio**:
    1. **Disponibilidade**: só aluga se cadastrado e não ocupado.
    2. **Validação de nome**: não permite dígitos em nomes.

3. **Multas pendentes**: impede novo aluguel até pagar.
4. **Armazenamento**: salva `locadorX`, marca `locadoX = 1`.
5. **Pausa**: pede ENTER antes de voltar.

---

9. Função `devolverLivro`

```
void devolverLivro() {
    int escolha;
    int diasUsados;
    float valorMulta = 0;
    listar_livros();
    printf("Escolha o número do livro que deseja devolver: ");
    scanf("%d", &escolha);
    switch (escolha) {
      case 1:
        if(strlen(titulo1) == 0 || locado1 == 0) { printf("Livro não alugado!\n"); break; }
        printf("Quantos dias você usou o livro '%s'? ", titulo1);
        scanf("%d", &diasUsados);
        if(diasUsados > diasRestantes1) {
            valorMulta = (diasUsados - diasRestantes1) * VALOR_MULTA;
            printf("Valor da multa: R$ %.2f\n", valorMulta);
            strcpy(nomemultado1, locador1);
            valormulta1 = valorMulta;
            break;
        }
        locado1 = 0;
        printf("Livro '%s' devolvido com sucesso!\n", titulo1);
        titulo1[0] = '\0';
        break;
      // Cases 2 a 5 idem
      default: printf("Opção inválida!\n");
    }
    printf("\nPRESSIONE ENTER PARA VOLTAR AO MENU PRINCIPAL...");
    getchar(); getchar();
}
```

- **Regra de negócio**:
    1. **Verifica aluguel**: só devolve se `locadoX == 1`.

2. **Cálculo de multa**: diferença entre `diasUsados` e `diasRestantesX * VALOR_MULTA`.

3. **Armazena multa**: preenche `nomemultadoX` e `valormultaX`.

4. **Devolve sem multa**: libera slot (marca `locadoX = 0` e esvazia `tituloX`).

---

10. Função `pagarmulta`

```
void pagarmulta() {
    int encontrou = 0;
    if(strlen(nomemultado1) > 0) {
        printf("Locador multado: %s\n", nomemultado1);
        printf("Valor da multa: R$ %.2f\n", valormulta1);
        printf("Multa paga com sucesso!\n");
        nomemultado1[0] = '\0';
        valormulta1 = 0;
        encontrou = 1;
    }
    // Repete para livros 2 a 5
    if(!encontrou) {
        printf("Nenhum locador multado.\n");
    }
}
```

- **Objetivo**: efetivar pagamento de multas.
- **Lógica**:
    - Para cada `nomemultadoX` não vazio, exibe dados, zera multas e nomes.
    - Se nenhuma multa houver (`encontrou == 0`), informa "Nenhum locador multado.".

---

11. Função `renovarLivro`

```
void renovarLivro() {
    int escolha;
    listar_livros();
    printf("Escolha o número do livro que deseja renovar: ");
    scanf("%d", &escolha);
```

```
    switch (escolha) {
      case 1:
        if(strlen(titulo1) == 0 || locado1 == 0) { printf("Livro não alugado!\n"); break; }
        printf("Renovando livro '%s' por mais %d dias.\n", titulo1, diasRestantes1);
        diasRestantes1 += 7; // Renovação de 7 dias
        printf("Novo prazo: %d dias.\n", diasRestantes1);
        break;
      // Cases 2 a 5 idem
      default: printf("Opção inválida!\n");
    }
    printf("\nPRESSIONE ENTER PARA VOLTAR AO MENU PRINCIPAL...");
    getchar(); getchar();
}
```

- **Regra**: permite estender o prazo em 7 dias se o livro estiver alugado.

---

12. Função `main`

```
int main(){
    int opcao;
    do {
        printf("\n---------------MENU---------------\n");
        printf("1. Adicionar livro\n");
        printf("2. Listar livros\n");
        printf("3. Alugar livro\n");
        printf("4. Devolver livro\n");
        printf("5. Renovar livro\n");
        printf("6. Pagar multa\n");
        printf("7. Sair\n");
        printf("----------------------------------\n");
        printf("Escolha uma opção: ");
        scanf("%d", &opcao);

        switch (opcao) {
        case 1: adicionarLivro(); break;
        case 2: listar_livros(); break;
        case 3: alugarLivro(); break;
        case 4: devolverLivro(); break;
        case 5: renovarLivro(); break;
        case 6: pagarmulta(); break;
        case 7: printf("Saindo do sistema...\n"); break;
        default: printf("Opção inválida!\n");
        }
    } while (opcao != 7);
```

```
  return 0;
}
```

- **Descrição**:
    1. Exibe menu principal repetidamente.
    2. Lê opção do usuário.
    3. Chama função correspondente.
    4. Encerra quando opção for 7.

---

13. Resumo das Regras de Negócio

- Apenas usuário autenticado (senha) pode **cadastrar** livros.
- Máximo de 5 livros no sistema.
- Não permite títulos duplicados.
- Aluguel:
    - Só se cadastrado e não alugado.
    - Nome do locador não pode conter números.
    - Não aluga se houver multa pendente.
- Devolução:
    - Se atraso, calcula multa (`dias atrasados × VALOR_MULTA`).
    - Multa vinculada ao locador.
    - Sem atraso, libera livro.
- Renovação: adiciona 7 dias ao prazo.
- Pagamento de multa: limpa registros de multa.

Fim da documentação.