

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
CIÊNCIA DA COMPUTAÇÃO
ALGORITMOS E ESTRUTURAS DE DADOS I

HEMILINEI TALITA DAL SANTO - 12421BCC014
LAIS CUSTÓDIO BORGES - 12421BCC009

TRABALHO FINAL:
SOLUÇÃO USANDO LISTA DUPLAMENTE ENCADEADA

UBERLÂNDIA, MG
2025

1. INTRODUÇÃO

Este trabalho apresenta uma solução para o gerenciamento de consultas médicas utilizando a estrutura de lista duplamente encadeada (LDE). O objetivo principal foi criar um sistema capaz de armazenar e manipular dados de pacientes, descrições de consultas e medicamentos associados a cada uma delas. A escolha da LDE se justifica pela sua eficiência em operações de inserção e remoção em posições arbitrárias, o que é fundamental para um sistema de gerenciamento dinâmico como este, onde consultas e medicamentos podem ser adicionados ou removidos a qualquer momento.

A solução foi projetada com duas estruturas de dados principais, ambas implementadas como listas duplamente encadeadas. A primeira, a lista principal de Consultas (Consultation), armazena as informações de cada consulta, a lista de medicamentos e permite navegação bidirecional. A segunda, aninhada dentro de cada nó de consulta, é uma lista duplamente encadeada de Medicamentos (Medication), permitindo que cada consulta possua sua própria lista de medicamentos e que a manipulação de itens (adição e remoção) seja eficiente. Essa abordagem hierárquica e a escolha por listas duplamente encadeadas garantem uma organização lógica e otimizada dos dados.

O problema foi resolvido através da implementação de um conjunto robusto de funções essenciais para a manipulação das listas, que foram divididas em arquivos .c e .h para modularizar o código, tornando-o mais organizado e fácil de manter. As principais funcionalidades incluem:

- **Inclusão:** A adição de dados é feita por funções que adicionam novas consultas à lista principal e novos medicamentos a consultas existentes.
- **Remoção:** A exclusão de dados é tratada por funções que removem consultas completas ou medicamentos específicos de uma consulta.
- **Busca e Obtenção:** O sistema permite a recuperação de informações de consultas e medicamentos para visualização, assim como a obtenção do tamanho das listas.
- **Funções Adicionais:** Além das operações básicas, o sistema inclui funcionalidades extras como o carregamento de dados a partir de um arquivo de texto e a geração de estatísticas.

2. DOCUMENTAÇÃO DO CÓDIGO

Este capítulo apresenta a documentação técnica das principais funções implementadas nos arquivos `consultation.c` e `display.c`, detalhando seu propósito e funcionamento no contexto da solução desenvolvida.

2.1 Funções de Manipulação da Lista de Consultas (`consultation.c`)

Tabela 1: Funções presentes em `consultation.c` e suas descrições

Função	Descrição
<code>List *create_list()</code>	Aloca memória para a estrutura da lista principal (<code>List</code>) e a inicializa, definindo o <code>head</code> , <code>tail</code> e <code>size</code> como nulos ou zero.
<code>int free_list(List *li)</code>	Libera a memória de todos os nós da lista de consultas e, para cada consulta, libera a memória de todos os medicamentos associados antes de liberar a lista principal.
<code>int insert_consultation(...)</code>	Cria um novo nó de consulta, copia os dados fornecidos e o insere no final da lista principal. Se a lista estiver vazia, o novo nó se torna o <code>head</code> e o <code>tail</code> ; caso contrário, é adicionado após o nó <code>tail</code> .
<code>int insert_medication(...)</code>	Percorre a lista de consultas para encontrar a posição correta. Em seguida, cria um novo nó de medicamento e o insere no final da lista de medicamentos da consulta encontrada.
<code>int remove_consultation(List *li, int pos)</code>	Localiza a consulta na posição <code>pos</code> , atualiza os ponteiros (<code>next</code> e <code>prev</code>) dos nós adjacentes para removê-la da lista, e libera a memória da consulta e de todos os medicamentos associados.
<code>int remove_medication(...)</code>	Encontra a consulta e, em seguida, navega pela lista de medicamentos para localizar a posição <code>med_pos</code> . Atualiza os ponteiros dos nós adjacentes para remover o medicamento e, finalmente, libera sua memória.
<code>int get_consultation_info(..)</code>	Percorre a lista até a posição desejada para obter e copiar os dados da consulta para as variáveis de saída.
<code>int get_medication_info(...)</code>	Similar à função anterior, primeiro encontra a consulta e depois navega pela lista de medicamentos para obter os dados do medicamento na posição desejada.
<code>int get_list_size(List *li)</code>	Retorna o tamanho atual da lista de consultas.

