# Sistema de Gerenciamento de Biblioteca

## Análise Detalhada do Trabalho de Algoritmos e Estruturas de Dados

## 📋 ****RESUMO EXECUTIVO****

Você precisa desenvolver um **sistema completo de gerenciamento de biblioteca** em linguagem C, utilizando:

* **Lista encadeada** implementada em arquivo binário
* **Interface em modo texto** (prompt/terminal)
* **Importação de dados** via arquivo texto
* **Apresentação oral obrigatória**

**⏰ Prazo:** Menos de 10 dias  
**👥 Trabalho em grupo**  
**⚠️ Presença obrigatória na apresentação**

## 🏗️ ****ESTRUTURAS DE DADOS****

### ****1. Estrutura LIVRO****

typedef struct {

int codigo; // Identificador único

char titulo[151]; // Máximo 150 caracteres + \0

char autor[201]; // Máximo 200 caracteres + \0

char editora[51]; // Máximo 50 caracteres + \0

int edicao; // Número da edição

int ano; // Ano de publicação

int exemplares; // Exemplares disponíveis

// Ponteiro para próximo (posição no arquivo)

} Livro;

### ****2. Estrutura USUÁRIO****

typedef struct {

int codigo; // Identificador único

char nome[51]; // Máximo 50 caracteres + \0

// Ponteiro para próximo (posição no arquivo)

} Usuario;

### ****3. Estrutura EMPRÉSTIMO****

typedef struct {

int codigo\_usuario; // Código do usuário

int codigo\_livro; // Código do livro

char data\_emprestimo[11]; // DD/MM/AAAA + \0

char data\_devolucao[11]; // DD/MM/AAAA + \0 (vazio se não devolvido)

// Ponteiro para próximo (posição no arquivo)

} Emprestimo;

## 🗂️ ****ARQUIVO BINÁRIO E LISTA ENCADEADA****

### ****Conceito Principal****

* **NÃO** usar ponteiros de memória tradicionais
* **SIM** usar posições no arquivo como "ponteiros"
* Simular lista encadeada usando números de posição

### ****Estrutura do Arquivo****

[CABEÇALHO] -> [REGISTRO 1] -> [REGISTRO 2] -> [REGISTRO N] -> ...

| | | |

Posição 0 Posição X Posição Y Posição Z

### ****Registro de Cabeçalho****

typedef struct {

long cabeca\_lista; // Posição do primeiro registro

long topo\_arquivo; // Primeira posição livre no fim

} Cabecalho;

### ****Como Funciona****

1. **Cabeçalho** no início do arquivo (posição 0)
2. **Registros** sequenciais, cada um com "ponteiro" para o próximo
3. **Ponteiro = -1** indica fim da lista
4. **Novos registros** sempre no final (topo\_arquivo)

## 📱 ****MENU DO SISTEMA (11 Funcionalidades)****

### ****1. Cadastrar Livro****

* **Input:** código, título, autor, editora, edição, ano, exemplares
* **Ação:** Inserir na lista encadeada no arquivo binário
* **Validação:** Código único

### ****2. Imprimir Dados do Livro****

* **Input:** código do livro
* **Ação:** Buscar e exibir todas as informações
* **Tratamento:** "Livro não encontrado" se inexistente

### ****3. Listar Todos os Livros****

* **Output:** código, título, autor, exemplares de TODOS
* **Formato:** Tabular ou lista organizada

### ****4. Busca por Título****

* **Input:** título do livro (string)
* **Ação:** Buscar por correspondência exata ou parcial
* **Output:** Todas as informações do livro encontrado

### ****5. Calcular Total****

* **Output:** Número total de livros cadastrados
* **Implementação:** Percorrer lista e contar registros

### ****6. Cadastrar Usuário****

* **Input:** código, nome
* **Ação:** Inserir na lista de usuários
* **Validação:** Código único

### ****7. Emprestar Livro****

* **Input:** código usuário, código livro
* **Validações:**
  + Usuário existe?
  + Livro existe?
  + Há exemplares disponíveis?
* **Ações:**
  + Criar registro de empréstimo
  + Diminuir exemplares em 1
  + Data empréstimo = data atual
* **Erro:** "Não há exemplares disponíveis"

### ****8. Devolver Livro****

* **Input:** código usuário, código livro
* **Ações:**
  + Localizar empréstimo ativo
  + Registrar data devolução = data atual
  + Aumentar exemplares em 1

### ****9. Listar Livros Emprestados****

* **Output:** usuário, nome usuário, código livro, título livro, data empréstimo
* **Filtro:** Apenas empréstimos SEM data de devolução

### ****10. Carregar Arquivo****

* **Input:** arquivo .txt com formato específico
* **Processamento:** Parse das linhas e inserção em lote
* **Formato:** Explicado na seção seguinte

## 📄 ****FORMATO DO ARQUIVO DE IMPORTAÇÃO****

### ****Sintaxe Padrão****

TIPO;campo1;campo2;campo3;...

