IPCA

Relatório do Projeto de Estruturas de Dados Avançadas

*Relatório*

*FASE 1*



Curso: Licenciatura Engenharia de Sistemas informáticos

Estruturas de Dados Avançadas

Autor: Rodrigo Henrique Silva Pinheiro

Docente: Lufer

[Capítulo 1. 3](#_Toc194249202)

[Introdução 3](#_Toc194249203)

[Capítulo 2 4](#_Toc194249204)

[Desenvolvimento 4](#_Toc194249205)

[2.1 Conceito do programa 4](#_Toc194249206)

[Estrutura do Projeto 5](#_Toc194249207)

[2.2 Arquivos Presentes 5](#_Toc194249208)

[2.2.1 Análise e Compreensão do Código 6](#_Toc194249209)

[Conclusão e observações finais 18](#_Toc194249210)

[Bibliografia 19](#_Toc194249211)

# Capítulo 1.

## Introdução

O presente relatório foi elaborado no âmbito da Unidade Curricular de Estruturas de Dados Avançadas do curso de Licenciatura em Engenharia de Sistemas Informáticos, durante o ano letivo de 2024/2025. Este trabalho tem como objetivo consolidar e melhorar competências práticas relacionadas com o desenvolvimento de um projeto utilizando a linguagem de programação C. Ao longo deste do desenvolvimento deste projeto são adotadas boas praticas de programação, incluindo modularização de código, o uso de ferramentas de automação como Makefile e a documentação utilizando ferramentas como o Doxygen, para documentação devida do código. Pretende-se também expandir os conhecimentos em linguagem de Programação, em uma parte mais especifica voltada as listas ligadas simples. Uso também do processo de Debug, ferramenta de auxílio para resolver problemas no código.

* Definição do problema e de sua análise;
* Implementação de funções;
* Uso de Listas Ligadas Simples;
* Resultados obtidos;
* Considerações finais;

# Capítulo 2

## Desenvolvimento

### 2.1 Conceito do programa

O programa considera uma cidade com várias antenas. Cada antena é sintonizada numa frequência específica, e a partir dessas frequências distintas, são detetadas as do mesmo tipo, e que estão mais próximas e a partir dai são calculadas zonas que podem surgir interferências.

**Principais funcionalidades:**

* Carregamento de dados: Leitura de um ficheiro com o um modelo específico de uma cidade, com as suas antenas.
* Gestão e Visualização: Listagem das Coordenadas das Antenas, dos Efeitos Nefastos.
* Implementação de inserção de Antenas novas na cidade.

O código como já referido, é documento em Doxygen, com um Makefile para compilação automatizada.

## Estrutura do Projeto

### 2.2 Arquivos Presentes

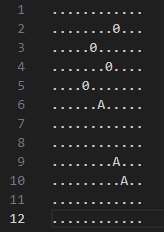
Dados .h – Definição de estruturas de dados

Funções .c - Implementação e desenvolvimento de funções declaradas anteriormente.

Makefile – Automatiza o processo de compilação, otimizando o processo de testes, e a realização do trabalho o que é muito importante para não perder tempo com ações que não influenciam o resultado final.

Main.c – Coordena a execução do programa, esta parte é a única que interage com o utilizador, esta parte também consiste na parte gráfica, onde são apresentadas as informações necessárias, tal como listagens e erros já prevenidos.

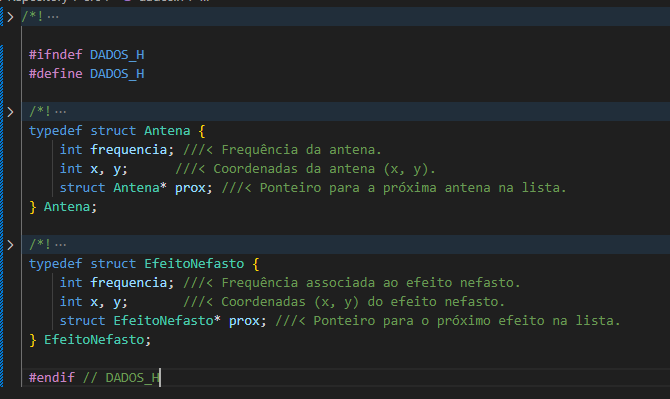
Antenas.txt – Contém o formato da cidade, com a sua estrutura de antenas a ser analisada pelo programa.