int get_consultation_med_count(...)	Percorre a lista de medicamentos de uma consulta específica para contar quantos itens ela contém.
int load_consultations(...)	Lê um arquivo de texto linha por linha, separando as informações de consultas e medicamentos com base no delimitador ';'. Ele insere cada consulta e seus respectivos medicamentos na lista.
insert_consultation_ptr	Função auxiliar que cria uma nova consulta a partir de um ponteiro para um nó de consulta existente e a insere na lista, sendo utilizada na busca.
insert_medication_to_consultation(...)	Função auxiliar que insere um novo medicamento diretamente em um nó de consulta, sem precisar de sua posição na lista, tornando o carregamento do arquivo mais eficiente.

2.1 Funções do arquivo display.c

Tabela 2: Funções presentes em display.c e suas descrições

Função	Descrição
clear_input_buffer()	Função utilitária para limpar o buffer de entrada do teclado, evitando erros de leitura com scanf e fgets.
display_menu()	Exibe o menu principal de opções para o usuário.
display_new_consultation(...)	Coleta os dados de uma nova consulta do usuário e chama a função de inserção para adicioná-la à lista.
display_list_consultations_meds	Lista as consultas cadastradas de forma simplificada, mostrando o índice, nome do paciente e descrição.
display_list_consultations(...)	Exibe uma lista completa, detalhada, de todas as consultas e seus medicamentos.
display_add_medication(...)	Permite ao usuário selecionar uma consulta existente e adicionar um novo medicamento a ela.
display_remove_consultation(..)	Solicita um índice de consulta e chama a função de remoção para excluir a consulta e todos os seus medicamentos.
void display_remove_medication(...)	Pede ao usuário que selecione uma consulta e um medicamento para remover o item específico da lista de medicamentos.

<code>void display_statistics(...)</code>	Calcula e exibe o número total de consultas e o número total de medicamentos. Também identifica e mostra qual consulta possui a maior quantidade de medicamentos.
<code>search_and_display_by_patient</code>	Solicita um nome de paciente e, em seguida, percorre a lista principal de consultas, exibindo na tela todas as consultas que correspondem à busca.

3. EXEMPLOS DE USO

A seguir, a demonstração de algumas operações do sistema, mostrando entradas de dados e as saídas esperadas.

3.1 Listar Consultas Carregadas do Arquivo

Quando o programa é iniciado, ele carrega automaticamente os dados do arquivo `consultations.txt`. A opção "3. Listar Todas as Consultas" mostrará todos os registros.

