Sistema de Gestão do Espaço Social

Relatório Técnico

Pedro Vilas Boas - 25453
Ricardo Marques - 25447

5 de janeiro de 2025

Resumo

Este trabalho apresenta o desenvolvimento de um sistema de gestão do espaço social, implementado em linguagem C. O sistema foi desenvolvido como parte da unidade curricular de Laboratórios de Informática, com foco em boas práticas de programação e estruturas de dados eficientes. O projeto utiliza estruturas de dados dinâmicas (listas ligadas) para gerir informações sobre funcionários, ementas e escolhas de refeições. São apresentadas as principais funcionalidades implementadas, incluindo gestão de funcionários, controlo de refeições e cálculo de médias de calorias, bem como os algoritmos e estruturas de dados utilizados em cada componente do sistema.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Objectivos

Este trabalho representa a continuação e evolução do projeto desenvolvido na Unidade Curricular de Programação Imperativa, agora no contexto da disciplina de Laboratórios Informáticos. O objectivo principal é melhorar e expandir o sistema anteriormente desenvolvido, aplicando conceitos avançados de desenvolvimento de programação e boas práticas de engenharia.

1.2 Descrição do Sistema

O sistema de gestão do espaço social foi desenvolvido para satisfazer as necessidades de controlo e organização de refeições num ambiente empresarial. O projecto foi implementado em linguagem C, utilizando conceitos avançados de programação como alocação dinâmica de memória, estruturas de dados ligadas e manipulação de ficheiros.

1.2.1 Funcionalidades Principais

1. **Gestão de Funcionários**
    - Carregar dados de funcionários
    - Listar informações dos funcionários
2. **Controlo de Ementas**
    - Carregar dados de ementas
    - Listar informações de ementas

3. **Gestão de Escolhas**

    - Carregar dados de escolhas dos funcionários
    - Listar refeições por dia
    - Listar utentes ordenados
    - Consultar refeições de um utente
    - Calcular médias de calorias

Capítulo 2

Desenvolvimento

2.1 Descrição do Problema

O problema a ser resolvido envolve a criação de um sistema de gestão do espaço social. O sistema deve permitir a entrada de dados através de ficheiros de texto, oferecendo funcionalidades como listagem de refeições, consulta de refeições por utente, e cálculo de médias de calorias.

2.2 Estruturas de Dados

O sistema utiliza estruturas de dados dinâmicas para gerir as informações de forma eficiente. As principais estruturas implementadas são:

2.2.1 Estrutura de Funcionários

A estrutura de funcionários armazena informações básicas sobre cada funcionário do sistema:

Listing 2.1: Estrutura de dados para Funcionários

```
typedef struct {
    int numero;         // Número único do funcionário
    char nome[50];      // Nome completo
    int nif;           // Número de Identificação Fiscal
    int telefone;      // Número de telefone
} Funcionario;
```

A estrutura do nó da lista encadeada para funcionários é definida da seguinte forma:

Listing 2.2: Estrutura do Nó da Lista de Funcionários

```
typedef struct NodeFunc {
    Funcionario func;           // Dados do funcionário
    struct NodeFunc* prox;      // Ponteiro para o pró
       ximo nó
} NodeFunc;
```

2.2.2 Estrutura de Ementas

Para gerir as ementas diárias, foi implementada a seguinte estrutura:

Listing 2.3: Estrutura de dados para Ementas

```
typedef struct {
    char dia[10];   // Dia da semana (ex: 2feira)
    char data[10]; // Data no formato DD.MM.AAAA
    struct {
        char nome[50]; // Nome do prato de carne
        int calorias;  // Calorias do prato de carne
    } prato_carne;
    struct {
        char nome[50]; // Nome do prato de peixe
        int calorias;  // Calorias do prato de peixe
    } prato_peixe;
} Ementa;
```

A estrutura do nó da lista encadeada para ementas é definida da seguinte forma:

Listing 2.4: Estrutura do Nó da Lista de Ementas

```
typedef struct NodeEmenta {
    Ementa ementa;
    struct NodeEmenta* prox;
} NodeEmenta;
```

2.2.3 Estrutura de Escolhas

As escolhas dos utentes são registadas usando:

Listing 2.5: Estrutura de dados para Escolhas

```
typedef struct {
    char dia[10];          // Dia da semana (ex: 2feira)
    int num_funcionario;     // Número do funcionário
```

```
4     char tipo_prato;      // 'C' para carne, 'P' para
          peixe
5 } Escolha;
```

A estrutura do nó da lista encadeada para escolhas é definida da seguinte forma:

Listing 2.6: Estrutura do Nó da Lista de Escolhas

```
1 typedef struct NodeEscolha {
2     Escolha escolha;
3     struct NodeEscolha* prox;
4 } NodeEscolha;
```

Capítulo 3

Implementação

3.1 Organização do Código

O código fonte está organizado em vários ficheiros:

- `estruturas.h`: Definição das estruturas de dados principais
- `funcoes.c`: Implementação das funções principais
- `main.c`: Programa principal e interface com utilizador

3.1.1 Estruturas de Dados

As principais estruturas implementadas são:

- **Funcionario**: Armazena dados dos funcionários (número, nome, NIF, telefone).
- **Ementa**: Mantém informações das refeições (dia da semana, data, nome e calorias dos pratos).
- **Escolha**: Regista escolhas de refeições (dia da semana, número do funcionário, tipo de prato ('C' para carne, 'P' para peixe)).

3.1.2 Funções Principais

- **Gestão de Funcionários**:
  - `carregarFuncionarios`: Carrega os dados dos funcionários do ficheiro (O(n)). Esta função lê o ficheiro especificado, processa cada linha para extrair os dados do funcionário e cria um novo

nó na lista encadeada de funcionários. Retorna o primeiro nó da lista.

  - `criarNoFunc`: Cria um novo nó da lista de funcionários (O(1)). Esta função aloca dinamicamente a memória para um novo nó da lista de funcionários e inicializa seus campos com os dados do funcionário passados como parâmetro.

- **Gestão de Ementas**:

  - `carregarEmentas`: Carrega os dados das ementas do ficheiro (O(n)). Esta função lê o ficheiro especificado, processa cada linha para extrair os dados da ementa e cria um novo nó na lista encadeada de ementas. Retorna o primeiro nó da lista.

- **Gestão de Escolhas**:

  - `carregarEscolhas`: Carrega os dados das escolhas do ficheiro (O(n)). Esta função lê o ficheiro especificado, processa cada linha para extrair os dados de escolha do funcionário e cria um novo nó na lista encadeada de escolhas. Retorna o primeiro nó da lista.

3.1.3 Gestão de Memória

- Alocação dinâmica para todas as estruturas usando `malloc`.
- Libertação adequada de memória ao finalizar o programa através da função `liberarMemoria`.
- A verificação de erros em operações de memória é realizada através da comparação do resultado da função `fopen` com `NULL`.

3.1.4 Validações

- O programa valida se os arquivos foram abertos com sucesso através da função `fopen`, retornando NULL caso a abertura do ficheiro falhe.
- A escolha das opções do menu é validada através da comparação da entrada do utilizador com os casos definidos no `switch` da função `main`.
- No carregamento de dados dos ficheiros, nenhuma validação dos dados é realizada, apenas é feita a leitura e o seu armazenamento nas estruturas.

- Os dados introduzidos pelo utilizador (dia da semana, data inicial e final) são lidos através da função `scanf`.

Ficheiro	Responsabilidade
main.c	Interface com utilizador, menu principal e controlo de fluxo do programa
estruturas.h	Definição das estruturas de dados e tipos utilizados, além de protótipos das funções
funcoes.c	Implementação das operações principais do sistema

Figura 3.1: Organização dos ficheiros do projeto

3.1.5 Principais Algoritmos

As principais operações são baseadas em percursos nas listas ligadas.

Carregamento de Dados

O sistema implementa um algoritmo que percorre o ficheiro de dados linha a linha.

