**TRABALHO PRÁTICO**

**FASE 1**

***Estruturas de dados avançadas***

***Docente:***

Luís Gonzaga Martins Ferreira

***Aluno:***

Rodrigo Lopes Ferreira Nº 31525

**1. Introdução**

Esta fase do trabalho prático foi desenvolvida no âmbito da unidade curricular de Estruturas de Dados Avançados.

O principal objetivo foi aplicar os conhecimentos teóricos e práticos adquiridos ao longo das aulas, através da criação e implementação de funções para gestão de antenas numa cidade.

O projeto inclui funções de inserção, remoção e listagem de antenas, e também carregamento de uma matriz de antenas predefinida pelo professor no enunciado do trabalho para uma lista ligada.

**2. Objetivos do Projeto**

A Fase 1 consistiu na implementação de uma lista ligada simples para representar antenas, cada uma caracterizada por uma frequência (um char) e coordenadas (x, y) numa matriz.

Os dados deveriam ser carregados a partir de um ficheiro de texto, com suporte para operações como inserção, remoção e listagem de antenas.

Adicionalmente, era requerida a identificação de localizações com efeito nefasto, definidas como pontos alinhados com duas antenas da mesma frequência, onde uma está a dupla distância da outra.

**3. Implementação**

**3.1 Definição das Structs**

Foi definida uma struct Antena para armazenar as informações de cada antena:

* char frequencia: representa a frequência da antena (A ou B);
* int x, y: indica as coordenadas x e y das antenas na matriz;
* Antena\* proxima: ponteiro para o elemento seguinte na lista ligada.

Foi definida uma struct Antena para armazenar as informações de cada antena:

* char frequencia: representa a frequência da antena (A ou B);
* int x, y: indica as coordenadas x e y das antenas na matriz;
* Antena\* proxima: ponteiro para o elemento seguinte na lista ligada.

**3.2 Funcionalidades Desenvolvidas**

As seguintes funcionalidades foram implementadas:

1. **Carregamento de Dados**: A função carregarAntenas lê um ficheiro de texto (e.g., "antenas.txt") e constrói a lista ligada, associando cada carácter não '.' (ponto) a uma antena nas respetivas coordenadas.
2. **Inserção**: A função inserirAntena adiciona uma nova antena no início da lista, utilizando alocação dinâmica de memória.
3. **Remoção**: A função removerAntena elimina uma antena com base nas suas coordenadas, ajustando os ponteiros e libertando memória.
4. **Listagem**: O programa inclui um menu interativo que apresenta as antenas numa tabela na consola, com opções para diferentes formatos.

O código foi organizado em 4 ficheiros, sendo estes **main.c, funcoes.c, funcoes.h, struct.h**.Existe também o ficheiro **antenas.txt** que tinha a matriz e o **Makefile**, que ajuda na execução do programa.. O código está todo documentado com Doxygen.

**4. Limitações**

A dedução das localizações com efeito nefasto não foi implementada.

Devido a constrangimentos de tempo e devido a não ficar a perceber o que era pedido, deixei esta função por fazer, e coloquei apenas referências da função nos ficheiros, para mais tarde durante a realização da fase 2 poder fazer e compreender melhor

**5. Conclusão**

A Fase 1 do projeto permitiu consolidar conhecimentos sobre listas ligadas e gestão de memória em C, cumprindo a maioria dos requisitos estabelecidos.

As operações de carregamento, inserção, remoção e listagem foram implementadas com sucesso, demonstrando a aplicação prática de structs dinâmicas.

Contudo, a ausência do cálculo do efeito nefasto constitui uma limitação relevante. Este trabalho servirá como uma base sólida para futuros projetos.

**6. Bibliografia**

**ChatGPT**

<https://chatgpt.com>

**Grok AI**

<https://grok.com>

**Listas Ligadas (outra explicação para perceber melhor)**

<https://www.youtube.com/watch?v=N6dOwBde7-M>

**Menu de opções do ficheiro main.c**

<https://github.com/LFtech6/PI-tp>

**Inspiração de código**

<https://github.com/luferIPCA/EDA-LESI-2024-2025/tree/master/Aulas>

**Repos GitHub**

<https://github.com/LFtech6/EDA_TP>