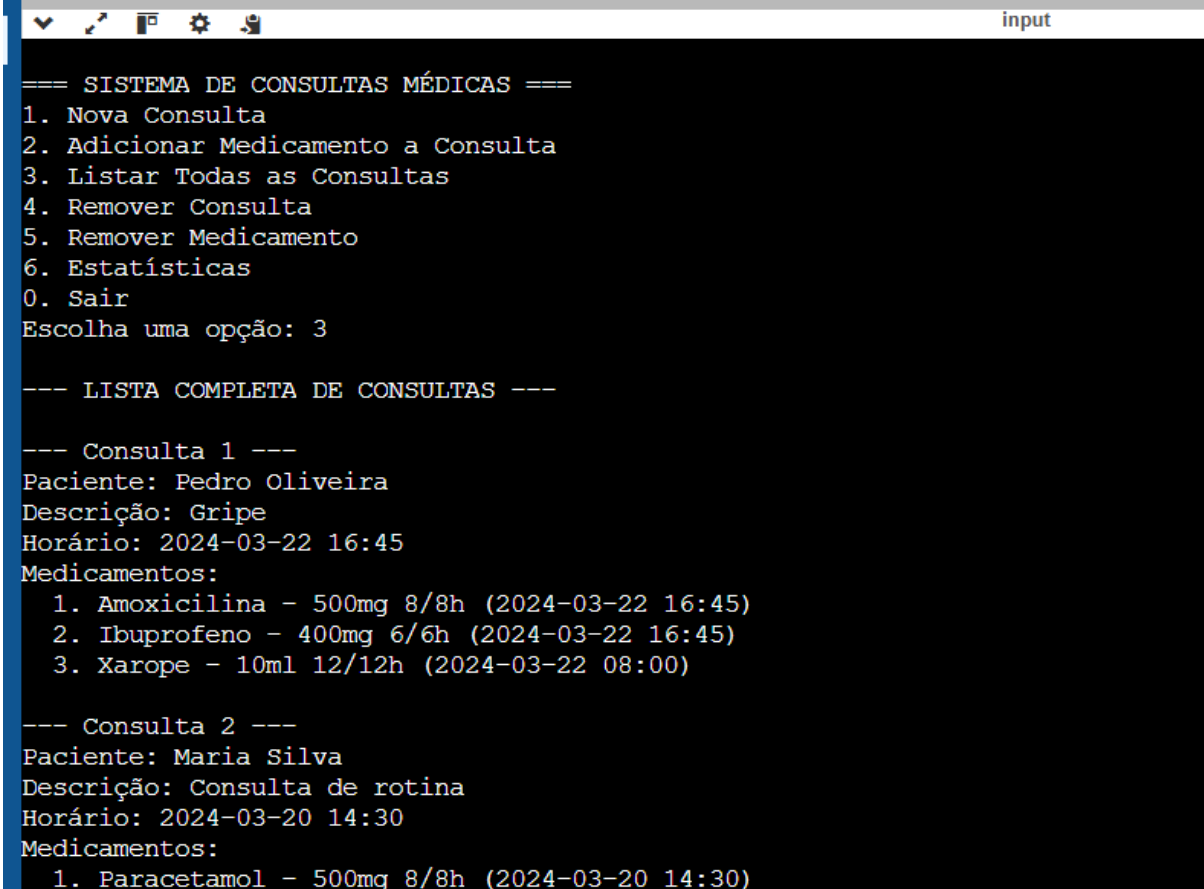
3.1.1 Código de Exemplo:

```
C
// Dentro do main()
//...
load_consultations(li, "consultations.txt");
display_list_consultations(li);
//...
```

3.1.2 Saída Obtida:

```
--- LISTA COMPLETA DE CONSULTAS ---
--- Consulta 1 ---
Paciente: Pedro Oliveira
Descrição: Gripe
Horário: 22-03-2024 08:00
Medicamentos:
1. Amoxicilina - 500mg (8/8h)
```

Figura 1- Saída obtida após selecionar a opção "3.Listar Todas as Consultas"



```
input
=== SISTEMA DE CONSULTAS MÉDICAS ===
1. Nova Consulta
2. Adicionar Medicamento a Consulta
3. Listar Todas as Consultas
4. Remover Consulta
5. Remover Medicamento
6. Estatísticas
0. Sair
Escolha uma opção: 3

--- LISTA COMPLETA DE CONSULTAS ---

--- Consulta 1 ---
Paciente: Pedro Oliveira
Descrição: Gripe
Horário: 2024-03-22 16:45
Medicamentos:
  1. Amoxicilina - 500mg 8/8h (2024-03-22 16:45)
  2. Ibuprofeno - 400mg 6/6h (2024-03-22 16:45)
  3. Xarope - 10ml 12/12h (2024-03-22 08:00)

--- Consulta 2 ---
Paciente: Maria Silva
Descrição: Consulta de rotina
Horário: 2024-03-20 14:30
Medicamentos:
  1. Paracetamol - 500mg 8/8h (2024-03-20 14:30)
```

3.2 Adicionar Novo Medicamento a uma Consulta

O usuário pode adicionar um novo medicamento a uma consulta já existente.