Listing 3.1: Algoritmo de carregamento de dados

```
NodeFunc* carregarFuncionarios(const char* filename) {
    NodeFunc* head = NULL;
    NodeFunc* atual = NULL;
    FILE* file = fopen(filename, "r");
    char linha[256];

    if (file == NULL) {
        printf("Erro ao abrir ficheiro %s\n", filename);
        return NULL;
    }

    while (fgets(linha, sizeof(linha), file)) {
        Funcionario func;
        // Parse da linha
        // Criar novo nó e inserir na lista
        if(parseLinha(linha, &func)){
            NodeFunc* novoNo = criarNoFunc(func);
            if (!head) {
                head = novoNo;
```

```
20                  atual = novoNo;
21              } else {
22                  atual->prox = novoNo;
23                  atual = novoNo;
24              }
25          }
26      }
27
28      fclose(file);
29      return head;
30  }
```

Este algoritmo, implementado na função `carregarFuncionarios`, inicia com a abertura do ficheiro especificado. Caso o ficheiro não possa ser aberto, retorna NULL. Em seguida, percorre linha a linha, usando a função `fgets`. Cada linha é lida, e um novo nó é criado e inserido na lista encadeada. A função retorna o primeiro nó da lista.

Cálculo de Estatísticas

O cálculo de médias é feito através de uma interação pelas listas de ementas e escolhas, de forma a que seja calculada a média de calorias por dia.

Listing 3.2: Algoritmo de cálculo de médias

```
1  void calcularMediasCalorias(NodeEmenta* ementas,
       NodeEscolha* escolhas,
2                              const char* data_inicio, const
                                   char* data_fim) {
3      // Para cada dia
4      NodeEmenta* ementa = ementas;
5      while (ementa) {
6          int total_calorias = 0;
7          int num_refeicoes = 0;
8          NodeEscolha* escolha = escolhas;
9          while (escolha) {
10             if (strcmp(escolha->escolha.dia, ementa->
                   ementa.dia) == 0) {
11                 if (escolha->escolha.tipo_prato == 'C')
12                     total_calorias += ementa->ementa.
                           prato_carne.calorias;
13                 else
14                     total_calorias += ementa->ementa.
                           prato_peixe.calorias;
```

```
15                  num_refeicoes++;
16              }
17              escolha = escolha->prox;
18          }
19          //Calcula e exibe a média
20          if (num_refeicoes > 0) {
21              float media = (float)total_calorias / num_
                    refeicoes;
22              printf("%-6s | %.2f\n", ementa->ementa.dia,
                    media);
23          }
24          ementa = ementa->prox;
25      }
26  }
```

Este algoritmo, implementado na função `calcularMediasCalorias`, itera pela lista de ementas, e para cada dia, percorre a lista de escolhas para calcular o total de calorias consumidas, utilizando os dados de calorias dos pratos da ementa. Em seguida, calcula a média de calorias para aquele dia, apresentando o resultado no output.

Capítulo 4

Exemplos de Uso do Sistema

4.1 Operações Básicas

4.1.1 Registo de Funcionário

Exemplo de como adicionar um novo funcionário ao sistema:

Listing 4.1: Exemplo de Registo de funcionário

```
// Dados do novo funcionário (este código é apenas
   exemplificativo, as informações são carregadas de
   ficheiros)
int numero = 1001;
char nome[] = "João Silva";
int nif = 123456789;
int telefone = 912345678;

// O novo nó é criado e adicionado à lista através da fun
   ção carregarFuncionarios
```

Este excerto de código demonstra como os dados de um funcionário (número, nome, NIF e telefone) são estruturados e carregados através do ficheiro `pessoas.txt` para a lista de funcionários, utilizando a função `carregarFuncionarios`.

4.1.2 Registo de Ementa

Exemplo de como registar uma nova ementa:

Listing 4.2: Exemplo de registo de ementa

```
// Dados da nova ementa (este código é apenas
   exemplificativo, as informações são carregadas de
   ficheiros)
```

```
char dia[] = "2feira";
char data[] = "05.01.2025";
char prato_carne[] = "Bacalhau à Brás";
int calorias_carne = 850;
char prato_peixe[] = "Salmão Grelhado";
int calorias_peixe = 700;
// O novo nó é criado e adicionado à lista através da fun
   ção carregarEmentas
```

Este exemplo ilustra como os dados de uma ementa (dia da semana, data e nome e calorias do prato de carne e peixe) são estruturados e carregados através do ficheiro `ementas.txt` para a lista de ementas, utilizando a função `carregarEmentas`.