1. Antena com frequência A 0- Antena com frequência 0 . – Representa os espaços vazios entre antenas

### 2.2.1 Análise e Compreensão do Código

Dados .h



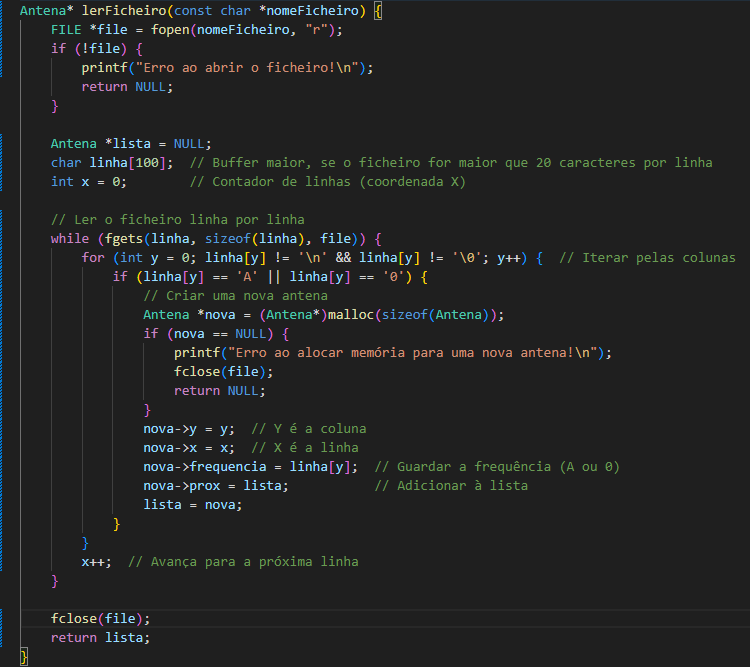
Este ficheiro, dados.h, define duas estruturas fundamentais para a modelagem do problema: **Antena** e **EfeitoNefasto**.

* A estrutura **Antena** representa uma antena de transmissão, que contem informações sobre a frequência e a posição no espaço bidimensional (coordenadas x, y). Além disso, utiliza um ponteiro para permitir a implementação de uma lista encadeada de antenas.
* A estrutura **EfeitoNefasto** modela interferências ou perturbações no ambiente, armazenando dados similares aos da antena, incluindo frequência e posição, além de um ponteiro para possibilitar a organização dos efeitos em uma lista encadeada.

Esse cabeçalho fornece a base para manipular dinamicamente as entidades dentro do sistema, permitindo armazenar, acessar e interligá-las de forma eficiente.

Funções .c

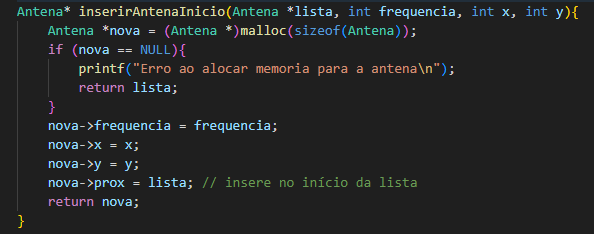
Dentro deste ficheiro, são apresentadas cerca de sete funções nesta versão do projeto.



Função: lerFicheiro()

Dentro deste temos a função lerFicheiro lê um ficheiro de texto contendo uma matriz de caracteres e constrói uma lista encadeada de antenas com base nos valores encontrados.

* Abre o ficheiro e verifica se foi carregado corretamente.
* Percorre linha por linha, associando a coordenada **x** à linha e **y** à coluna.
* Sempre que encontra um caractere **'A'** ou **'0'**, cria dinamicamente uma nova estrutura **Antena**, armazenando sua posição e frequência.
* Insere a nova antena no início da lista encadeada.
* Retorna a lista encadeada contendo todas as antenas encontradas no ficheiro.

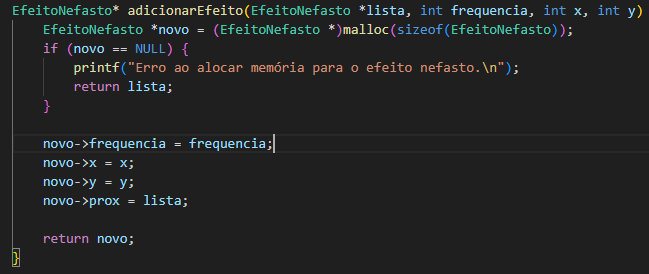


Função: inserirAntenaInicio()

A função inserirAntenaInicio adiciona uma nova antena no início de uma lista encadeada.

**Funcionamento:**

* **Aloca dinamicamente** memória para uma nova antena.
* **Verifica** se a alocação foi bem-sucedida.
* **Preenche** a estrutura com a frequência e coordenadas fornecidas.
* **Insere** a nova antena no início da lista, ajustando o ponteiro.
* **Retorna** o novo início da lista com a antena recém-adicionada.



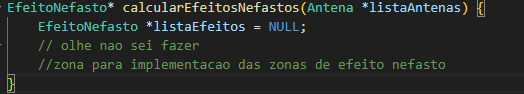
Função: adicionarEfeito

A função adicionarEfeito adiciona um novo efeito nefasto no início de uma lista encadeada.