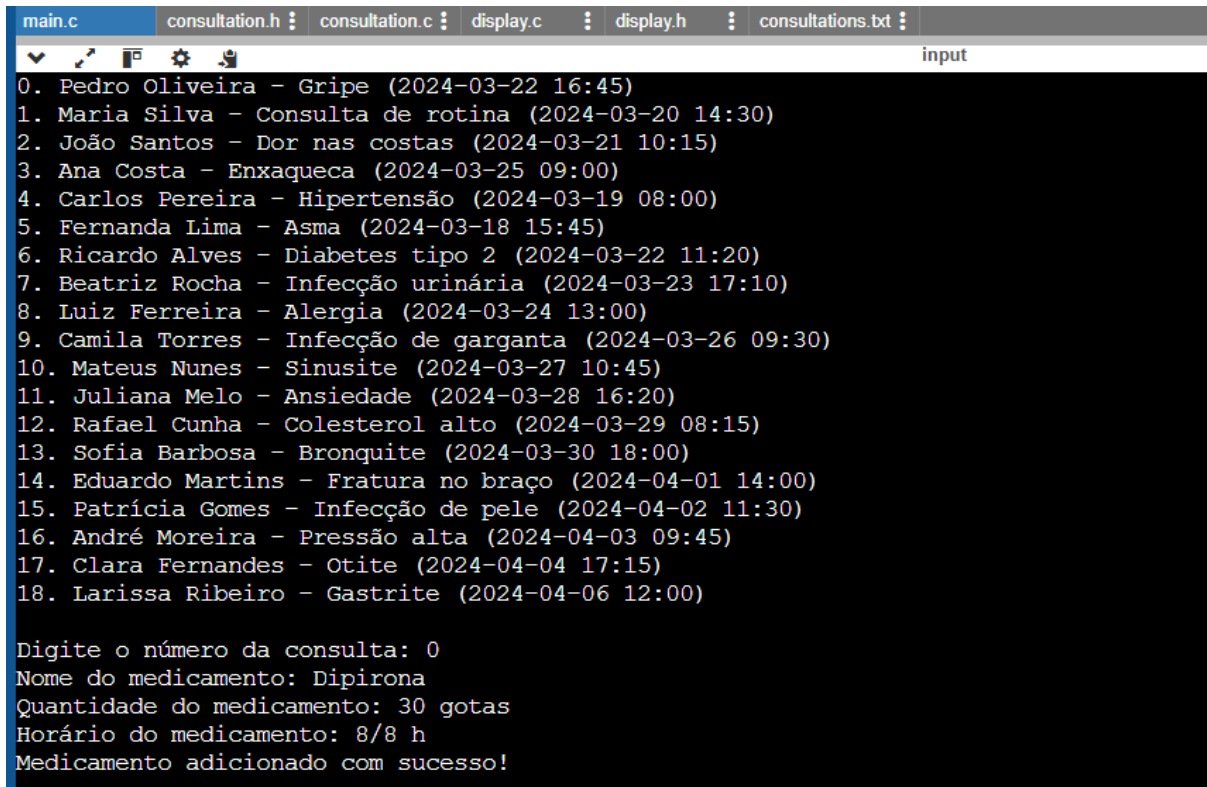
3.2.1 Entrada de Dados (Usuário):

```
Menu principal: Escolha a opção 2
--- ADICIONAR MEDICAMENTO ---
--- CONSULTAS DISPONÍVEIS ---
0. Pedro Oliveira - Gripe (2024-03-22 16:45)
1. Maria Silva - Consulta de rotina (2024-03-20 14:30)
...
Digite o número da consulta: 0
Nome do medicamento: Rinosoro
Quantidade do medicamento: 1 jato se necessário
Horário do medicamento: 2024-03-22 17:00
```

3.2.2 Saída Obtida:

Medicamento adicionado com sucesso!

Figura 2- Saída obtida após adicionar um medicamento a consulta



```
main.c consultation.h : consultation.c : display.c : display.h : consultations.txt : input
0. Pedro Oliveira - Gripe (2024-03-22 16:45)
1. Maria Silva - Consulta de rotina (2024-03-20 14:30)
2. João Santos - Dor nas costas (2024-03-21 10:15)
3. Ana Costa - Enxaqueca (2024-03-25 09:00)
4. Carlos Pereira - Hipertensão (2024-03-19 08:00)
5. Fernanda Lima - Asma (2024-03-18 15:45)
6. Ricardo Alves - Diabetes tipo 2 (2024-03-22 11:20)
7. Beatriz Rocha - Infecção urinária (2024-03-23 17:10)
8. Luiz Ferreira - Alergia (2024-03-24 13:00)
9. Camila Torres - Infecção de garganta (2024-03-26 09:30)
10. Mateus Nunes - Sinusite (2024-03-27 10:45)
11. Juliana Melo - Ansiedade (2024-03-28 16:20)
12. Rafael Cunha - Colesterol alto (2024-03-29 08:15)
13. Sofia Barbosa - Bronquite (2024-03-30 18:00)
14. Eduardo Martins - Fratura no braço (2024-04-01 14:00)
15. Patrícia Gomes - Infecção de pele (2024-04-02 11:30)
16. André Moreira - Pressão alta (2024-04-03 09:45)
17. Clara Fernandes - Otite (2024-04-04 17:15)
18. Larissa Ribeiro - Gastrite (2024-04-06 12:00)

Digite o número da consulta: 0
Nome do medicamento: Dipirona
Quantidade do medicamento: 30 gotas
Horário do medicamento: 8/8 h
Medicamento adicionado com sucesso!
```