4.1.3 Registo de Escolha

Exemplo de como registar a escolha de um funcionário:

Listing 4.3: Exemplo de registo de escolha

```
// Dados da escolha (este código é apenas exemplificativo
   , as informações são carregadas de ficheiros)
char dia[] = "2feira";
int num_func = 1001;
char tipo_prato = 'C'; // 'C' para carne e 'P' para peixe

// A nova escolha é adicionada à lista através da função
   carregarEscolhas
```

Este excerto de código mostra como os dados de escolha do funcionário (dia da semana, número do funcionário e tipo de prato) são estruturados e carregados através do ficheiro `menu_escolhido.txt` para a lista de escolhas, utilizando a função `carregarEscolhas`.

4.2 Casos de Uso Completos

4.2.1 Gestão Diária de Refeições

Exemplo de como o sistema pode ser usado para listar as refeições de um dia:

Primeiro, os ficheiros `pessoas.txt`, `ementas.txt` e `menu_escolhido.txt` são preenchidos da seguinte forma:

pessoas.txt

Listing 4.4: Exemplo do ficheiro pessoas.txt

```
1;João Silva;123456789;912345678
2;Maria Santos;987654321;923456789
3;Pedro Costa;456789123;934567890
```

Este ficheiro contém dados de exemplo para três funcionários, no formato: `numero;nome;nif;telefone`.

ementas.txt

Listing 4.5: Exemplo do ficheiro ementas.txt

```
2feira;05.01.2025;Bacalhau à Brás;850;Salmão Grelhado;700
3feira;06.01.2025;Bife com Arroz;950;Pescada Cozida;600
4feira;07.01.2025;Frango Assado;750;Dourada Grelhada;800
```

Este ficheiro contém dados de exemplo para as ementas de três dias da semana, no formato: `dia;data;prato_carne;calorias_carne;prato_peixe;calorias_peixe`.

menu_escolhido.txt

Listing 4.6: Exemplo do ficheiro menu_escolhido.txt

```
2feira;1;C
2feira;2;P
2feira;3;C
3feira;1;C
3feira;2;P
```

Este ficheiro contém dados de exemplo para as escolhas dos funcionários, no formato: `dia;num_funcionario;tipo_prato`.

De seguida, o programa é executado e no menu principal, é escolhida a opção 4 e de seguida é pedido o dia da semana, o utilizador insere `2feira` e o output na consola é:

Listing 4.7: Exemplo de output na consola

```
+============================================+
|         Refeicoes para 2feira              |
+============================================+
| Func. | Nome                  | Prato      |
+--------+-----------------------+------------+
| 1      | João Silva            | Bacalhau à Brás |
```

```
| 2       | Maria Santos           | Salmão Grelhado|
| 3       | Pedro Costa            | Bacalhau à Brás |
+---------+------------------------+-------------+
Total de refeicoes para 2feira: 3
+=============================================+
```

Este é um exemplo do output do programa, após o utilizador inserir a opção 4 e inserir o dia da semana "2feira". O programa lista as refeições para esse dia específico, mostrando o número do funcionário, o nome e o prato escolhido.

Capítulo 5

Conclusão

O desenvolvimento deste sistema trouxe diversas contribuições importantes:

- Implementação de um sistema completo de gestão de refeições.
- Utilização de estruturas de dados dinâmicas para gerir informações.
- Desenvolvimento de algoritmos para processamento de dados.
- Aplicação prática de conceitos de programação em C.

5.1 Desafios Encontrados

Durante o desenvolvimento, encontrámos vários desafios:

- Gestão de memória em estruturas dinâmicas.
- Validação e tratamento de dados de entrada.
- Implementação de algoritmos.
- Manipulação de ficheiros.

5.2 Valor Acrescentado

O projeto proporcionou várias melhorias em relação à versão anterior:

- Melhor organização do código.
- Interface mais intuitiva.
- Maior eficiência no processamento de dados.
- Melhor gestão de memória.

5.3 Trabalho Futuro

Para futuras versões do sistema, sugerimos:

- Implementação de uma interface gráfica.
- Adição de funcionalidades de exportação de relatórios.
- Otimização adicional dos algoritmos de pesquisa.
- Implementação de um sistema de cópias de segurança automáticas.