### ****Tipos de Registro****

#### ****L - Livro****

L;codigo;titulo;autor;editora;edicao;ano;exemplares

**Exemplo:**

L;7;Memorias Postumas de Bras Cubas;Machado de Assis;Bookman;4;2022;5

#### ****U - Usuário****

U;codigo;nome

**Exemplo:**

U;11;Joao da Silva

#### ****E - Empréstimo****

E;codigo\_usuario;codigo\_livro;data\_emprestimo;data\_devolucao

**Exemplos:**

E;5;27;03/02/2025;15/02/2025 // Já devolvido

E;22;4;15/05/2025; // Ainda não devolvido

### ****⚠️ Tratamento de Espaços****

O sistema deve **remover espaços** no início e fim de cada campo:

L; 7; Memorias Postumas ; Machado de Assis ; Bookman ;4 ;2022; 5

É igual a:

L;7;Memorias Postumas;Machado de Assis;Bookman;4;2022;5

## 💻 ****IMPLEMENTAÇÃO TÉCNICA****

### ****Funções Principais Necessárias****

#### ****Arquivo Binário****

void inicializar\_arquivo();

Cabecalho ler\_cabecalho();

void escrever\_cabecalho(Cabecalho cab);

long inserir\_registro(void\* registro, size\_t tamanho);

void\* buscar\_registro(long posicao, size\_t tamanho);

#### ****Lista Encadeada****

void inserir\_livro(Livro livro);

Livro\* buscar\_livro\_codigo(int codigo);

Livro\* buscar\_livro\_titulo(char\* titulo);

void listar\_todos\_livros();

int contar\_livros();

#### ****Empréstimos****

int emprestar\_livro(int cod\_usuario, int cod\_livro);

int devolver\_livro(int cod\_usuario, int cod\_livro);

void listar\_emprestimos\_ativos();

#### ****Importação****

void carregar\_arquivo\_texto(char\* nome\_arquivo);

void processar\_linha\_livro(char\* linha);

void processar\_linha\_usuario(char\* linha);

void processar\_linha\_emprestimo(char\* linha);

char\* trim\_espacos(char\* str);

### ****Estrutura de Arquivos Sugerida****

projeto/

├── main.c // Menu principal

├── biblioteca.h // Definições e protótipos

├── arquivo.c // Funções de arquivo binário

├── livros.c // Operações com livros

├── usuarios.c // Operações com usuários

├── emprestimos.c // Operações de empréstimo

├── importacao.c // Carregar arquivo texto

└── README.pdf // Documentação

## 📝 ****DOCUMENTAÇÃO OBRIGATÓRIA****

### ****Documentação de Funções****

Para cada função implementada, inclua:

/\*\*

\* Propósito: Insere um novo livro na lista encadeada

\* Pré-condições: arquivo inicializado, livro válido

\* Pós-condições: livro inserido, arquivo atualizado

\*/

void inserir\_livro(Livro livro) {

// implementação

}

### ****README.pdf****

Deve conter:

1. **Nomes dos integrantes do grupo**
2. **Lista de funcionalidades implementadas**
3. **Breve manual do usuário**
4. **Como compilar e executar**

## 🎯 ****CRITÉRIOS DE AVALIAÇÃO****

### ****Distribuição das Notas****

* **Documentação/Organização:** Peso 1 (10%)
* **Implementação das Funcionalidades:** Peso 9 (90%)

### ****Apresentação Oral****

* **OBRIGATÓRIA** para todos os integrantes
* Arguição sobre código e teoria
* **Ausência = Nota ZERO**

### ****⚠️ Alertas Importantes****

* **Trabalhos copiados:** NOTA ZERO
* **Plágio:** NOTA ZERO
* **Ausência na apresentação:** NOTA ZERO

## 📅 ****CRONOGRAMA SUGERIDO (10 DIAS)****

### ****Dias 1-2: Planejamento e Estrutura****

* Definir estruturas de dados
* Criar esqueleto dos arquivos
* Implementar funções básicas de arquivo

### ****Dias 3-4: Funcionalidades Básicas****

* Cadastro de livros e usuários
* Busca e listagem básica

### ****Dias 5-6: Empréstimos****

* Implementar empréstimo e devolução
* Listagem de empréstimos ativos

### ****Dias 7-8: Importação e Refinamentos****

* Função de carregar arquivo
* Tratamento de erros
* Validações

### ****Dias 9-10: Documentação e Testes****

* Documentar código
* Criar README
* Testes finais
* Preparar apresentação

## 🚀 ****DICAS DE IMPLEMENTAÇÃO****

### ****1. Comece Simples****

* Implemente primeiro sem arquivo binário (usando arrays)
* Depois migre para arquivo binário

### ****2. Debug e Testes****

* Crie funções para "debugar" o arquivo binário
* Teste cada funcionalidade separadamente

### ****3. Tratamento de Erros****

* Sempre verifique se arquivo foi aberto
* Valide entradas do usuário
* Trate casos de registros não encontrados

### ****4. Organização****

* Use nomes descritivos para variáveis e funções
* Mantenha funções pequenas e específicas
* Comente código complexo

## ❓ ****PONTOS DE ATENÇÃO****

### ****Dúvidas Comuns****

**"Como implementar lista encadeada em arquivo?"**

* Use long para armazenar posições no arquivo
* fseek() para navegar para posições específicas
* Simule ponteiros com números

**"Como tratar datas?"**

* Use strings no formato "DD/MM/AAAA"
* Para data atual, use funções da biblioteca time.h

**"Como remover espaços dos campos?"**

* Implemente função trim() que remove espaços do início e fim

**"Como validar códigos únicos?"**

* Percorra lista antes de inserir
* Retorne erro se código já existir

## 🎯 ****CHECKLIST FINAL****

### ****Funcionalidades****

* [ ] Cadastrar livro
* [ ] Imprimir dados do livro
* [ ] Listar todos os livros
* [ ] Busca por título
* [ ] Calcular total de livros
* [ ] Cadastrar usuário
* [ ] Emprestar livro
* [ ] Devolver livro
* [ ] Listar livros emprestados
* [ ] Carregar arquivo texto

### ****Requisitos Técnicos****

* [ ] Lista encadeada em arquivo binário
* [ ] Registro de cabeçalho
* [ ] Interface em modo texto
* [ ] Tratamento de espaços em branco
* [ ] Validações apropriadas

### ****Entregáveis****

* [ ] Código fonte documentado (.c e .h)
* [ ] README.pdf completo
* [ ] Preparação para apresentação oral

**Boa sorte com o trabalho! 🍀**