**Funcionamento:**

* **Aloca dinamicamente** memória para um novo efeito nefasto.
* **Verifica** se a alocação foi bem-sucedida.
* **Preenche** a estrutura com a frequência e coordenadas fornecidas.
* **Insere** o novo efeito no início da lista, ajustando o ponteiro.
* **Retorna** o novo início da lista com o efeito recém-adicionado.

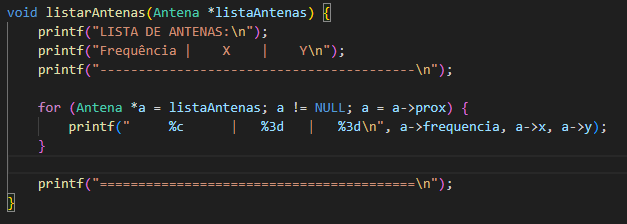
**Esta é relativamente parecida à função de inserir a antena a logica utilizada é a mesma.**



Função: calcularEfeitosNefastos

Tentativa de uma função não implementada ainda que, localiza os locais com efeitos nefasto, segundo a posição de antenas com a mesma frequência, estando na mesma linha, coluna ou ate mesma diagonal.

Uma função que será revista na fase dois do projeto.

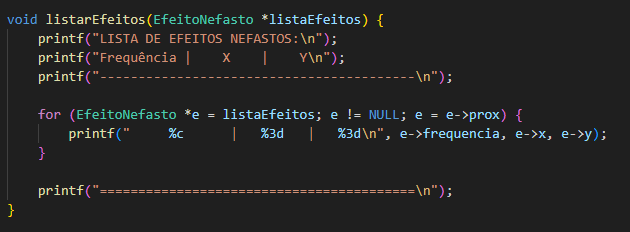


Função: listarAntenas

Esta função percorre e mostra a lista de antenas armazenadas.

**Funcionamento:**

* Mostra um cabeçalho indicando que se trata da **lista de antenas**.
* Imprime uma tabela com os títulos das colunas (**Frequência, X e Y**).
* Percorre a lista encadeada de antenas, imprimindo a frequência e as coordenadas de cada antena.
* Finaliza com uma linha de separação para melhor visualização.



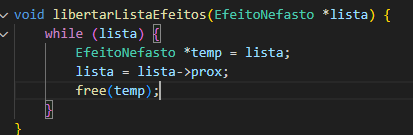
Função: listarEfeitos

Esta função percorre e mostra a lista de efeitos nefastos armazenados.

**Funcionamento:**

* Mostra um cabeçalho indicando que se trata da **lista de efeitos nefastos**.
* Imprime uma tabela semelhante à das antenas, com os títulos das colunas (**Frequência, X e Y**).
* Percorre a lista encadeada de efeitos nefastos, imprimindo as informações de cada um.
* Finaliza com uma linha de separação para organizar a saída.

Ambas as funções têm a mesma estrutura, diferenciando-se apenas na lista que percorrem (antenas ou efeitos nefastos).

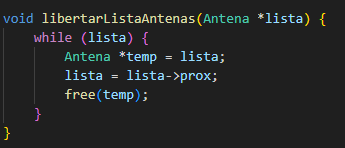


Função: libertarListaEfeitos

Esta função liberta toda a memória alocada dinamicamente para a lista de antenas.

**Funcionamento:**

* Percorre a lista de antenas enquanto ela não estiver vazia.
* Guarda temporariamente o ponteiro para o primeiro nó (**temp**).
* Avança para o próximo nó da lista.
* Liberta a memória do nó armazenado em **temp** usando free().
* Repete o processo até que todos eles sejam liberados.



Função: libertarListaAntenas

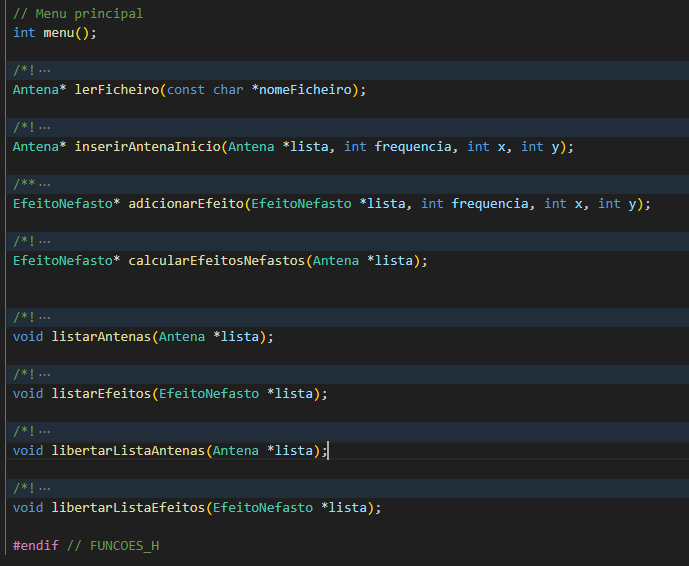
Esta função liberta toda a memória alocada dinamicamente para a lista de efeitos nefastos.