3.3 Remover uma Consulta Completa

Removendo a primeira consulta de índice 0, porém, a mesma remoção pode ocorrer em qualquer parte da lista duplamente encadeada.

3.3.1 Entrada de Dados (Usuário):

Menu principal: Escolha a opção 4
--- REMOVER CONSULTA ---
--- CONSULTAS DISPONÍVEIS ---
0. Pedro Oliveira - Gripe (2024-03-22 16:45)
1. Maria Silva - Consulta de rotina (2024-03-20 14:30)
...
Digite o número da consulta a remover: 0

3.3.2 Saída Obtida:

Consulta removida com sucesso!

Figura 3- Saída obtida após remover uma consulta por completo



```
main.c consultation.h consultation.c display.c display.h consultations.txt input
--- REMOVER CONSULTA ---

--- CONSULTAS DISPONÍVEIS ---
0. Pedro Oliveira - Gripe (2024-03-22 16:45)
1. Maria Silva - Consulta de rotina (2024-03-20 14:30)
2. João Santos - Dor nas costas (2024-03-21 10:15)
3. Ana Costa - Enxaqueca (2024-03-25 09:00)
4. Carlos Pereira - Hipertensão (2024-03-19 08:00)
5. Fernanda Lima - Asma (2024-03-18 15:45)
6. Ricardo Alves - Diabetes tipo 2 (2024-03-22 11:20)
7. Beatriz Rocha - Infecção urinária (2024-03-23 17:10)
8. Luiz Ferreira - Alergia (2024-03-24 13:00)
9. Camila Torres - Infecção de garganta (2024-03-26 09:30)
10. Mateus Nunes - Sinusite (2024-03-27 10:45)
11. Juliana Melo - Ansiedade (2024-03-28 16:20)
12. Rafael Cunha - Colesterol alto (2024-03-29 08:15)
13. Sofia Barbosa - Bronquite (2024-03-30 18:00)
14. Eduardo Martins - Fratura no braço (2024-04-01 14:00)
15. Patrícia Gomes - Infecção de pele (2024-04-02 11:30)
16. André Moreira - Pressão alta (2024-04-03 09:45)
17. Clara Fernandes - Otite (2024-04-04 17:15)
18. Larissa Ribeiro - Gastrite (2024-04-06 12:00)

Digite o número da consulta a remover: 0
Consulta removida com sucesso!
```

4. CONCLUSÃO

A implementação deste sistema de gerenciamento de consultas médicas utilizando listas duplamente encadeadas permitiu a consolidação de conceitos fundamentais de Algoritmos e Estruturas de Dados. O maior desafio enfrentado foi o gerenciamento de memória e ponteiros, especialmente na implementação das funções de remoção. Garantir que os ponteiros fossem atualizados corretamente após a remoção de um nó, e que a memória de todos os nós aninhados fosse liberada para evitar vazamentos de memória, exigiu atenção extra aos detalhes.

A principal lição aprendida foi a importância de uma estrutura de dados adequada para resolver um problema. A escolha de listas duplamente encadeadas para as consultas e para os medicamentos se mostrou eficiente e intuitiva. Além disso, a prática de dividir o código em módulos (.c e .h) demonstrou ser uma metodologia importante para a organização e escalabilidade do projeto. O trabalho

proporcionou uma compreensão acerca da manipulação de ponteiros e da aplicação prática de estruturas de dados dinâmicas na resolução de problemas do mundo real.