**Funcionamento:**

* Funciona de forma idêntica à função libertarListaAntenas, mas aplicada à lista de efeitos nefastos.
* Percorre a lista, armazenando temporariamente o primeiro nó em **temp**.
* Avança para o próximo nó e liberta a memória do nó anterior.
* Continua até que todos eles tenham sido removidos.

Ambas as funções são essenciais para evitar vazamento de memória ao finalizar o programa.

Funções .h



Este ficheiro define os protótipos das funções utilizadas no programa.

* **menu()** → Exibe o menu principal.
* **lerFicheiro()** → Lê um ficheiro e carrega as antenas numa lista.
* **inserirAntenaInicio()** → Adiciona uma antena no início da lista.
* **adicionarEfeito()** → Adiciona um efeito nefasto na lista.
* **calcularEfeitosNefastos()** → Gera a lista de efeitos nefastos com base nas antenas.
* **listarAntenas()** → Exibe todas as antenas armazenadas.
* **listarEfeitos()** → Exibe todos os efeitos nefastos armazenados.
* **libertarListaAntenas()** → Liberta a memória da lista de antenas.
* **libertarListaEfeitos()** → Liberta a memória da lista de efeitos nefastos.

Este ficheiro serve apenas para declarar as funções que serão implementadas noutro ficheiro.

Main .c

Neste ficheiro o código implementa um programa interativo baseado em menus para gerir uma lista de antenas e os respetivos efeitos nefasto.

É de se lembrar que esta é a única parte que interage com o utilizador

**Estrutura do código:**

1. **Menu (menu())**
   * Apresenta um conjunto de opções ao utilizador e retorna a opção escolhida.
2. **Função main()**
   * Inicializa as listas de antenas e efeitos nefastos como NULL.
   * Executa um ciclo do-while para manter o menu ativo até que o utilizador escolha sair.
   * Utiliza um switch para tratar cada opção do menu:
     + **Opção 1:** Pede ao utilizador o nome do ficheiro, liberta a lista de antenas antiga (se existir) e carrega uma nova lista.
     + **Opção 2:** Permite inserir uma nova antena na lista, pedindo os dados ao utilizador.
     + **Opção 3:** Calcula os efeitos nefastos com base na lista de antenas existente.
     + **Opção 4:** Lista todas as antenas carregadas.
     + **Opção 5:** Lista todos os efeitos nefastos calculados.
     + **Opção 6 e 7:** Libertam a memória das listas de antenas e efeitos, respetivamente, e evitam acessos inválidos ao definir os ponteiros como NULL.
     + **Opção 8:** Sai do programa.
     + **Default:** Trata entradas inválidas.
3. **Libertação de memória**
   * Antes de sair, o programa certifica-se de que toda a memória alocada dinamicamente é libertada para evitar fugas de memória.

O código garante um funcionamento seguro e estruturado, prevenindo acessos inválidos e libertando corretamente os recursos utilizados.

# Conclusão e observações finais

Este projeto teve como objetivo o desenvolvimento de um sistema para gerir antenas e calcular efeitos nefastos associados a elas. Através da implementação de listas encadeadas, conseguimos organizar e manipular eficientemente as antenas e os efeitos nefastos, permitindo ao utilizador carregar, inserir, listar e libertar dados conforme necessário. O uso de um menu interativo tornou o programa fácil de usar, proporcionando uma boa experiência ao utilizador.

Embora a maior parte das funcionalidades tenha sido implementada com sucesso, houve dificuldades na implementação da função de **calcular efeitos nefastos**. Esta função deveria ser responsável por identificar os locais onde as antenas geram interferência, considerando suas frequências e posições. No entanto, não consegui completar essa parte do projeto devido a problemas de lógica na forma como as antenas interagem entre si e como os efeitos nefastos deveriam ser calculados com base nessas interações.

Futuros desenvolvimentos do projeto poderiam focar na revisão e otimização dessa função, considerando um modelo mais claro para as interações entre antenas e seus efeitos na área ao redor delas, além de uma análise mais detalhada sobre como calcular as áreas de interferência de forma eficiente.

# Bibliografia

Link do Repositório GIT:

<https://github.com/Rodriguiinhoo/MAIN-Repository.git>

https://chatgpt.com

Link dos Trabalhos Feitos na aula que ajudaram na Resolução do Projeto:

https://github.com/luferIPCA/EDA-LESI-2024-2025.git